NTNU – Trondheim
Norwegian University of
Science and Technology

Methods for Analysis of Big Data.

A Combination of the Lasso Method and the
Case-Crossover Design for Investigating
Potential Drug Side Effects on Myocardial
Infarction

Ioannis Vardaxis

Master of Science in Mathematics (for international students)
Submission date: May 2014
Supervisor: Bo Henry Lindqvist, MATH

Norwegian University of Science and Technology
Department of Mathematical Sciences

”Ἑσπερίδας θ’, ᾗς μῆλα πέρην κλυτοῦ Ὠκεανοῖο
χρύσεα καλὰ μέλουσι φέροντά τε δένδρεα καρπόν.
Καὶ Μοίρας καὶ Κῆρας ἐγείνατο νηλεοποίνους,
[Κλωθώ τε Λάχεσίν τε καὶ Ἄτροπον, αἵτε βροτοῖσι
γεινομένοισι διδοῦσιν ἔχειν ἀγαθόν τε κακόν τε,]
αἵτ’ ἀνδρῶν τε θεῶν τε παραιβασίας ἐφέπουσιν·
οὐδέ ποτε λήγουσι θεαὶ δεινοῖο χόλοιο,
πρίν γ’ ἀπὸ τῷ δώωσι κακὴν ὅπιν, ὅς τις ἁμάρτῃ.”

Ησίοδος, Θεογονία 215-222

”And the Hesperides, who guard the rich, golden apples
and the trees bearing fruit beyond the glorious Ocean. Also
she bare the Destinies and ruthless avenging Fates, [Clotho
and Lachesis and Atropos, who give men at their birth both
evil and good to have], and they pursue the transgressions
of men and of gods and these goddesses never cease from
their dread anger until they punish the sinner with a
sore penalty”

Isiodos, Theogonia 215-222

Abstract

The current master thesis was written during the academic year 2013 − 2014 at the Norwegian University of Science and Technology (NTNU). It concerns the implementation of the case-crossover design in a combination with the lasso method, for investigating potential effects of a set of drugs on myocardial infarction. The datasets that were used in the analysis were generated based on information about the usage frequencies of the drugs in the period from 2008 to 2012. The reason for using generated datasets was because the actual dataset would not be available before the deadline for delivering this thesis. Furthermore, the thesis provides a brief explanation of how the lasso method can be used in the case of generalized linear models, as well as the case-crossover design. The main analysis was based on two datasets such that probably weak aspects of the lasso could be discovered. Another aspect of the current thesis was the implementation of the relatively new inference method for the lasso, as well as the implementation of two forms of the lasso method: simple lasso and bootstrap lasso. Finally, the current thesis shows, with numerical results, that the bootstrap form of lasso is an effective variable selection method, and that the lasso inference is not yet sufficiently developed.

Sammendrag

Masteroppgaven ble skrevet ved NTNU i løpet av det akademiske året 2013 – 2014. Oppgaven omhandler implementeringen av case-crossover designet i en kombinasjon med lasso metoden for å undersøke potentialle effekter av et sett av medisiner på myocardial infarction. Det virkelige datasettet var ikke tilgjengelig før innleveringsfristen, derfor ble genererte datasett brukt i analysen. Datasettene som ble brukt i analysen var generert basert på forbruksfrekvensene til medisinene i perioden mellom 2008 – 20012. Videre gir oppgaven en forklaring på hvordan lasso metoden kan bli brukt både i generaliserte lineære modeller og i case-crossover design. Analysen var basert på to datasett sånn at de svake aspektene til lasso metoden ble funnet. I tillegg omhandler oppgaven implementeringen av den nye lasso inferensen og to forkjellige lasso-typer, enkel lasso og bootstrap lasso. Til slutt viser oppgaven, ved bruk av numeriske resultater, at bootstrap lasso er en effektiv variabel utvalgsmetode, og at lasso inferens ikke er tilstrekkelig utviklet.

Preface

The reason that I chose to work on the current subject was because I am very interested in bio-statistical science. From the evolution of the environment that we live in, to the cells that we are consisted of and to every step we take further, can be modelled as a mathematical problem. And where a problem exists, there exists a solution. I believe that mathematical applications are the actual key to most of the biological/medical problems.

I would like to express my gratitude to my supervisor Professor Bo Henry Lindqvist for his useful comments and remarks through the developing process of this master thesis. I feel very fortunate of having Bo as my supervisor both for his knowledge and his kindness. Moreover, I would like to thank Professor Imre Janszky for introducing me to the topic as well for his passion on the subject. This encouraged me to do my best. Furthermore, I would like to thank Professor Pål Sætrom and Professor Jo Eidsvik for their very helpful suggestions for some of the algorithms and for the smart handling of the datasets. I would also like to thank my family and my friends, for their support during the whole period of my master studies. Finally, I would like to give my special thanks to Ina-Mari Norum for trying to keep me calm with her delicious cakes!

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction to the Problem

1.1 Chapter Description

The current master thesis consists of six chapters and four appendices. Chapter 1 gives an introduction to the problem that the current thesis concerns, as well as a brief description of how the datasets were generated. Chapter 2 concerns the case-crossover design and presents the basic mathematical formulas for this design. In chapter 3 an introduction to the generalized linear models is given, as well as the presentation of the main algorithms that were used in the analysis. Moreover in the same chapter, the lasso method is concerned, which was the main estimation method that was used in the current analysis. Furthermore, chapter 4 gives an adaptation of the theory to the problem under research. The results of the theory are presented in chapter 5. Furthermore, chapter 6 discusses the results of the analysis, as well as the probabilities of potential improvements. Moreover, appendix A concerns difficulties and challenges that occurred during the analysis and provides a presentation of the algorithms. Finally, appendices B and C concern a proof of an equation and the tables of the results from the analysis, and appendix D contains some Quantile-Quantile plots. References of the appendices are given throughout the chapters.

1.2 The Problem

1.2.1 Myocardial Infarction

Myocardial infarction (MI) is a heart disease [1]. It occurs when a *coronary artery*[1] is blocked [18], which disables the blood supply from the artery to the *myocardial tissue* and eventually results to the death of the myocardial cells [1]. The event of myocardial infarction can either be with mild or disastrous results, which could lead to the death of the patient [27].

[1] Definitions of some terms that are in *Italic* can be found in the glossaries section.

1.2.2 Purpose of the Analysis

The current master thesis had three purposes. The first one was to investigate if any of the drugs from the *NRD* dataset (will be presented in the next subsection) could have caused MI to a set of patients, by estimating their risk ratios. For that reason, the case-crossover design was used as the main layout of the analysis, as proposed by Professor Imre Janszky. The second purpose was the investigation and the understanding of the bootstrap lasso behaviour on two different datasets. The lasso method was proposed by Professor Bo Henry Lindqvist and it was the main estimation method for the log risk ratios. Finally, the third purpose was the implementation of the newly proposed covariance test statistic. Emphasis is given on the second and third purposes. This is because we were more interested in how the lasso method works and because the datasets were not real. However, the results from the first purpose are interpreted as they should be interpreted for real datasets.

1.3 The Datasets

1.3.1 Overview

Two generated datasets were used for the analysis. The reason for generating the datasets was that the actual data couldn't be available soon enough and the deadline for delivering this thesis would have passed. Each dataset was a set of information for each patient's drug intake history in the time period from 2008 to 2012. Those information were (1) the year, the month and the day of birth for each patient, (2) the year, the month and the day of MI and (3) the year, the month and the day for each drug prescription for each patient. The difference between the two datasets is that the one dataset is randomly generated, meaning that the MI event was generated independently from the drug intakes of each patient, while the other dataset had some weights placed on 100 of the 775 drugs, making those drugs having a probability of causing MI. Those 100 drugs were randomly chosen by the algorithm and were therefore unknown. The reason for including a completely random dataset is for understanding how the lasso method behaves on two different datasets. It would be reasonable to expect that the completely random dataset will give us no significant results, although non-randomness might have been caused when generating this dataset, but it should not be of big significance. For the rest of the thesis we will call the random dataset RD and the non-random dataset NRD.

1.3.2 Generation

This subsection describes how the two datasets were generated. First of all, the only available data that could be used for generating the whole datasets was:

1. A list of 775 drug names that were used in real life.
2. A list of 775 numbers scaling from 1 to 1.000. Each number corresponded to how many people per thousand took each drug in the interval of years between $2008 - 2012$, i.e. the observation period.
3. A list of 775 numbers scaling from 1 to 52. Each number corresponded to the mean number of weeks per year that the corresponding drug was used by each patient.

The above information were given by Imre Janszky MD PhD. The rough estimation of the frequencies was based on the frequencies of the general population provided by the Norwegian Prescription Database `http://www.norpd.no`. Those information were used for creating the datasets, each of 75.000 people which would consist of the following outputs:

- ID ranging from 1 to 75.000 and sex of each patient.
- The year, month and day of birth for each patient.
- The year, month and day on which the MI event occurred for each patient.
- The year, month and day on which each patient took a specific drug.
- Drug names (which were already given as input).

As we will see in chapter 2, only the date for the MI event and the drug intakes were needed for the analysis (personal characteristics like age and sex are being removed from the model). For the MI event and the drug intakes it seemed easier to generate the events according to the days between 1 and 1827 which was the total number of days in the period of five years between 2008 and 2012, taking the leap years into account. Then according to those days, the exact dates could be calculated using the *lubridate* R-package. However, for the analysis part, only those variables were needed. Therefore the rest of the variables (sex, date of birth etc.) were generated only for illustration purposes for showing how the actual data would look like. Furthermore, each patient who took the same drug several times, had to have taken the drug with at least seven days[2] difference. This was actually very time consuming to achieve in R, if not impossible, because there is no straightforward way to generate numbers within a given distance. Therefore, without loss of generality, the generation of the days was made in a sequence from 1 to 1827 by 7 (every seventh day). That would not cause any problems in the

[2]Every seventh day for reasons based on the case-crossover design which will be discussed in subsection 4.2.1.

actual analysis anyway. Of course, a patient might have taken two or more different drugs at the same time.

The difficult parts were that of simulating the age of each patient when the MI event occurred (only for illustration purposes), and the total number of weeks that each patient took a specific drug, given the mean number of weeks for each drug. Those simulations, although, should be done under some assumptions:

- The age of each patient when he/she experienced the MI event, was let to vary between 30 and 105 with increasing rate from 30 to 65 $\sim$ 70 and then it decreases down to 105. This information was again given by Imre Janszky MD PhD.
- For both variables of ages and weeks a Chi-squared distribution was used for generating identically independent values and then round them down to the nearest integer. Ages were assumed to be Chi-squared distributed with mean 65. Furthermore, the values that were generated were in the interval $[30, 105]$. This was done by simply rejecting those who weren't. Weeks were assumed to be Chi-squared distributed with mean given from the total weeks input set. Again, their generated values were in the interval $[1, 52]$.

The choice of Chi-squared distribution was made after working with the Normal distribution without getting the desired results. The problem with the Normal distribution was that it had to be skewed since, especially for weeks, the mean could be much smaller or larger than the median. However, it was impossible to generate random skewed Normally distributed numbers without further information but the mean. Chi-squared distribution worked perfectly, on the other hand, as we can see from the simulated examples in figure 1.1.

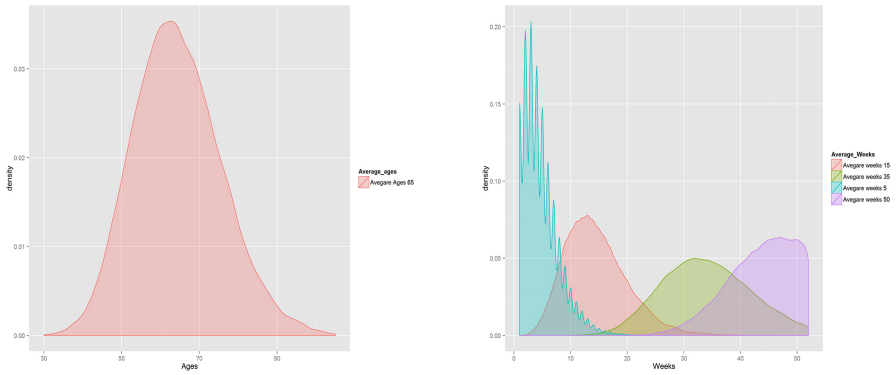

Figure 1.1: Plot of 50.000 simulations from the Chi-squared distribution for Ages (left) and Weeks (right). For the right figure, the variables are "Average weeks 15", "Average weeks 35", "Average weeks 5" and "Average weeks 50", from top to bottom of the right panel. Note that, this is just an example of a simulation and therefore, the blue density has edges. However this is not of any concern.

The two datasets were generated by almost the same way. The only difference between them was that the one dataset was completely random, meaning that the drug intakes and the MI event were generated independently. That is, the MI event could have occurred on any of the 1827 days with the same probability 1/1827. While the other one was not so random, meaning that the drug days were first generated, then 100 drugs were randomly chosen, for whom weights were assigned such that not all days would have equal probability of 1/1827 for getting MI. The reason for doing that will be discussed afterwards.

In table 1.1, one can see an example of how a part of the datasets looks like for one patient. The table consists of the patient's ID, the ID of each drug that this patient took, the day of the drug intake (ranging from 1 to 1827), and the MI event day (ranging again from 1 to 1827). The rest of the variables, as well as the patient's ID, are actually irrelevant for the analysis. Finally we can see from the table, that the patient took the same drug more than two times, namely "C10AA01" (red), on days 281, 323 and 29. Clearly those days have more than 7 days distance in-between them. On the other hand, drugs "B01AC06" and "R05CB01" (blue) were taken on the same day, this is considered as a potential interaction. Note that the two drugs were taken on exactly the same day because the generation was every seventh day, as said before. In a real dataset, drugs with intake date distance less than seven days should be considered as potential interactions. The two generated datasets were further modified for the analysis purposes. We shall come back to this after the presentation of the theory in the following chapters.

Table 1.1: An Excerpt From a Patient's Data.

Patient ID	Drug ID	Drug Day	MI Day	Day of Birth	Moth of Birth	Year of Birth	Sex
16543	N02BE01	617	638	22	5	1974	F
16543	R05CB01	442	638	22	5	1974	F
16543	C09AA05	288	638	22	5	1974	F
16543	R05CB01	407	638	22	5	1974	F
16543	C10AA01	281	638	22	5	1974	F
16543	N05BA04	260	638	22	5	1974	F
16543	B01AC06	575	638	22	5	1974	F
16543	R05CB01	575	638	22	5	1974	F
16543	B01AC06	638	638	22	5	1974	F
16543	C09AA05	533	638	22	5	1974	F
16543	C10AA01	323	638	22	5	1974	F
16543	C09AA05	29	638	22	5	1974	F
16543	A10AD05	428	638	22	5	1974	F
16543	B01AC06	106	638	22	5	1974	F
16543	D08AC02	505	638	22	5	1974	F
16543	R05CB01	106	638	22	5	1974	F
16543	N02BE01	624	638	22	5	1974	F
16543	N05BA04	477	638	22	5	1974	F
16543	N06AX11	372	638	22	5	1974	F
16543	C10AA01	29	638	22	5	1974	F

Finally, algorithm 1 gives a pseudo-code for how the datasets were generated. The initialization step creates an empty data matrix of 75.000 rows and 775 columns, for the total drugs. On that step we can also generate the sex and age of each patient, or any other internal characteristic (like region etc.), and insert it as a column of the matrix. However those internal characteristics will be removed afterwards and will therefore not be treated as factors at all. On step

one, we first use the people frequencies for assigning drugs to each patient (namely information number 2 from the total 3 informations named previously). This is done by simply multiplying the frequencies by 75, since they correspond to 1000 people for those 5 years, and then round them to the nearest integer. Then for each frequency, which corresponds to a drug, we randomly choose IDs, for assigning them the drug. Note that the total randomly chosen patient IDs for each drug, will be equal to the corresponding drug frequency. By that time we know exactly by how many patients a drug was taken, and we also know which patient took a specific drug. Then for each patient and each drug that the patient took, we use the chi-square distribution for the week frequencies of the drugs, and we generate the weeks/times that a drug was taken by that patient. Thus, by that time, we should have a complete image of how many times a drug was taken by each patient. Then for each patient and for each time that the patient took a drug, we generate the date of the drug intake by simply generating numbers in the interval $(1, 1827)$, by 7. The initialization step and the step one are common to both datasets.

Step two differs between the datasets. This step generates the date of the MI event for each patient. For the RD dataset we generate the dates using the "sample$(1 : 1827, 1, prob)$" command in R, which randomly chooses integers in the interval $(1, 1827)$. There $prob$ is a vector of length 1827 which contains the element $1/1827$. That is, all the days are equally chosen and independent from the drug intakes. Note that each element of the vector corresponds to each day from 1 to 1827. For the NRD dataset, we first randomly choose 100 drugs and we assign to them probabilities in the interval $(2/1827, 1)$, that is, greater than $1/1827$. Then for each patient, if he or she took one or more of those drugs, we find the date of those drug intakes. Then the corresponding positions on the $prob$ vector, are assigned the probabilities chosen before (those greater than $1/1827$ for the corresponding drug). Then the MI event is generated according to that new vector. Note that, the sum of the elements of the vector does not need to be equal to 1 in $R - studio$;

Data: The initial drug frequencies.
Result: Generated dataset.
Initialization: Create the 75.000 matrix rows with the IDs.
Step 1: Generate the total drug uses for each patient.
Step 2: Generate the MI event for each patient according to the command: $sample(1 : 1827, 1, prob)$

Algorithm 1: Data Generation Algorithm

Chapter 2

Case-Crossover Studies

A case-control study is a study for which the case and the control subjects are different subjects from the same population. For example, two different people, one deceased (case) and one not (control) [6]. Although case-control studies serve well for many kinds of analysis, they do not provide a way of studying the time it takes for an exposure, trigger, to have an effect. Because case-control studies gather exposures over time. Moreover, another problem with the case-control studies is that the subjects who serve as cases are more likely to remember "what they did" before the event, than the control subjects[1]. This can lead to misclassified studies. Furthermore, internal characteristics of the subjects (like sex, age, geographic region etc.) can not be eliminated. Finally, one of the most difficult aspects of case-control studies is that the control group should be representative of the population where the case group comes from. And this is very difficult to achieve [25].

2.1 The Case-crossover Design

The case-crossover design was purposed by Maclure [18], and is able to solve the problems that under a case-control study would remain unsolved. They are most commonly used in studies where the effect of exposures on acute health events need to be measured [14]. The key aspect with the case-crossover studies is that the case and the control is the same subject but on different time periods. In case-control studies we have a group of subjects which serve as cases at a given time and a group of other subjects, different than the first ones, which serve as controls on the same time. On the other hand, in case-crossover studies we have a group of subjects which serve as cases at a given time, and the same group of subjects which serve as controls for themselves at a different time [19], thus characteristics of the subject like sex and age, are eliminated [14]. By doing that, it becomes easier to research a cause of an event, the frequency of that cause and how common that cause was

[1]Note that this problem does not apply to drug registries, because each patient's information is probably stored.

among the subjects. Moreover, case-crossover studies work perfectly on exposures that are sporadic over time [19], and they assume that the event happened a short time after an exposure [16]. Finally, the name of the case-crossover studies comes from the fact that the analysis crosses over times of exposure and non-exposure for each subject [18]. That is, in contradiction with the case-control studies, where the subject has to be always at risk, in case-crossover studies the subject's history includes both risky and non-risky periods [16]. Figure 2.1 shows the difference between a case-crossover and a case-control study.

Although case-crossover studies provide big advantages to the researcher wishing to study a rare event, their advisability can be threatened by some factors. One of those factors is the uncertainty about the effect period of a potential trigger. In such studies, the researcher has no control over the sequence of the exposures nor the exact occurrence of the event under study. Those characteristics might completely be controlled by the subject itself and thus, they might lead to within-individual confounding. Finally, because the subjects are usually interviewed before the research, such that the exposures' sequences are recorded, false or forgotten information might lead to selection or information bias[2] [18].

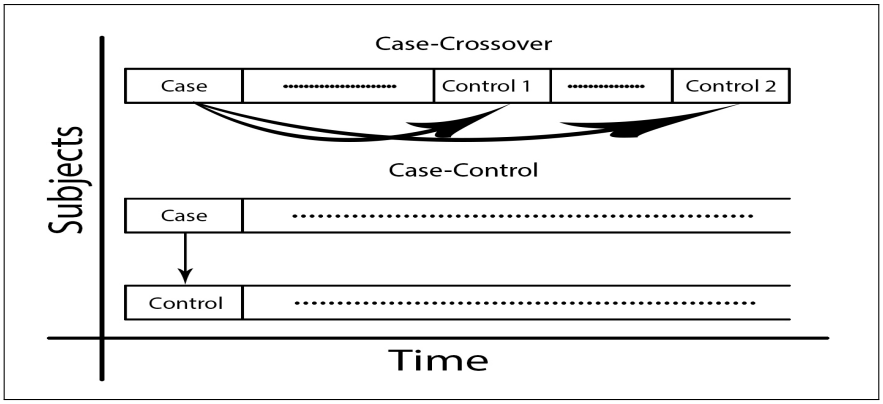

Figure 2.1: This figure is a simplified version of the figure given by Maclure and Mittleman [19], and it shows the difference between a Case-crossover and a Case-control design. For the former, the subject is the case and the control of itself on different time periods. For the latter, the case is the subject itself, but the control is another subject on the same time period. Here, the number of case and control boxes is only an example.

2.1.1 Windows and Triggers

A case-crossover study is mainly based on windows and triggers. The triggers are the potential causes of the event, while the windows are time frames that construct the case-crossover design. There exist various designs both for the kind of windows and the effect periods of the triggers [13, 16]. In this section we present the main background theory for constructing such windows, as well as the effect that the triggers have on them.

[2]This again does not apply to cases where the information is registered.

2.1.1.1 Definition and Characteristics of Windows

A specific time frame is called a window. It is the time unit under observation, where we expect to see a trigger [19]. Windows are of two kinds: control windows and case windows. A case window is a time frame in which we assume that a trigger caused the event. The case window can be a time frame that includes the event time, or it can be a time frame prior to the event time. The relationship of the case window and the event time depends on the induction time and effect period of the trigger, as we will see in subsection 2.1.1.3. A control window is also a time frame, where we expect to see a trigger but not the event [14], and it is placed before the case window [25]. We can have single or multiple control windows in the study [25], but only as many case windows as the total events occurred for each subject [12, 16]. At least one control window should be provided for each subject, such that confounding from constant characteristics are minimized as much as possible [19].

Case and control windows can be either exposed or unexposed. If a window is exposed, it means that on that specific time frame a trigger occurred. In a case-crossover study, pairs of case and control windows that are both unexposed or both exposed, do not contribute anything in the analysis and they should not be taken into account [25]. Therefore in a case-crossover study, each subject should have at least one exposure event [19]. According to Schneeweiss et al. [25], Hernández-Díaz et al. [12] and Maclure and Mittleman [19], if we have a study which consists of more than one control window, then subjects that have all their control windows and all their case windows exposed (or all the control and case windows unexposed) at the same time, should not be included in the study.

The control and case windows should not overlap with each other. Furthermore, a gap can exist between a control and a case window, or between a case window and the actual event. Those gaps are specified by the effect period [12, 18], as well as the washout period [18] of the trigger. We shall revisit this in subsection 2.1.1.3. Finally, defining the length of the time frame for a window is something that needs to be done carefully and depends on information about the time it takes for a trigger to put the subject in the effect period [19]. Note however, that the smaller the length of the windows is, the smaller becomes the risk for seasonal bias [16].

Each subject in the study has its own "referent" window, which is the set of control and case windows that correspond to that subject. The time on which the referent window is placed differs among the subjects, because the corresponding set of case and control windows depends on the event time of each subject [14].

2.1.1.2 The Main Categories of Referent Windows

Considering an observation period from $\{1, ..., T\}$, there exist two main categories for the designs of the referent windows. The one category corresponds to *localizable* designs. Those designs, divide the time period into disjoint strata[3] whose placement on the timeline does not depend on the event itself. That is, the strata are localized a priori. Then according to which stratum the event occurs, those designs choose

[3]Strata are defined as relatively big time periods, such as months or years [13, 16].

all or some of the time points[4] in that specific stratum as control windows. For example, if we define strata as months and the event for a subject is on a Friday, then this Friday is the case and some or all the previous Fridays on that month are the controls. The previous example is an example of a *time stratified* design whose strata are months [13]. Another example of localizable design is the *Navidi's* design [16], or as called by Janes et al. [13] *bidirectional* design, which is a time stratified design with only one stratum, corresponding to the whole time period [13].

The other category corresponds to *non-localizable* designs [13], or according to Lumley and Levy [16], *Referent window designs*. Those designs choose a fixed number of time points before and/or after the event, as control times [16]. The reason for doing that is mainly for reducing potential seasonal bias. Moreover for those designs, the time of the event completely determines the placement of the windows (both case's and control's). Thus the windows are not localized a priori, but they are random variables which depend on the event itself [13]. There exist various designs which belong to that category, such as the *unidirectional* design, which first finds the event time and treats it as a case window and the whole previous time (from the beginning up until the event time) as one control window, the *unidirectional*-"numbers" designs[5] which treat the event time as case window and some fixed time points, previous to the time event, as control windows. Finally, there also exist *symmetric bidirectional* designs, which have as control windows, fixed time points both before and after the event time [13].

Choosing the appropriate design depends mainly on the kind of the event, as well as the available data. For events that can occur only once (such as death), one cannot choose designs which will sample control windows after the event. Moreover, some events may cause changes to some model factors when they occur (such as changes on drug intakes after the event), therefore control windows after the event are again not suitable. Furthermore, another reason that there exist many different designs is that some of the designs cause bias to the estimates. This is the main difference in the estimates obtained by a localizable design and a non-localizable design. According to Lumley and Levy [16] and Janes et al. [13], localizable designs result to unbiased estimators, while non-localizable designs result to bias estimators. However, the bias for the non-localizable designs is not big.

In general, there does not exist an ideal design. The two main categories are the localizable designs and the non-localizable designs, each one having its own subcategories according to the placement of the control windows. The difference between the two designs is that the localizable designs compare control windows within a stratum, which is placed a priori and independently of the event time, and this results to unbiased estimates. On the other hand, the non-localizable designs compare control windows that do not belong to a pre-specified stratum, and are placed according to the event time. That is, the windows are random variables that are placed according to the event time and thus, the event time is completely determined by them [13]. The reason for at those designs result to biased estimators

[4]By time points we mean time measurements, like days, hours etc., that the study is based on [16].

[5]Those designs are of the form: unidirectional 7, 14, 21, or unidirectional 3, 4, 5, or any numbers we want. The numbers indicate which windows are chosen as controls [13].

will be discussed later.

2.1.1.3 Triggers

By triggers we mean a potential cause of the event under study. The subjects are exposed to triggers and we wish to investigate their immediate[6] effect on the event, by studying their exposure frequencies. That is, we want to study if that trigger had anything to do with the caused event. A trigger for example, can be a drug intake, the event can be a disease and we want to find out if the drug could have caused the disease. Moreover, some factors affect the subject in the opposite way than the triggers do, by preventing the event from happening. Then the interruption of those factors can be treated as a trigger. Defining the triggers depends on what we think that a cause of an event could be. The induction time, the effect (or hazard), and washout period are units of observation that divide the exposure (trigger) in different time intervals, each giving a different point of view of the exposure's effect [19].

The induction time becomes more easily understood by dividing it in minimum and maximum induction times. Minimum induction time is the time interval between the time when the trigger occurred and the time when the trigger begins to affect the subject [18]. For example, a drug intake might take some hours before it actually begins to affect the person. Maximum induction time is the time interval after the effect period and is analogous to the washout period [19]. If an event occurred within the minimum or the maximum induction times of a trigger, then that event cannot be associated with that trigger, because the minimum and maximum times are not hazardous periods [18].

The effect period is the time period in the interval between the minimum and maximum induction times [19]. This is the hazard period of the subject and it can be divided into periods of higher and lower risk. The effect period of a trigger is the period that actually affects the subject. If an event occurred during the effect period of a trigger, then that trigger could be the cause of that event [18].

The minimum and maximum induction times specify the times where the windows should be placed, while the effect period decides the length of the windows, both case's and control's [12]. If the trigger has a minimum induction time different than zero, meaning that the trigger starts to affect the subject after that minimum induction time, then the case window should be placed at a time point before the event such that the time distance between the time on which the trigger occurred and the time of the event, has minimum value equal to the induction time and maximum value equal to the minimum induction time plus the effect period. The length of the interval between the minimum and maximum times is specified by the effect period. Accordingly we place the control windows [12, 14, 19]. The time interval between two windows is specified by the maximum induction time of the previous window, because this corresponds to the washout period and we

[6]Note that, a trigger has to have an immediate effect on the event. For example, smoking for many years cannot be considered as a trigger under a case-crossover design. Case-crossover designs consider only triggers with an immediate effect, such as drug intake before a health event, or a telephone call before a car accident [19].

don't want the windows to overlap. Finally, the times where the windows are being placed, differ among the subjects. Because they depend on the time of the event, which is probably different for each subject. On the other hand, the length of those windows is the same for each subject, since it depends on the effect time of a specific trigger, which is common to all subjects [14].

The theory discussed in subsection 2.1.1 constructs the design of the referent window for each subject. In the next section we shall develop the conditional likelihood for the case-crossover design. This likelihood model will, among others, depend on the construction of the referent window.

2.2 Designing the Study

In this section the risk ratio and the conditional likelihood for a case-crossover study are presented. The models developed here are mainly based on the articles by Lumley and Levy [16] and Janes et al. [13], although, some modifications on the notations had to be done for giving better explanations.

2.2.1 The Risk Ratio

Assume that we have a total of N individuals who have experienced the event somewhere in the time interval $(1, ..., T)$, where T is the observation period. Let $t \in \{1, ..., T\}$ be the discrete time points where each case was checked (followed up). The hazard function for the i-th subject at time t is defined by [16]:

$$\lambda_i(t; X_{it}) = \lambda_{0it} e^{X_{it}\beta}$$

If we let Y_{it} be the risk occurrence indicator for subject i at time t, that is $Y_{it} = 1$ if "event" at time t and $Y_{it} = 0$ if not. Then according to Janes et al. [13], we can use that hazard function as the likelihood at time t for the i-th subject, if we assume that the events are rare. More specifically, we can assume that:

$$P\left(Y_{it} = 1 | X_{it}\right) = \lambda_{0it} e^{X_{it}\beta} \tag{2.1}$$

there $X_{it} = (X_{it1}, ..., X_{itp})$ is the p dimensional exposure time series vector, and p is the total number of log risk ratio coefficients $\beta = (\beta_1, ...\beta_p)'$. The baseline function λ_{0it} is assumed to vary only among subjects, carrying characteristics associated to that subject [13].
This is the probability that the i-th subject will experience the event exactly at time t, given its exposure series. The conditional likelihood model developed in the next section is based on equation (2.1).

2.2.2 The Conditional Logistic Regression Likelihood

In this subsection the conditional logistic regression likelihood is obtained as proposed by Janes et al. [13] and Lumley and Levy [16]. We shall furthermore restrict

ourselves to one event in the time interval $(1, ..., T)$ and thus, we will have one case window.

To start with, let $t_i \in \{1, ...T\}$ be the event time for the i-th subject and W_i be the corresponding referent window for the i-th subject, either localizable or not, as defined in subsection 2.1.1.2. Define furthermore $W = (\text{length of } W_i)$[7], which should be common to all subjects, since they all have the same length of referent window. As said in the previous section, for the i-th subject at time t we have the exposures vector $X_{it} = (X_{it1}, ..., X_{itp})$. Since t runs in the referent window, we define for the i-th subject the exposure matrix for its referent window as follows:

$$X_i = \begin{bmatrix} X_{i1} \\ X_{i2} \\ \vdots \\ X_{it} \\ \vdots \end{bmatrix}$$

where each X_{it} is of dimensions $1 \times p$ and X_i is of dimensions $W \times p$. Finally, let the total exposures matrix that consists of all subjects be:

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_N \end{bmatrix}$$

which is of dimensions $(N \cdot W) \times p$. Therefore, an arbitrary row from that matrix corresponds to a subject at a given time t. According to Lumley and Levy [16] and Janes et al. [13], the score function of the conditional logistic regression likelihood for the i-th subject is:

$$U_i(\beta) = X_{it_i} - \sum_{t \in W_i} X_{it} \frac{e^{X_{it}\beta}}{\sum\limits_{s \in W_i} e^{X_{is}\beta}}$$

That is, the score function for each subject is the exposure of the subject on the event time, minus a weighted average of exposures among all windows. Those equations, however, do not have necessarily zero mean. For those equations to have a zero mean, need the windows W_i to be created with respect to a localizable design. Because under that design, the event is not determined by the windows and thus the following likelihood function:

[7]By length of the referent window we mean the total number of windows that it consists of. One should think of it as the total number of rows in a data matrix that correspond to the same subject but at different time points.

$$L_{loc}(\beta) = \prod_{i=1}^{N} P\left(T_i = t_i | X, W_i, \sum_{s=1}^{T} Y_{is} = 1\right) =$$
$$= \prod_{i=1}^{N} \frac{P\left(T_i = t_i, \sum_{s=1}^{T} Y_{is} = 1 | X, W_i\right)}{\sum_{t=1}^{T} P\left(T_i = t, \sum_{s=1}^{T} Y_{is} = 1 | X, W_i\right)} =$$
$$= \prod_{i=1}^{N} \frac{\lambda_{0it_i} e^{X_{it_i}\beta}}{\sum_{t \in W_i} \lambda_{0it} e^{X_{it}\beta}} = \prod_{i=1}^{N} \frac{e^{X_{it_i}\beta}}{\sum_{t \in W_i} e^{X_{it}\beta}}$$

has meaning. There we condition on the total exposures matrix, the referent windows and on the fact that only one event could have been occurred in the interval $(1, ..., T)$. According to Lumley and Levy [16] and Janes et al. [13], the baseline function λ_{0it} can be considered constant within small intervals and therefore, it is being cancelled out. Replacing the sum of the denominator of the third equality (sum of the whole period) to that of the forth (sum over the control and case windows), has only meaning under a localizable design. Under a non-localizable design, the previous likelihood will result to the value of one and thus, it will be non-informative. The reason for that, is that conditioning on the referent windows, we know where the event occurred, since the windows are placed according to that event. If for example we use an unidirectional design, we know that the event will be on the last window. If we use a symmetric bidirectional design, the event will be in the middle window. According to Lumley and Levy [16] and Janes et al. [13], the random variables under a non-localizable design are the referent windows and thus, the appropriate likelihood for that design would be:

$$L_{nonloc}(\beta) = \prod_{i=1}^{N} P\left(W_i = w_i | X, \sum_{s=1}^{T} Y_{is} = 1\right) =$$
$$= \prod_{i=1}^{N} \frac{P\left(W_i = w_i, \sum_{s=1}^{T} Y_{is} = 1 | X\right)}{\sum_{t=1}^{N} P\left(W_i = w, \sum_{s=1}^{T} Y_{is} = 1 | X\right)} =$$
$$= \prod_{i=1}^{N} \frac{\lambda_{0it_i} e^{X_{it_i}\beta}}{\sum_{t=1}^{T} \lambda_{0it} e^{X_{it}\beta}} = \prod_{i=1}^{N} \frac{e^{X_{it_i}\beta}}{\sum_{t=1}^{T} e^{X_{it}\beta}} \tag{2.2}$$

with corresponding scoring functions:

$$U_i(\beta) = X_{it_i} - \sum_{t=1}^{T} X_{it} \frac{e^{X_{it}\beta}}{\sum_{s=1}^{T} e^{X_{is}\beta}}$$

That is, the sum is taken among the whole period and not only among the windows. This however, might not be desirable for a study. The reason for that, is that not all kinds of events can give desirable data after their occurrence. As said before, events such as death or events that alter crucial factors of the study after their occurrence, can not be measured on the whole time period. According to Lumley and Levy [16] however, one can actually change the sum of the likelihood in equation (2.2), to a sum over the control and case windows, such that:

$$L_{nonloc}(\beta) = \prod_{i=1}^{N} \frac{e^{X_{it_i}\beta}}{\sum\limits_{t \in W_i} e^{X_{it}\beta}} \tag{2.3}$$

This however, is not an actual likelihood and thus, it will result to biased estimates. This bias is called overlap bias, as said before, and according to Lumley and Levy [16], it will be small. We however, will use the likelihood in equation (2.3) for our analysis.

Chapter 3

Iterative Methods and The Lasso Method

This chapter concerns the mathematical models that are going to be used in the later analysis. The models here are first presented in general and in the next chapter we will adapt the models in our case-crossover design. It is important to understand that a case-crossover design can be analysed in many different ways, depending on many factors like for example the total number of windows or how we wish to estimate a tuning parameter, as we will see in this chapter.

This chapter begins with a short presentation of theory for generalized linear models, which is given in section 3.1. Then it continues in section 3.2 with an introduction to iterative methods which were used for constructing the algorithms of this thesis and were the main methods for estimating the parameters of the models. Finally, the main focus of this chapter is placed in section 3.3 where the lasso method is presented. Although there are many versions of the lasso method, we will focus on those needed in the analysis of this thesis.

3.1 Generalized Linear Models

The term "linear models" comes from the fact that the response and the covariates have a linear relationship. However, especially for binary or count responses, this is not always applicable. Moreover the normality assumptions of a model are not always easy to achieve. Under those cases, the covariates have to be transformed through a linear function in such a way that the support of the response is feasible. Those models are called generalized linear models (GLM), because they generalize-extend the linear regression models through that linear function [20].

3.1.1 The Link Function

The linear function that connects the response and the covariates, under a generalized linear model, is called the *link function* and it plays the most important role

on such models [20]. Assume that we have a response of the form:

$$y|X, \beta \sim f(y|X\beta)$$

where the f distribution is a linear combination of X, given that the response and the explanatory variables have a linear dependence. Furthermore, let $y = (y_1, y_2, ..., y_n)$ be the response, $\beta = (\beta_1, ..., \beta_p)'$ be the coefficients and

$$X = [x_1, x_2, ..., x_p] = \begin{bmatrix} x_{11} & x_{12} & ... & x_{1p} \\ x_{21} & x_{22} & ... & x_{2p} \\ \vdots & \vdots & ... & \vdots \\ x_{n1} & x_{n2} & ... & x_{np} \end{bmatrix}$$

be the $n \times p$ matrix of the explanatory variables. For a GLM to be valid needs the conditional density of y, given X, to belong to the exponential family and it also needs to be parametrized by $\mu = \mu(X) = E[y|X]$, the expectation parameter. Furthermore, essential is the existence of the link function g because it connects the mean response μ with the covariate vector such that, $g(\mu) = X\beta = \eta$, where η is called the *linear predictor* of the model [20, 24]. The link function has to be one-to-one function such that its inverse exist. If those criteria are satisfied, the generalized linear model is valid and we can write:

$$\mu = E[y|X] = g^{-1}(X\beta) = g^{-1}(\eta)$$

For a distribution to belong to the exponential family, it needs to be shown that the density function can be transformed to the form:

$$f(y_i) = e^{\frac{y_i\theta_i - b(\theta_i)}{\alpha_i(\phi)} + c(y_i,\phi)} \Leftrightarrow log(f(y_i)) = \frac{y_i\theta_i - b(\theta_i)}{\alpha_i(\phi)} + c(y_i, \phi)$$

there $\alpha_i(\phi) = \frac{\phi}{\rho_i}$, where p_i is a weight usually set equal to one and ϕ is a dispersion parameter [24]. Furthermore, for a model that belongs to the exponential family we have the following facts:

- $E(Y_i) = \mu_i = b'(\theta_i)$
- $Var(Y_i) = \sigma_i^2 = b''(\theta_i)\alpha_i(\phi)$

Those equations are very important for some of the iterative methods when applied to the lasso method.

3.2 Iterative Methods

This section focuses on some iterative methods that are widely used for estimating unknown parameters of a given function $f(\beta)$. For every method, a summary of its usage is given as well as the algorithm itself.

3.2.1 Coordinate Algorithms

Iterative methods like Newton-Rapshon [24], who estimate the coefficients β of a function $f(\beta)$ by using scoring techniques, are not always easy to apply because the inverse of the Hessian or the observed information matrix is needed and for a model with many parameters this is extremely time costly [32]. Coordinate algorithms on the other hand, update one parameter at a time and thus, no Hessian is needed, but only the gradient vector. This makes those algorithms very fast and they are commonly used in models of high dimensions. As for the other scoring algorithms, so for the coordinate algorithms, their use is essential when we need to estimate the coefficients of a function [32].

Coordinate algorithms are very useful when we need to estimate the parameters of a function which is non-differentiable but it has directional derivatives along its forward and backward direction. Consider a non-differentiable objective function $f(\beta)$, where $\beta = (\beta_1, ...\beta_p)'$ is the vector of parameters, and let u_j and $-u_j$ be the forward and backward coordinates, respectively, along which the β_j parameter varies. Then according to Wu et al. [32], if the directional derivatives exist, then they are given by:

Forward Derivative

$$d_{u_j} f(\beta) = \lim_{t \to 0} \frac{f(\beta + tu_j) - f(\beta)}{t}$$

Backward Derivative

$$d_{-u_j} f(\beta) = \lim_{t \to 0} \frac{f(\beta - tu_j) - f(\beta)}{t}$$

Coordinate algorithms use one directional scoring for updating one of the parameters at a time. They compute the directional derivatives and choose the coordinate which maximizes (if $f(\beta)$ is concave [32]), or minimizes (if $f(\beta)$ is convex [11]) the $f(\beta)$ function [11, 32]. As we will see later, the directional derivatives can be found by soft thresholding [10, 11].

There are two basic models of coordinate algorithms according to the way they update the parameters. The one is called cyclic coordinate and the other is called greedy coordinate. Cyclic coordinate updates one parameter at each loop, while greedy coordinate updates the parameter which gives the biggest desired change in the function. We shall consider only cyclic coordinate algorithms here because they are faster in the logistic regression problems [32].

Coordinate algorithms can be further divided into two kinds according to the path they follow until convergence. If the goal of the algorithm is to compute the

parameters of a convex function that we want to minimize, then the coordinate algorithm is called descent, because it goes down the path of the function [11]. If the algorithm computes the parameters of a concave function that we need to maximize, then the algorithm is called ascent because it goes up the path of the function [32].

Given an objective function $f(\beta)$, the general idea of the cyclic coordinate algorithm is to update a parameter through its partial derivative, when the rest of the parameters are kept fixed on their last update. That is, if we have p parameters in the model, we start with some initial values and then for each parameter from $1, .., p$ we compute the first order derivative, then we solve with respect to that parameter and then we update. This is done in circles until convergence has been reached. Note that each time we update a parameter, we use the updated value of the parameter when we update the next one [4]. In algorithm 2 we see an overview of the cyclic coordinate algorithm.

Data: Forward and backward partial derivatives, initial values for β.
Result: Parameter estimation via Cyclic Coordinate Descent/Ascent.
Initialization;
while $\|\beta_{new} - \beta_{old}\| > 1e-06$ *(tol.))* **do**
  for $j \leftarrow 1$ **to** *length(β)* **do**
    Update β_j by the coordinate which gives the greatest reduction/growth of the objective function $f(\beta)$;
    Update the vector of parameters β **(Important)**;
  end
end

Algorithm 2: Cyclic Coordinate Descent/Ascent Algorithm

3.2.2 Bootstrap

Bootstrap is a technique which uses the observed data for making inferences for the parameters of a model. Bootstrap is actually easy to implement and very useful for big scale data. Having a data matrix which consists of N rows, a bootstrap sample is constructed by creating another matrix of the same size as the initial data matrix. This is done by sampling rows from the initial matrix with replacement. The bootstrap then, simply takes B bootstrap samples from the dataset and it computes the desired statistic s for each bootstrap sample. Those statistics are called bootstrap replicates. Finally, the mean of the bootstrap replicates is an estimate for the statistic s, and the standard deviation of the bootstrap replicates is an estimate for the standard error of the statistic s. More specifically, for a statistic s, its standard error can be estimated by [7]:

$$\hat{SE}_{bootstrap} = \hat{SD} = \left(\frac{\sum_{b=1}^{B} \left(s(x^{*b}) - s(\cdot) \right)^2}{B-1} \right)^{1/2} \tag{3.1}$$

there $s(\cdot) = \sum_{b=1}^{B} s(x^{*b})/B$ is the estimate for the statistic, and x^{*b} is the b-th bootstrap sample. Furthermore, 95% confidence intervals (for $\alpha = 0.05$) can be computed either by sorting the B summary statistic and then take all observation that lay between the $B(\alpha/2)$ and the $B(1-\alpha/2)$ observations, or one can rely on the central limit theorem where for large sample N we expect that $\bar{x} \sim N(s(\cdot), \hat{SE}^2_{bootstrap})$ and therefore the 95% confidence interval becomes $(s(\cdot) - 2\hat{SE}_{bootstrap}, s(\cdot) + 2\hat{SE}_{bootstrap})$. Algorithm 3 illustrates the bootstrap algorithm [7].

Data: Dataset, number of bootstrap samples B.
Result: Estimated statistics and its standard error.
for $b \leftarrow 1$ **to** B **do**
  Take a sample of the same size as the data, with replacement, and compute the desired statistic ;
end
Compute the standard error;

Algorithm 3: Bootstrap Algorithm

3.2.3 Cross Validation

Cross validation is a method used for estimating the error of a fitted model. The error of a fitted model is defined by $E(y-\hat{y})^2$ where y is the observed response and $\hat{y}$ is the fitted response. This error is called prediction error (PE) and it expresses the ability of a fitted model to predict the observed response. Of course, low values of the prediction error are preferred, since the lower the error, the better the fit [7].

According to Efron and Tibshirani [7], the ideal way to compute the prediction error of a model would be to have an extra data that is not used for estimating the parameters but is used for computing the fitted response. By doing that, we can tell how well the fitted model can respond to a new data, like for example future observations. The problem with this approach is that usually, new data is not available. If this is the case, cross validation is an easy and stable approach which is commonly used for estimating the prediction error. Cross validation breaks the data into K roughly equal parts, sub datasets, and uses the i-th part as the "testing" dataset on the other $K-1$ parts which were used for fitting the model, the "training" dataset. In other words, we estimate the model with the training set and we fit it with the testing set, then we compute the prediction error and the process is repeated K times. Finally a combination of the estimated prediction errors will give the final result, usually we take the mean of all prediction errors.

The number of K sub datasets depends on the size of the initial dataset. For large datasets a number of $K = 2$ datasets is satisfactory. However, for smaller datasets K-fold cross validation is needed and some times "leave-one-out" cross validation is the best method to use, where we choose $K = n$. For a K-fold cross validation the estimator of the prediction error is:

$$CV = \frac{1}{K}\sum_{i=1}^{k}(y_i - y_i^{-k(i)})^2$$

where $y_i^{-k(i)}$ is the fitted response of the model when the $k(i)$ part was removed [7]. In algorithm 4, a pseudo-code of the cross validation procedure is given.

Data: Dataset, regression model, the number K of the folds.
Result: Cross Validation Prediction Error.
Initialization: Break the data into K roughly equal disjointed parts;
for $i \leftarrow 1$ **to** K **do**
    Create the training set from the rest of the $K-1$ parts and compute the coefficients of the model;
    Create the testing set from the i-th part and use it to fit the model whose coefficients were computed by the training set;
    Calculate the Prediction error;
end
Take the average of the Prediction errors.;

Algorithm 4: Cross Validation Algorithm

3.3 The Lasso Method

This section concerns the lasso method, which is the most important part of this thesis. A brief presentation of the lasso method is given, as well as its advantages and disadvantages. The iterative methods that were presented in the previous sections are useful methods that are being used for estimating the coefficients of a model and assessing its significance. Those methods are also applicable under a lasso design. Furthermore, in subsection 3.4.1 is given a brief presentation of the most important algorithm of this thesis. Finally, the lasso inference is being concerned in subsection 3.4.2.

3.3.1 Introduction

The least absolute shrinkage and selection operator method (lasso) was first proposed by Tibshirani [28] on his article "*Regression Shrinkage and Selection via Lasso*". It is a shrinkage method which was initially proposed for estimating the coefficients of linear models [28], although it can also be applied to generalized linear models [4, 11, 28].

The lasso minimizes the residual sum of squares given that the sum of the absolute values of the coefficients is less than a constant $\lambda \geq 0$. One of the most important abilities of the lasso method is that it tends to set some coefficients exactly equal to zero, thus it is an estimation and a model selection method at the same time [28]. This can become more clear by looking at figure 3.1 which is an iconic representation of how the lasso method estimates the coefficients in the 2D case. The rhombus in the figure is the lasso penalty and the elliptical lines are changes that the lasso applies to some initial estimates $\hat{\beta}$ throughout the estimation process. The first place that one of the elliptical lines touches the rhombus is the lasso estimate of $\hat{\beta}$. According to the figure, it is possible that the contact can be achieved at one of the corners of the rhombus, which would result to an estimated value of zero for the corresponding coefficient [28].

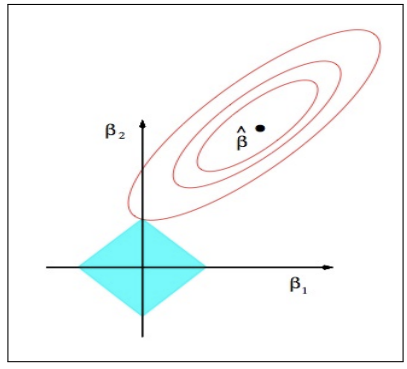

Figure 3.1: This figure is taken from Tibshirani [28] and shows how the lasso method estimates the coefficients, in a 2D case. The lasso estimates are where the circles touch the rhombus while the dot is the initial estimates. Clearly the lasso can set some coefficients exactly equal to zero.

The lasso method is very convenient when the model consists of many parameters and, not only estimation, but also selection of the parameters that should be in the model has to be done [2, 28, 32]. Usually, when testing many coefficients one should implement multiple correction techniques for assessing their significance. Lasso, however, lacks in the ability of multiple correction (some of the reasons will be discussed afterwards) and significance testing under the lasso method is still under research [2, 32]. However, a recently published paper, raises the issue of significance testing under the lasso method. The methods discussed in that paper are proven for the usual linear regression model cases, but they are **not** proven for the generalized linear model cases, yet [15]. However, Lockhart et al. [15] state that simulation results seem to substantiate the GLM cases also. Finally, there are a variety of different assumptions for the data matrix X, that can be made under the lasso method [28] and its inference [15]. We, however, did not make any assumptions about the data matrices (like orthogonality, standardisation etc.), but we rather used the rough generated matrices as they were.

3.3.2 Penalizing the Likelihood

The article by Tibshirani [28] was focused in the application of the lasso method on linear models. Tibshirani [28] states in the article that the lasso method can be applied in generalized linear models. Other authors like Breheny and Huang [4] and Friedman et al. [11] have applied the lasso method in generalized linear models with more details. We shall focus on the generalized version of the lasso method, where we assume a link function g to exist and that the model is well defined according to the criteria discussed in subsection 3.1.1.

Suppose that we have a generalized linear model with n observations, p predictors, response $y = (y_1, ..., y_n)$, link function g and the following data matrix:

$$X = [x_1, x_2, ..., x_p] = \begin{bmatrix} x_{11} & x_{12} & ... & x_{1p} \\ x_{21} & x_{22} & ... & x_{2p} \\ \vdots & \vdots & ... & \vdots \\ x_{n1} & x_{n2} & ... & x_{np} \end{bmatrix}$$

Where the data matrix X can be either orthogonal or non-orthogonal. Furthermore, we can either assume independence between the observations or conditional independence for the response given the explanatory variables [28].

According to Wu et al. [32], Friedman et al. [11] and Avalos et al. [2], one can maximize the penalized log-likelihood function, for obtaining the lasso estimates. More specifically one can maximize:

$$\hat{\beta}(\lambda) = argmax_{\beta} \left(\ell(\beta) - \lambda \sum_{j} |\beta_j| \right) \tag{3.2}$$

For the rest of the thesis we will use the lasso penalty form given in equation (3.2).

3.3.3 Estimating the Coefficients

Consider the objective function:

$$f(\beta, \lambda) = \ell(\beta) - \lambda \sum_j |\beta_j| \tag{3.3}$$

where $\ell(\beta)$ is the log-likelihood of the model and λ is the lasso penalty. Both $\ell(\beta)$ and $-\lambda \sum_j |\beta_j|$ are concave functions, thus the objective function is also a concave function. Unfortunately, the objective function is not differentiable but it possesses directional derivatives on its forward and backward coordinate directions [32].

As we saw in subsection 3.2.1 the directional derivatives of the objective function are:

Forward Derivative:

$$d_{u_j} f(\beta, \lambda) = d_{u_j} \ell(\beta) + \begin{cases} -\lambda \text{ if } \beta_j \geq 0 \\ +\lambda \text{ if } \beta_j < 0 \end{cases}$$

Backward Derivative:

$$d_{-u_j} f(\beta, \lambda) = d_{-u_j} \ell(\beta) + \begin{cases} +\lambda \text{ if } \beta_j > 0 \\ -\lambda \text{ if } \beta_j \leq 0 \end{cases}$$

If the log-likelihood function is differentiable, then its directional derivatives along the forward and backward directions are the same as the usual partial derivative with positive and negative signs, respectively [32]. For a differentiable log-likelihood function, the objective function is differentiable only for $\beta_j \neq 0$ [11]. This can be easily seen by setting the directional derivatives equal to zero, when we have differentiated the objective function with respect to β_j [11, 32]:

Forward Derivative:

$$d_{u_j} f(\beta, \lambda) = 0 \Leftrightarrow \frac{\partial \ell(\beta)}{\partial \beta_j} + \begin{cases} -\lambda \text{ if } \beta_j \geq 0 \\ +\lambda \text{ if } \beta_j < 0 \end{cases} = 0$$

Backward Derivative:

$$d_{-u_j} f(\beta, \lambda) = 0 \Leftrightarrow -\frac{\partial \ell(\beta)}{\partial \beta_j} + \begin{cases} +\lambda \text{ if } \beta_j > 0 \\ -\lambda \text{ if } \beta_j \leq 0 \end{cases} = 0 \Leftrightarrow$$

$$\Leftrightarrow \frac{\partial \ell(\beta)}{\partial \beta_j} + \begin{cases} -\lambda \text{ if } \beta_j > 0 \\ +\lambda \text{ if } \beta_j \leq 0 \end{cases} = 0$$

As we can see, the forward directional derivative will give us $-\lambda$ for $\beta_j = 0$, while the backward directional derivative will give us $+\lambda$ for $\beta_j = 0$. Since the rest of the results are the same, the partial derivative of the objective function exists only for $\beta_j \neq 0$ [10, 11, 32].

For estimating the coefficients of the objective function, one can either maximize the concave function $f(\beta, \lambda)$ using coordinate ascent algorithms [32], or minimize the convex function $-f(\beta, \lambda)$ using coordinate descent algorithms [11]. Whichever way we choose to do the estimation, the updating equation of β_j will be of the form[1,2] [10, 11]:

$$\hat{\beta}_j \leftarrow \frac{S(A, \lambda)}{B} \tag{3.4}$$

where A is a function of the explanatory variables x_{ij} and the coefficients $\beta_{l \neq j}$, B is a function of the explanatory variables x_{ij} and $S(A, \lambda)$ is the soft threshold function given by [10, 11]:

$$S(A, \lambda) = sign(A)(|A| - \lambda)_+ = \begin{cases} A - \lambda & \text{if } A > 0 \text{ and } \lambda < |A| \\ A + \lambda & \text{if } A < 0 \text{ and } \lambda < |A| \\ 0 & \text{if } \lambda \geq |A| \end{cases} \tag{3.5}$$

Thus, given initial values $\tilde{\beta}$, one can either use the cyclic coordinate descent [11], or the cyclic coordinate ascent algorithm [32], for updating the coefficients through the functions given in equation (3.4), until convergence has been reached [11,28,32].

3.3.4 Likelihood Approximation

The case of generalized linear models is more complicated because a cyclic coordinate update step like that in subsection 3.2.1 cannot be easily achieved. This problem arises from the fact that we cannot always solve the partial derivative with respect to β_j because we do not have linearity in the coefficients. Friedman et al. [11] suggested a quadratic Taylor series approximation of the log-likelihood, around some current estimates $\tilde{\beta}$, and an implementation of a Newton-Raphson step in the cyclic coordinate algorithm. Thus we create a hybrid of cyclic coordinate algorithm and iteratively re-weighted least squares.

The iteratively re-weighted least squares method is based on iteratively updating the current estimates until convergence is reached. Starting with initial values of the estimates, the algorithm uses a working response z_i that is based on the current value of the estimate and iterative weights w_i which are proportional to the variance of the working response, and then it updates the estimates [24].

---

[1] See Appendix B for the proof of the equation.

[2] Although the soft threshold function will be revisited afterwards, an example is given here for avoiding potential confusions. Consider minimizing, with respect to the coefficients, the penalized linear regression model $f(\beta, \lambda) = 1/2 \sum_i (y_i - X_i\beta)^2 + \lambda \sum_j |\beta_j|$. Differentiating $f(\beta, \lambda)$ with respect to a positive β_j and setting the derivative equal to zero, we get: $-\sum_i (y_i - X_i\beta) x_{ij} + \lambda = 0 \Rightarrow \beta_j = \frac{\sum_i x_{ij}(y_i - \sum_{l \neq j} x_{il}\beta_l) + \lambda}{\sum_i x_{ij}^2}$. There $A = \sum_i x_{ij}(y_i - \sum_{l \neq j} x_{il}\beta_l)$ and $B = \sum_i x_{ij}^2$. We treat $\beta_j < 0$ and $\beta_j = 0$ accordingly [11].

Consider a current estimate $\hat{\beta}$ and the linear predictor $\eta_i = X_i\beta$ of the generalized linear model, which can be estimated by $\hat{\eta}_i = X_i\hat{\beta}$. Furthermore, it is trail to compute the fitted values of the generalized linear model by $\hat{\mu}_i = g^{-1}(\hat{\eta}_i)$. The working response of the algorithm is computed by the following formula [24]:

$$z_i = \hat{\eta}_i + (y_i - \hat{\mu}_i)\frac{d\eta_i}{d\mu_i} \tag{3.6}$$

And the iterative weights are given by [24]:

$$w_i = \frac{p_i}{b''(\theta_i)(\frac{d\eta_i}{d\mu_i})^2} \tag{3.7}$$

where p_i is usually set to 1, as said before. Thus, according to Friedman et al. [11], the updated version of the log-likelihood function $\ell(\beta)$ becomes:

$$\ell_Q(\beta) = \frac{1}{2N}\sum_{i=1}^{N} w_i \left(z_i - x_i\beta\right)^2 + C(\tilde{\beta})^2 \tag{3.8}$$

where z_i and w_i are evaluated on the current estimate $\tilde{\beta}$ and $C(\tilde{\beta})$ is a constant. Note that, the term $1/N$ in equation (3.8) is a weight that can be simply absorbed by the weights w_i. Finally, Friedman et al. [11] suggested that it would be wise, when using this approximation in an algorithm, to set the fitted probabilities to 1 or 0 if they lay in the interval $(1 - 10^{-5}, 1)$ or $(0, 10^{-5})$, respectively, and the weights to 10^{-5}. This is done for ensuring that the fitted probabilities will be either 1 or 0 and they will not diverge.

3.3.5 Estimating λ

The tuning parameter λ defines the strength of the lasso penalty. As the tuning parameter increases, the number of non-zero coefficients decreases [3, 11, 28, 32]. Selecting the right λ is very crucial for the analysis as the number of coefficients who enter the model strongly depends on it [2, 3, 28]. Cross validation is usually applied for the estimation of the optimal λ [2, 3, 28], although there exist other techniques such as the use a constant λ [28], or a coefficient estimation through a sequence of λ values (path-wise coordinate algorithm), instead of using only one estimated value [11, 26].

According to Efron and Tibshirani [7], a cross validation estimate for a given value of λ would be:

$$CV(\lambda) = \frac{1}{K}\sum_{i=1}^{k}(y_i - y_i^{-k(i)})^2$$

There y_i is the observed response and $y_i^{-k(i)}$ is the cross validated response computed on the testing set (k-th fold) and whose coefficients were estimated by the training set ((k-1)-th folds).

Furthermore, Tibshirani and Tibshirani [29] suggested that a good method for finding the optimal λ is to compute the cross validation estimate for each λ in a

sequence of λ values and then choose the λ which gives the lowest cross validation error. According to Friedman et al. [11] we can compute λ_{max}, where none of the coefficients enter the model and thus $\hat{\beta} = 0$ (the null model [3]), by noticing that in the soft threshold in equation (3.5), a coefficient will stay at zero if $\lambda \geq |A|$. Thus $\lambda_{max} = max_l|A|$, where l is the coefficient index. Then, it is easy to compute a sequence from λ_{max} down to a minimum and then perform a cross validation to each λ for finding the one which minimizes the prediction error [3, 11, 29].

Moreover, Simon et al. [26], suggest a formula for computing the λ sequence when we have found λ_{max}. The formula is the following:

$$\lambda_j = \lambda_{max} \left(\frac{\lambda_{min}}{\lambda_{max}} \right)^{j/m} \quad \forall j = 1, ...m$$

where m is the length of the desired sequence, $\lambda_{min} = \epsilon\lambda_{max}$ and ϵ is chosen according to the size of the data. For big datasets $\epsilon = 0.0001$ should be enough [26]. Note that one does not need to compute the sequence all the way down to zero and thus, potential over-fitting might be avoided.

Finally, computing a λ sequence leads to path-wise solution. Simon et al. [26] also suggest that when the coefficients have been estimated for the current λ_j in the sequence, we can use those estimates as "warm starts" in the coordinate algorithm, for estimating the coefficients for the immediate next λ_{j+1}. This leads to more stable coefficients, as well as a faster coordinate algorithm.

3.3.6 Uncertainties of the Lasso Method

A serious side effect of the Lasso method is that when a group of variables is highly correlated, the lasso might choose only a subgroup of the whole correlated group [2]. Moreover, lasso sometimes allows a few irrelevant variables to enter the model [3]. Finally, according to Tibshirani [28] the standard errors for the lasso estimates are difficult to acquire, because the lasso estimates are non-differentiable functions with respect to the response.

Those problems can be reduced by bootstrap. Tibshirani [28], Avalos et al. [3] and Avalos et al. [2] suggested a bootstrap version of lasso which is called bolasso and it reduces the uncertainties of the lasso estimates. Bolasso runs a bootstrap on many samples and on each sample it estimates the parameters using the lasso approach. Then the parameters frequently selected by the lasso are taken into account, while the others are set to zero [3]. Furthermore, the tuning parameter λ can be chosen by cross validation in each bootstrap sample [3, 28], or it can be set equal to a constant, common in each bootstrap sample [28].

Avalos et al. [3] suggested using the Akaike's information criterion (AIC) for estimating the frequency threshold of bolasso. Which corresponds to an estimation of the allowed number of times that a coefficient could have been set to zero among the bootstraps. AIC is a model selection criterion which is based on the following formula [21]:

$$AIC = -2\ell(\hat{\beta}) + 2k \tag{3.9}$$

where $\ell(\hat{\beta})$ is the log-likelihood function applied on the estimated coefficients from each model under selection, and k is the length of the β vector. According to Pan [21], if we have a number of M models, we compute the AIC according to each model and then we choose the model which gives the lowest AIC value.

Finally, according to everything that has been discussed in that chapter, a complete algorithm for estimating the regression coefficients of a generalized linear model using the bolasso approach would be the one presented in the next subsection.

3.3.7 The Final Algorithm

In this subsection we present the final algorithm of the bolasso. The data matrix X, the response vector Y, the objective function and, in general, all the functions discussed previously, are assumed to have been theoretically found. Such that they can be used in the algorithm.

The initialization step of the algorithm is done by creating the total B bootstrap samples from the data matrix X. This is done by sampling rows from the data matrix with replacement. The statistic that we want to estimate here is the vector of the estimates of the objective function. Furthermore, for each bootstrap sample we need to compute the bootstrap replicate of the statistic. That is, for each bootstrap sample we need to estimate the vector of the coefficients of the objective function.

For doing that, we first need to find the optimal λ by which the likelihood will be penalized. This is done by choosing the λ which gives the minimum cross validation error, from a sequence of λ values (the generation of the λ sequence will be discussed in the next section). For each λ in that sequence, we define the training and testing sets. Furthermore, we use the training set with that λ, in the cyclic coordinate algorithm for estimating the parameters of the objective function. Then we fit the model with the estimated parameters on the testing set. The prediction error for that fold is the norm of the difference between the fitted values and the response vector Y. We repeat the procedure on the other sets, with the same λ, and we take the average of the errors. This is done for each λ in the sequence and then the one which gives the lowest error is chosen as optimal.

When we have the optimal λ, we estimate the coefficients via the cyclic coordinate algorithm. Note that, this is also done in each cross validation step for each λ, but for that case we use the training sets, and not the whole data matrix. Therefore, after finding the optimal λ, we have to re-estimate the coefficients using the whole data matrix X. First we initialize the coordinate algorithm by giving initial values for the coefficients. Usually a vector of zeros is given. Then, for the β_j coefficient, we estimate its value using the weights, the working response, the soft threshold and the optimal λ. This is done in circles starting from β_1 and finishing one circle at β_p. Each time a circle is done, we check if the norm of the difference between the newly estimated vector and the one from the previous circle, or the initial one, is smaller than a tolerance. If it is, then we are done with the cyclic coordinate algorithm and with that bootstrap sample. If it isn't, then we have to run more circles until the tolerance is reached.

After all the bootstrap replicates have been computed, we use the AIC for finding the optimal threshold. The output from the bootstrap should be a matrix of dimensions $p \times B$. Each column of the matrix corresponds to one bootstrap replicate, that is, the estimated vector of coefficients from that bootstrap sample. Each row of the matrix corresponds to the values that the specific coefficient took among the bootstrap samples. For each row in the matrix, we compute the total number of zeros. From those numbers we create a frequencies vector of length p. Those are the frequency thresholds and AIC will choose the optimal. For each value in the frequencies vector we create a model. This is done by checking which of the coefficients had been set to zero less times that the chosen threshold. For those, their mean among the bootstraps is taken as an estimate, while the others are simply set to zero. Then the AIC is computed by the equation (3.9), where k is the number of non-zero estimates in the newly computed β vector. Finally, this is done for all thresholds in the sequence and the one which gives the lowest AIC is chosen as optimal. For that optimal threshold we again check which of the coefficients have been set to zero less times that the optimal threshold and we take their mean (for each coefficient respectively), among the bootstrap values. The others are set to zero. This vector of coefficients is the bolasso estimation of the coefficients for the objective function. The complete form of the algorithm is given in algorithm 5.

Data: Data matrix X, response vector Y, link function g, formulas for working response z_i and iterative weights w_i, the number K of the folds for the cross validation, the number B of bootstrap samples.
Result: Estimated generalized linear regression coefficients using the bolasso method.
Initialization: Sample B bootstrap samples from the data matrix, with replacement.
for *Each bootstrap sample b* **do**
  Compute λ_{max} and set a value to $\lambda_{min} \geq 0$, or compute the λ sequence.
  Compute the optimal $\lambda_{optimal}$ via cross validation (using the proper cyclic coordinate algorithm on each fold, with or without warm starts).
  Cyclic coordinate step (used also in each cross validation):
  Initialization: Give initial values for β and compute the initial working responses z and weights w
  while $\|\beta_{new} - \beta_{old}\| > 1e-06$*(tol.)* **do**
    for $j \leftarrow 1$ **to** *length(*β*)* **do**
      Update β_j by the coordinate which gives the greatest reduction/growth of the objective function;
      Update the vector of parameters β;
    end
    Update the z and w with the new β;
  end
end
AIC step: Select the optimal model threshold by using the *AIC* criterion, based on all the bootstraps;
Find the "allowed" number of times a coefficient could be zero in each bootstrap. Here, k in the AIC criterion is the number of non-zero coefficients for that model.

Algorithm 5: Bolasso Algorithm

3.4 Inference of Lasso

The lasso method is relatively new and therefore, there is only a little literature for the lasso inference. Confidence intervals, false discovery rates and p-values for the lasso estimates do not exist yet. Because research around that subject was not of great concern until recently [15]. However, as said before, a paper which discusses the significance of the lasso estimates has been recently published. Lockhart et al. [15] proposed a significance test for lasso in their paper "*A significance test for the lasso*", which tests the significance of a coefficient as it enters the lasso model. They referred to this significance test as *Covariance test statistic*. The challenge, though, with the covariance test statistic is that the test is made between nested[3] models and thus, one have to find a λ value where the increase (or decrease) of the total coefficients in the model is only one at a time. That is, one have to find a λ sequence for which the total coefficients which are in the model will change by one at a time, as we move through the λ sequence from λ_{max} to λ_{min}. Those λ values are called λ-knots and, especially for high dimensional data and generalized linear models, are very hard to find [15, 22]. Lockhart et al. [15] refer to an article from Park and Hastie [22], which develops an efficient algorithm for approximating those λ-knots for a generalized linear model.

In the next subsection, we shall develop all the needed tools for the covariance test statistics and give a brief explanation of the algorithm which approximates the λ-knots. Finally, in the final subsection we shall develop the covariance test statistics for a generalized linear model.

3.4.1 λ-knots

Park and Hastie [22] developed an efficient algorithm for approximating the λ-knots for a generalized linear model with the lasso penalty. The algorithm is based on an algorithmic method which is called *the predictor-corrector method* and it can be implemented on any convex likelihood function. The main idea of the algorithm is to "predict" the next λ-knot on which the next coefficient will enter the model (or someone exits the model), based on the current coefficients that are in the model. In other words, the algorithm computes the step by which we have to jump in a given λ sequence for finding the next λ-knot.

Consider the objective function given in equation (3.3). That objective function is a concave function. On the other hand, the predictor-corrector algorithm works only on convex functions [22]. Therefore, we consider the problem of minimizing the negative objective function $-f(\beta, \lambda)$, which is convex with respect to the coefficients. That is we consider the following problem [22]:

$$\hat{\beta}(\lambda) = argmin_{\beta}\left(-f(\beta,\lambda)\right) = argmin_{\beta}\left(-\ell(\beta) + \lambda \sum_{j} |\beta_j|\right) \tag{3.10}$$

[3] By nested models in the lasso case, we mean a model on a λ_k-knot which gives an active set of total j parameters, and the immediate next model on the λ_{k+1}-knot which gives an extra coefficient and thus, the active set is of size $j + 1$ [15].

The minimization of that objective function can be done by any algorithm which minimizes convex functions (like cyclic coordinate descent), for a given λ-knot. The problem here is to find λ-knots, not to estimate the coefficients itself, although the estimation is a by-product of the process.

Let Ω be the active set of coefficients (the ones that are already in the model) for the current λ-knot. The algorithm finds all the λ-knots as we move from λ_{max} [4] down to zero and, each time it finds a knot, it updates the Ω set. Note that, a coefficient in Ω^c may enter a model in a given knot and thus, the corresponding coefficient moves into Ω, or a coefficient in Ω may exit the model and move into Ω^c. In general, a λ-knot corresponds to a value of λ for which the active set Ω changes (increases or decreases), and we wish to find all those knots [22].

There exists various algorithms in the literature for computing the exact λ paths, like **LARS** algorithm and the *Support vector machine path* procedure, but they are not applicable in the case of GLM. Because of the luck of linearity of a GLM, the Ω set could be the same for different values of λ, but the coefficients could have been estimated slightly differently. Finding the exact estimated coefficients for a given λ is very crucial when we need to find the next λ-knot in the path. The predictor-corrector algorithm computes the exact values of λ-knots and the corresponding coefficients, leading to a better and more accurate λ path, where for each value in the path the active set Ω changes. The algorithm consists of four steps, the *step length*, the *predictor step*, the *corrector step* and the *active set step* [22]. Algorithm 6 gives an overview of the predictor-corrector algorithm. Each step is briefly presented in the next subsections.

[4] By this time we will let λ_{max} be the first λ value for which the first coefficient has entered the model, not to be confused with the λ_{max} value presented previously, for which all the coefficients are zero. The reason for that, is that we take into account an intercept term, and it will be the first coefficient who enters the model. So for λ_{max} we will already have the intercept in the model [22].

Data: Main: Dataset, regression model, λ_{max}. Additional: The number K of the folds, if the CV error might need to be computed for each knot.
Result: Main: The exact λ-knots path. Additional: The estimated coefficient for each knot, the CV error of the corresponding knot.
Initialization: Compute the coefficients for the given λ_{max} and create the active set Ω, which by that time will only contain the intercept.;
while *Active set* Ω *not full* **do**
  Step 1: Step Length: Given the λ_i-knot, approximate the next knot λ_{i+1};
  Step 2: Predictor: Find an approximate of the coefficients vector β (β^{i+});
  Step 3: Corrector: Using β^{i+}, find the exact solution of β^i at λ_{i+1};
  Step 4: Active Set: Test if any coefficient moves from Ω to Ω^c, or the other way around. If Ω doesn't change, the step is too small, reduce λ_i more. If Ω changes by more than one value, the step is too big, increase λ_i. ;
end

Algorithm 6: Predictor-Corrector Algorithm

3.4.1.1 Step 1: Step Length

Consider a likelihood function that belongs to the exponential family, as discussed in subsection 3.1.1. By penalizing that likelihood and then derivating it with respect to the β coefficients, we get the following equation [22]:

$$H(\beta, \lambda) = -\frac{\partial f(\beta, \lambda)}{\partial \beta} = -X'W(y - \mu)\frac{\partial \eta}{\partial \mu} + \lambda sgn\begin{pmatrix}0\\ \beta\end{pmatrix} \tag{3.11}$$

where $f(\beta, \lambda)$ is the objective function[5], β is the vector of the coefficients, X is the data matrix, W is the diagonal matrix of weights given by equations (3.7), y is the response vector, μ is the vector of the fitted values and η is the linear predictor. Furthermore, λ is the lasso penalty and $sgn\binom{0}{\beta}$ is the sign function, which will return -1 if the corresponding β is negative, $+1$ if it is positive, and zero if it is zero [22].

For a model with an intercept, one can find λ_{max} by finding the λ value which satisfies $H((\beta_0, 0, 0..., 0), \lambda) = 0$. This value can be found by:

$$\lambda_{max} = max_j |X_j' \hat{W}(y - \bar{y}I)g'(\bar{y})|$$

where g is the link function. This λ can be used as an initialization step for the algorithm. Therefore the active set Ω contains only the intercept term on the beginning [22].

[5]Remember that $f(\beta, \lambda)$ is a concave function, thus the negative is convex.

After the initialization step, or after the k-th step in general, we wish to find the next λ-knot at which the active set changes, namely λ_{k+1}. By that time, we should have computed $\hat{\beta}(\lambda_k)$, the estimates on the previous knot. If we let $\hat{\mu}$ be the fitted values from the $\hat{\beta}(\lambda_k)$ estimates, the corresponding weighted correlations are given by [22]:

$$\hat{c} = X'\hat{W}(y - \hat{\mu})\frac{\partial \eta}{\partial \mu} \tag{3.12}$$

One should expect that $|\hat{c_j}| = \lambda_k \ \forall \ j \in \Omega$, except for the intercept term, and $|\hat{c_j}| < \lambda_k \ \forall \ j \in \Omega^c$. According to Park and Hastie [22], for a reduction step $h > 0$ on the current λ_k, the reduction in the absolute correlations can be found by:

$$c(h) = \hat{c} - hX'\hat{W}X_\Omega(X'_\Omega\hat{W}X_\Omega)^{-1}sgn\begin{pmatrix} 0 \\ \hat{\beta}(\lambda_k) \end{pmatrix} \tag{3.13}$$

where X_Ω is the data matrix consisting only of the columns of the coefficients which are in the active set Ω by that time. For finding h for which any of the coefficients in Ω^c gets the same absolute correlation as those in Ω, one have to find the minimum positive h, for every $j \in \Omega^c$ which satisfies the following equations [22]:

$$c_j(h) = |\hat{c}_j - h\alpha_j| = \lambda_k - h \tag{3.14}$$

where $\alpha_j = X'_j\hat{W}X_\Omega(X'_\Omega\hat{W}X_\Omega)^{-1}sgn\binom{0}{\hat{\beta}(\lambda_k)}$. Accordingly, equations (3.14) are equivalent with [22]:

$$h = min^+_{j\in\Omega^c}\left(\frac{\lambda_k - \hat{c}_j}{1 - \alpha_j}, \frac{\lambda_k + \hat{c}_j}{1 + \alpha_j}\right) \tag{3.15}$$

Once the next λ-knot has been found, one can continue to the predictor step. However, estimating λ_{k+1} does not necessarily mean that this value is the right knot. This value can be smaller than it should be, leading to a greater increase in the Ω set than it should, or it can be greater than it should be, leading to the same Ω set. If the new λ is greater than it should be, it means that the h step was not big enough, therefore, one should repeat the length step estimation, but with the new λ in equations (3.14) and (3.15). Finally, if the new λ is smaller than expected, meaning that h is too big, one should estimate an increase in the new computed λ, that is, increase the λ this time [22].

3.4.1.2 Step 2: Predictor Step

When the λ_{k+1}-knot has been found, $\hat{\beta}(\lambda_{k+1})$ can be linearly approximated by β^{k+}. This can be easily done by solving the equation $\beta^{k+} = \hat{\beta}(\lambda_k) + (\lambda_{k+1} - \lambda_k)\frac{\partial \beta}{\partial \lambda}$. Furthermore, $\frac{\partial \beta}{\partial \lambda}$ can be found by solving $\frac{\partial H(\beta,\lambda)}{\partial \lambda} = \frac{\partial H}{\partial \lambda} + \frac{\partial H}{\partial \beta}\frac{\partial \beta}{\partial \lambda} = 0$ with respect to the current active set Ω and the current estimates $\hat{\beta}(\lambda_k)$ [22].

However, Park and Hastie [22] state that this step is not always necessary for problems with big data. This is because in such problems, the active set Ω changes with small decreases in λ and thus, no approximation is needed. On the other hand, the predictor step could still be used in the cases were the h step is big.

3.4.1.3 Step 3: Corrector Step

The corrector step computes the exact value of $\hat{\beta}(\lambda_{k+1})$. Given the value of the linear approximation β^{k+} (or $\hat{\beta}(\lambda_k)$ if the predictor step is skipped) as warm starts, one can minimize the objective function $-f(\beta, \lambda)$ by means of any algorithm which minimizes a convex function. According to Park and Hastie [22], the time that the algorithm would need to compute the exact value $\hat{\beta}(\lambda_{k+1})$ won't take took long, because β^{k+} (or $\hat{\beta}(\lambda_k)$), would lay near the true value of $\hat{\beta}(\lambda_{k+1})$. After each corrector step the algorithm should check if the active set Ω should change, this is done in the fourth step of the algorithm.

3.4.1.4 Step 4: Active Set Step

Every time the algorithm exits the corrector step, a check should be done for any potential update of the active set Ω. According to Park and Hastie [22], this can be done by checking the absolute correlations given in equation (3.12). This time, the weights $\hat{W}$ and the fitted values $\hat{\mu}$ are computed by $\hat{\beta}(\lambda_{k+1})$, the estimate from the corrector step.

One should check if $|\hat{c}_j| > \lambda_{k+1}$ for every $j \in \Omega^c$. Every time this inequality is satisfied, the corresponding j predictor should be moved from the non-active set Ω^c into the active set Ω. After each active set step, a corrector step should be again applied, then again an active set step. This circle should be done as many times as needed such that the active set Ω is no longer changing. Finally, any coefficient that has a value of zero should be removed from the active set. Then we go again to step one for finding the next λ-knot [22].

The algorithm provides also a step for checking if any coefficient should be set to zero before moving down to the next knot. That is, if we have a reduction in the active set Ω [22]. However, this step was not needed in our analysis, but if one wishes to know more about it, the article by Park and Hastie [22] is the best way to do it.

The predictor-corrector algorithm is an efficient algorithm that approximates the λ-knots in the case of GLM. It might not be of great importance if one wishes to run bootstrap and for each bootstrap sample find the exact λ-knot, but it is of great importance if one wishes to assess the significance of the lasso estimates.

3.4.2 Assessing the Significance

This subsection concerns the significance test for the lasso estimates, which was proposed by Lockhart et al. [15]. We will however, consider only the case of the generalized linear models, although its efficiency has not yet been proven by the authors. The covariance statistic tests the significance of a coefficient which enters the model, in a sequence of nested lasso models. Therefore, the use of the predictor-corrector algorithm is essential for finding the correct λ-knots which will give nested models.

The null hypothesis which is being tested here is that "*the current active coefficients in the lasso model are the ones which should be in the model*". The covariance

statistic tests the significance of a coefficient that is going to enter the model, given the ones that already are in the active set by that time. In other words, we test H_0: "$\beta_j = 0$ *given the other coefficients in the model*" (note that the hypothesis is conditional; we come back to this later in this section), for the j-th predictor to enter the model. Although the interpretation is actually not so straightforward [15];

The lasso method estimates the coefficients with an adaptive and greedy way. For that reason, the residual sum of squares under the null hypothesis becomes much larger than the X_1^2 distribution and thus, the usual chi-squared test is no longer applicable. Lockhart et al. [15] however, state that the covariance test statistic accounts for the adaptive sequence of the lasso estimates and that it balances between the shrinkage and the adaptivity of the lasso. They have shown, that under the null hypothesis, the covariance statistics follows asymptotically the $Exp(1)$ distribution, for any linear model. Based on simulation results they argue that this assumption also holds for the generalized linear model cases.

We consider estimates from the equation (3.10), where all the requirements about the exponentiality of the likelihood function, discussed in section 3.1, apply also here. Let Ω be the active set of parameters just before the effect of the λ_k-knot (that is, the non-zero coefficients from the λ_{k-1}-knot;) and let β_j be the predictor which is going to enter the model when estimation is done on the λ_k-knot. That is, when we estimate on the λ_k-knot the active set will become $\Omega \cup \{j\}$. We wish to test the significance of the j-th predictor. Then Lockhart et al. [15] define the covariance statistic for a generalized linear model under the lasso penalty to be:

$$T_k = \frac{\langle I^{-1/2}S, X\hat{\beta}(\lambda_{k+1})\rangle - \langle I^{-1/2}S, X_\Omega \tilde{\beta}_\Omega(\lambda_{k+1})\rangle}{2} \tag{3.16}$$

where $I = \bigtriangledown^2(\ell(\beta))$, $S = \bigtriangledown(\ell(\beta))$ calculated on the active set Ω. Those equations can be found by the weighted procedure $z = \eta + I^{-1}S$ of a generalized linear model (although previously we used another formula for the weights.). Furthermore, λ_{k+1} is the value of the next knot for which the active set Ω changes (becomes $\Omega \cup \{j\} \cup \{j+1\}$), $\hat{\beta}(\lambda_{k+1})$ are the estimated coefficients penalized by λ_{k+1} under the $\Omega \cup \{j\}$ active set of predictors, and $\tilde{\beta}_\Omega(\lambda_{k+1})$ are the estimated coefficients penalized by λ_{k+1} under the Ω active set of predictors. Finally, the symbols $\langle, \rangle$ indicate the inner product of those matrices [15].

Note that, for testing the significance og the j-th predictor (inserted in the model by the λ_k-knot), we use the next λ_{k+1}-knot. This is done because if we had computed T_k on the λ_k-knot we should have gotten $T_k = 0$ because $\hat{\beta}_\Omega(\lambda_k) = \tilde{\beta}_\Omega(\lambda_k)$, since the solution of the full problem for λ_k-knot restricted on the Ω set($\hat{\beta}(\lambda_k)$ under the active set Ω and not $\Omega \cup \{j\}$;), is the same as the solution of the reduced problem ($\tilde{\beta}_\Omega(\lambda_k)$ under Ω). Therefore $X\hat{\beta}(\lambda_k) = X_\Omega \hat{\beta}_\Omega(\lambda_k) = X_\Omega \tilde{\beta}_\Omega(\lambda_k)$. Moreover, the new predictor will have gained its full power on λ_{k+1}-knot [15].

For each covariance statistic T_k we can compute the corresponding p-value. Big values of T_k mean that the current coefficient has a big impact on the model. This will result to a small p-value, which means that the current coefficient is significant, and the null hypothesis will be rejected. In simpler words, this means that the Ω set (the set of active coefficients without the one being tested), does not contain

all the truly active coefficients, therefore, the new one has to enter the set. Finally, Lockhart et al. [15] show that the degrees of a freedom for a model with k predictors are simply k under the lasso model. Therefore the degrees of freedom for the T_k statistic are $k + 1 - k = 1$.

Discussions have been made around the paper of Lockhart et al. [15] which concern the lasso inference and whether the covariance test statistic can be trusted or not, see Bühlmann et al. [23], Cai and Yuan [5], Fan and Ke [9] and Lv and Zheng [17]. Those articles mainly concern the occurrence of the p-values from the covariance test, as well as their interpretation. For example Bühlmann et al. [23], argue against the interpretation of the p-values by stating that the p-values from the covariance statistic are based on a conditional test (given the other active coefficients in the model) and therefore, cannot be interpreted by the same way as the usual p-values. However, all those discussion-papers congratulate the authors of "A significance test for the lasso". For our case, since there is not other publications around the significance of the lasso estimates, we shall consider the paper by Lockhart et al. [15] for assessing the significance of our estimates.

All those methods that were presented in this chapter, will be combined with the case-crossover design for our problem. The next chapter will adapt the methods discussed here to our model. Furthermore, some algorithmic modifications will also be presented, which were mainly applied because of time efficiency problems.

Chapter 4

Adaptation to the Theory

The purpose of this chapter is to adapt the theory discussed so far to our problem. Section 4.1 discusses the way by which the datasets were modified. Furthermore, in sections 4.2 and 4.3, an adaptation of the theory given on chapters 2 and 3 is presented.

4.1 Drug Frequencies and Information Flow

In this section, we give an overview of how the two datasets were evolved during the stages of the analysis. In figure 4.1 the initial frequencies of the 775 drugs are given. Drugs with total intakes less than 100 (below the red line) were excluded from the analysis because the information of those drugs was too little in conjunction with the total 75.000 patients. Those drugs would probably not give any important results but they could significantly reduce the running time of the algorithms, which was a major issue even for the rest of the drugs[1]. On the right up corner of the figure, a zoomed version of the figure is printed so that the red line can be more easily seen.

Figure 4.2 shows how the total number of patients were reduced during the analysis. The reason for that reduction was because not all patients could contribute any information on the analysis. After defining the windows of our analysis, which will be discussed in subsection 4.2.1.1, some patients had the same exposure frequencies on both case and control windows. Thus, according to subsection 2.1.1, they could not contribute anything on the analysis so they were removed.

[1]See appendix A for the challenges of Big Data.

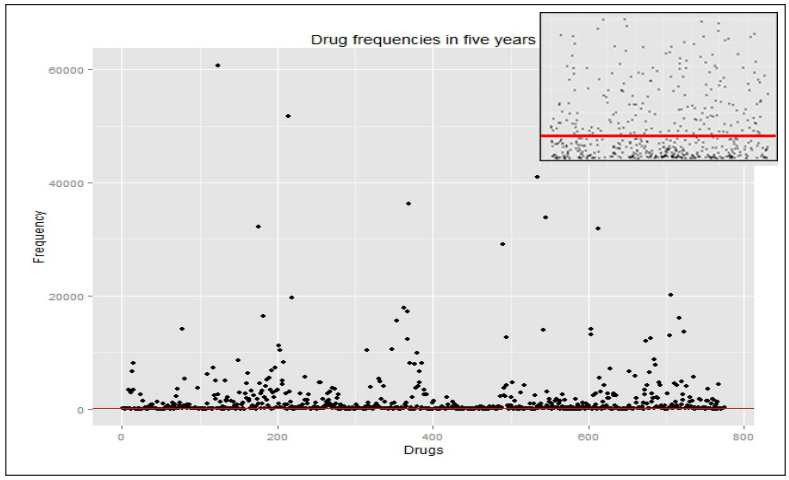

Figure 4.1: This figure shows the initial frequencies multiplied by 75.000 which is the initial total number of patients for both datasets. Drugs below the red line were excluded by the analysis. The red line is placed on $y - axis = 100$. Note that because the two datasets were generated separately, the total drugs included in the analysis are not the same for both datasets, because the intakes depend on random number generations. This can also be seen in stage 3 in figure 4.2. But the line is approximately the same for both datasets.

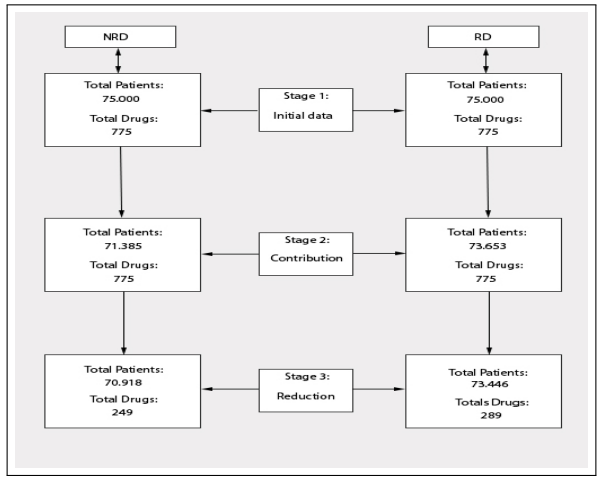

Figure 4.2: This figure shows the changes in the two datasets throughout the analysis. Stage 1 is the initial stage where the two datasets were generated and it represents the two complete datasets. Stage 2 is where the contribution check took place. On that stage only the patients that could contribute in the analysis were taken into account, the rest were excluded. Stage 3 is the reduction stage, were only drugs with total intakes more than 100 were taken into account. Note that on stage 3 we also have a reduction in the total number of patients. This happened because, by removing some drugs, some patients ended up with same exposures on both windows and they should therefore be excluded, again.

4.2 Basic Model Design

This section forms the basic model of the analysis to follow. It starts with subsection 4.2.1 where we develop the case-crossover layout for our problem. It then continues to subsection 4.2.2 where the conditional logistic regression likelihood is being adapted to the current case-crossover design. Finally, subsections 4.2.3, 4.2.4 and 4.2.5 concern some mathematical models, that were used in the analysis.

4.2.1 The Case-Crossover Layout

The analysis was based on the case-crossover design described in chapter 2. The subjects were the patients and each patient was both the case and the control of itself, but of course on different time periods. Therefore, for reasons discussed in chapter 2, personal characteristics like age or sex were excluded from the model. The analysis was based on the time period between 2008 to 2012, where all the drug intakes for each patient were recorded. The follow up time period for each patient stopped when he or she experienced the first MI event. Thus, the observation periods varied among the patients since each one of them experienced the MI event at times independent from the other patients.

4.2.1.1 Windows and Triggers

For the whole analysis we used a referent window for each patient which consisted of one case and one control window. Those referent windows were placed after observing the event for each patient and were therefore not placed a priori with respect to disjoint strata. Thus, our design is a non-localizable design and, more specifically, a unidirectional design [13]. Furthermore, the fact that only one[2] MI event occurred for each patient in the period of the analysis, constricted the total number of case windows to one. Moreover, we used only one control window such that seasonal[3] bias would be reduced as much as possible. The way that the control and case windows were placed will be discussed in the next subsection.

The potential triggers of the analysis were of course the drugs. The analysis began with the generated datasets at stage 3 in figure 4.2, thus the total drugs for the NRD dataset were 249, while the total drugs for the RD dataset were 289. The estimation and the significance was made only for the main effects of each drug, that is, no interactions were modelled. The reasons for not testing interactions will be discussed later. However we briefly discuss how potential interactions can be tested.

[2]For a real dataset, the total number of MI events in the observation period for each patient, might not be equal to one. However, the drug intakes might change after the first occurrence of the event and therefore, we stopped the study for each patient when he or she experienced the first MI event. Therefore here, we consider only one MI event happening in the observation period.

[3]Some drugs are taken more often on specific seasons or days of the year.

4.2.1.2 Induction Time and Effect Period

The drugs were assumed to have minimum induction time zero and an effect and washout period of seven days each. The fact that the drugs were assumed to have a minimum induction time equal to zero means that each drug began affecting the patient immediately after the first intake. Moreover, the effect period of seven days means that each drug was setting the patient at a risk period which varied in seven days and therefore, if that drug could have caused the event, the event should have occurred in that period of seven days. For that reason, the case window was placed seven days before the MI event for each patient. That is, if a patient for example experienced the MI event at the day number ten, then the case window is from day four to day ten. The effect period was also assumed to have equal risky periods. This means that each of the seven days could only give the same amount of risk to the patient. Finally, the washout period of seven days let us put the control window twenty one days before the MI event for each patient. For summing up, the design was seven days for the effect period of the control window, seven days for the washout period of the control window and seven days for the effect period of the case window. By doing that we ensured that the control and case windows did not overlap, which is very important for a case-crossover analysis. Finally, each case and control window was placed according to the MI event of each patient and therefore, the dates on which the referent windows were placed differed among the patients. In figure 4.4 an illustration of the hazard period is given.

The case and the control windows could be exposed or unexposed. For testing the main effects of the drugs, a window was considered exposed if the date of the drug intake was within the time interval of the corresponding window. Each patient could take the same drug more than two times[4], only if the interval between the corresponding drug intakes was at least of seven days length. On the other hand, each patient could have taken two or more different drugs within overlapping periods. Thus, the investigation of interaction effects was possible. Two or more drugs were considered to potentially interact if their effect periods overlapped. Therefore, for investigating interactions, a window was considered exposed if an interaction took place within the period of the window and the interaction interval crossed over the end of the time period for that window.

Figure 4.3 gives an example of how the windows could have been placed for three patients and two drugs. Consider the first patient. The control window of that patient is completely unexposed, while the case window is exposed. Note that the case window is only exposed on the second drug, not the first nor their interaction. The control window of the second patient is exposed on the second drug and its case window is exposed both on the first drug, on the second drug and on their interaction. Finally, the control window of the third patient is only exposed on the first drug, while the case window of the same patient is exposed on both the first drug, the second drug as well as their interaction. On the description of the figure one can find more information about how a window is considered exposed.

[4]By more than two times we mean that more than two prescriptions could have been given to a patient. A drug could have been taken on a daily basis or not, but this is not of our interest, we define the intakes as the number of prescriptions given.

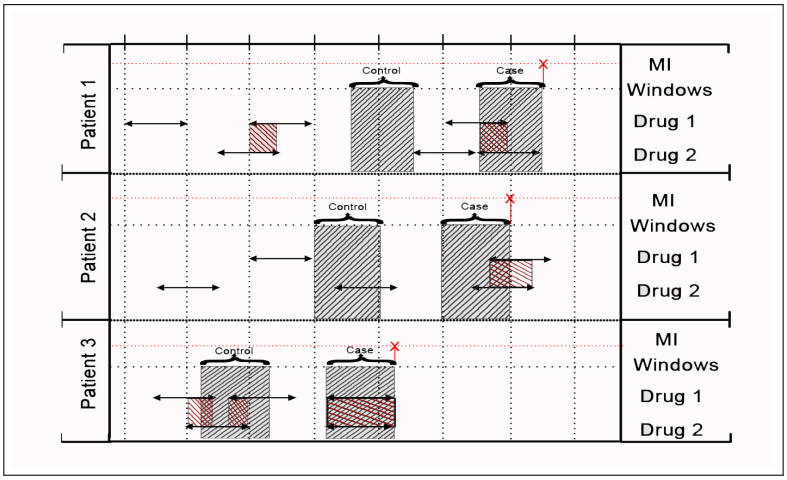

Figure 4.3: This figure gives an example of how the control and case windows could have been placed. For each patient the red X is where the MI event occurred, then according to the time of the event, the windows are placed. Each window is of seven days length and between the control and case windows, there is a washout period which is also of seven days length. Each arrow corresponds to the effect period of the drug. The beginning of each arrow is the time of the intake, the end of each arrow is the end of its effect period and the length of each arrow is seven days. Arrows in the same row correspond to the same drug, but for different intakes. That is, one can take the same drug more than one times, as long as the effect periods do not overlap. Furthermore, the black lined boxes correspond to the "captures" of each window. If such a box contains the beginning of an arrow, followed by the whole arrow or a part of it, then the window is considered exposed to the corresponding drug. Finally, the red lined boxes correspond to potential interactions between the drugs. If an interaction box is inside or partially inside a window box and it crosses over the right end of the window box, then the window is considered exposed to that interaction. From the figure, it can be easily seen that the maximum length of an interaction is of seven days, while the minimum length is of one day (since we count days).

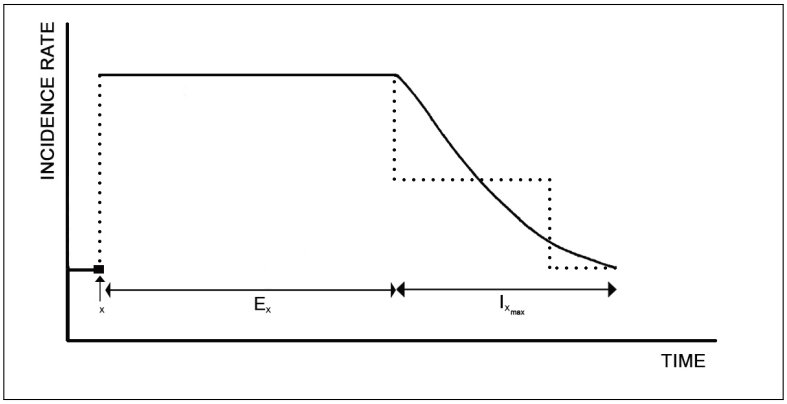

Figure 4.4: This figure is an illustration of the hazard period of each drug for the current model and it is made according to the illustration given by Maclure [18]. The drug intake is at point "x", there the effect period immediately begins. The effect period E_x has a length of seven days and then it is followed by the washout period $I_{x_{max}}$, which is also of seven days length.

4.2.2 The Conditional Logistic Regression Likelihood

In this subsection, we develop the conditional likelihood for our case-crossover design. The likelihood is a modified version of the likelihood discussed in subsection 2.2.2. According to subsection 2.2.2, the conditional logistic likelihood for one case and one control window, under a non-localizable design, would be the following:

$$L_{nonloc}\left(\beta\right) = L\left(\beta\right) = \prod_{i=1}^{N} \frac{e^{X_{it_i}\beta}}{\sum\limits_{t \in W_i} e^{X_{it}\beta}} = \prod_{i=1}^{N} \frac{e^{X_{it_i}\beta}}{e^{X_{it_i}\beta} + e^{X_{is_i}\beta}}$$

where N is the total number of patients and p is the total number of coefficients in the model, t_i corresponds to the case window of the i-th patient and s_i corresponds to the control window of the i-th patient. Furthermore, X_{it_i} is a vector of length p corresponding to the exposure of each drug on the case window, and X_{is_i} is a vector of the same length corresponding to the exposures on the control window. Both vectors contain the values 0 and 1 for non-exposure and exposure respectively.

This likelihood can be modified in a simpler form. Consider two X matrices: X_{cases}, which contains all the case vectors, and $X_{controls}$, which contains all the control vectors, both of dimensions $N \times p$, where the i-th row of both matrices corresponds to the i-th patient. Thus, we have the following matrices:

$$X_{cases} = [x_{1_{case}}, x_{2_{case}}, ..., x_{p_{case}}] = \\ = \begin{bmatrix} x_{11_{case}} & x_{12_{case}} & ... & x_{1p_{case}} \\ x_{21_{case}} & x_{22_{case}} & ... & x_{2p_{case}} \\ \vdots & \vdots & ... & \vdots \\ x_{n1_{case}} & x_{n2_{case}} & ... & x_{np_{case}} \end{bmatrix}$$

$$X_{controls} = [x_{1_{control}}, x_{2_{control}}, ..., x_{p_{control}}] = \\ = \begin{bmatrix} x_{11_{control}} & x_{12_{control}} & ... & x_{1p_{control}} \\ x_{21_{control}} & x_{22_{control}} & ... & x_{2p_{control}} \\ \vdots & \vdots & ... & \vdots \\ x_{n1_{control}} & x_{n2_{control}} & ... & x_{np_{control}} \end{bmatrix}$$

Consider also two response vectors Y_{cases} and $Y_{controls}$, both of dimensions $N \times 1$, where the response vector for the cases contains only the element 1 and the response vector for the controls contains only the element 0. This comes from the fact that we have conditioned on that only one event can happen for each patient and that event should be in the case window. Then the previous likelihood can be written in the form:

$$L\left(\beta\right) = \prod_{i=1}^{N} \frac{e^{X_{it_i}\beta}}{e^{X_{it_i}\beta} + e^{X_{is_i}\beta}} = \prod_{i=1}^{N} \frac{1}{1 + e^{-X_i\beta}} \tag{4.1}$$

where $-X_i = -(X_{i\text{ case}} - X_{i\text{ control}}) = -(X_{it_i} - X_{is_i})$, that is, we subtract the control vector from the case vector for each patient. Avalos et al. [3] suggested that

for a case-crossover design with one case and one control window, the likelihood can be written in the form of the unconditional likelihood for the binary logistic regression with no intercept and constant response which is equal to one. The fact that the response is constant equal to one, becomes more clear when we subtract the $Y_{controls}$ from Y_{cases} response vector. According to Avalos et al. [2] a case-crossover design with more than one control windows, can be model as a Cox model, however this is not our case. Note, however, that now the data matrix X consists of values $-1, 0, 1$.

For the rest of the analysis the likelihood given in equation (4.1) was used. The data matrix X was of dimensions $N \times p$, where the i-th row corresponded to the substruction of the control vector from the case vector of the i-th individual, and a constant response Y of dimensions $N \times 1$ which contained only the element one.

4.2.3 Generalized Linear Model

In this section, we shall develop some modifications of the likelihood, based on the theory discussed in section 3.1. Those modifications were not only used for creating the final form of the lasso equations, but they were also used in the algorithms of this analysis.

The first step is to rewrite the likelihood given in equation (4.1) in a convenient form for showing that we are dealing with a generalized linear model. Thus, if we let p_i be the contribution of the i-th individual to the likelihood in equation (4.1), that is $p_i = \frac{1}{1+e^{-X_i\beta}} = \frac{e^{X_i\beta}}{1+e^{X_i\beta}}$, then the likelihood can be written in the following form:

$$L(\beta) = \prod_{i=1}^{N} \frac{1}{1+e^{-X_i\beta}} = \prod_{i=1}^{N} p_i = \prod_{i=1}^{N} (p_i)^{y_i}(1-p_i)^{1-y_i} \tag{4.2}$$

Note that since $y_i = 1 \; \forall \; i$, $1 - y_i = 0 \; \forall \; i$, the second term always vanishes. The log-likelihood is of the form:

$$\begin{aligned} \ell(\beta) = log(L(\beta)) &= \sum_{i=1}^{N} log\left((p_i)^{y_i}(1-p_i)^{1-y_i}\right) = \\ &= \sum_{i=1}^{N} (y_i log(p_i) + (1-y_i) log(1-p_i)) = \\ &= \sum_{i=1}^{N} (y_i logit(p_i) + log(1-p_i)) \end{aligned} \tag{4.3}$$

Thus, it is obvious that the log-likelihood, and therefore the likelihood, belongs to the exponential family with:

- $$\theta_i = logit(p_i) = X_i\beta = \eta_i$$
- $$b(\theta_i) = -log(1-p_i)$$

- $$c(y_i, \phi) = 0$$
- $$\alpha_i(\phi) = \phi/1 = \phi$$
- $$\phi = 1$$

Therefore the link function is:

$$g(x) = \frac{e^x}{1 + e^x}$$

And the linear predictor is:

$$\eta_i = g(p_i)^{-1} = logit(p_i)$$

Finally, we can compute the expected response $E(Y_i)$ and the variance $Var(Y_i)$:

$$\begin{aligned} E(Y_i) = \mu_i = b'(\theta_i) = \left(-log(\frac{1}{1 + e^{\theta_i}})\right)' = \\ = \frac{1 + e^{\theta_i}}{(1 + e^{\theta_i})^2} e^{\theta_i} = \frac{e^{\theta_i}}{1 + e^{\theta_i}} = p_i = X_i\beta \end{aligned}$$

$$\begin{aligned} Var(Y_i) = \sigma^2 = b''(\theta_i)\alpha_i(\phi) = \frac{e^{\theta_i}}{(1 + e^{\theta_i})^2} \cdot 1 = \\ = p_i(1 - p_i) \end{aligned}$$

Based on the above results we can continue to the next subsection for finding the equations for the working response and the weights.

4.2.4 Weights and Working Response

According to subsection 3.3.4, we can compute the working response z_i and the weights w_i of a generalized linear model from equations (3.6) and (3.7). Thus, for the equations described in subsection 4.2.3, we get the following results for the working response and the weights respectively:

$$\begin{aligned} z_i = \eta_i + (y_i - \mu_i)\frac{d\eta_i}{d\mu_i} = X_i\beta + (y_i - \mu_i)\frac{logit(p_i)}{d\mu_i} = \\ = X_i\beta + (y_i - \mu_i)\frac{logit(\mu_i)}{d\mu_i} = \\ = X_i\beta + (y_i - \mu_i)\frac{1}{\mu_i(1 - \mu_i)} = X_i\beta + \frac{y_i - p_i}{p_i(1 - p_i)} \end{aligned} \tag{4.4}$$

$$w_i = \frac{\rho_i}{b''(\theta_i)(\frac{d\eta_i}{d\mu_i})^2} = \frac{1}{\left(\frac{e^{\theta_i}}{(1+e^{\theta_i})^2}\right)\left(\frac{1}{p_i(1-p_i)}\right)^2} =$$
$$= \frac{1}{(p_i(1-p_i))\left(\frac{1}{p_i(1-p_i)}\right)^2} = p_i(1-p_i) \tag{4.5}$$

Because, $\mu_i = p_i$, as said in subsection 4.2.3.

4.2.5 Concavity of the log-likelihood

The final step before developing the penalized lasso likelihood is to identify the kind of curvature that the log-likelihood function given in equation (4.3) has. This is also an important step because the kind of curvature of the log-likelihood determines the kind of coordinate algorithm that can be used.

The log-likelihood function given in equation (4.3) is a concave function with respect to the β coefficients. This can be easily seen by rewriting the function in the following form:

$$\ell(\beta) = \sum_{i=1}^{N} (y_i log(p_i) + (1-y_i) log(1-p_i)) =$$
$$= \sum_{i=1}^{N} (y_i logit(p_i) + log(1-p_i)) =$$
$$= \sum_{i=1}^{N} \left(y_i X_i \beta + log\left(\frac{1}{1+e^{X_i\beta}}\right)\right) =$$
$$= \sum_{i=1}^{N} \left(y_i X_i \beta - log\left(1+e^{X_i\beta}\right)\right)$$

and since e^x and $log(x)$ are convex functions, then $log(1+e^x)$ is also a convex function. That makes $-log(1+e^x)$ a concave function.

The results that were developed in section 4.2 are very important for the further analysis. Those results form the basic model where the lasso method on the next section is going to be based on. Not all of the above results are necessary if one wishes to use the lasso method, but they are of great importance for the implementations of the algorithms.

4.3 The Lasso Design

In this section we adapt the lasso method to our case-crossover layout. Subsections 4.3.1 and 4.3.2 concern the objective function, the partial derivatives as well as the soft threshold for our model. Moreover, subsection 4.3.3 gives a brief description of how the optimal λ was found in our analysis. This was also the most difficult and time consuming part of the whole analysis. Furthermore, subsections 4.3.4 and 4.3.5 concern the Akaike's information criterion and discuss the reason for not testing interactions, respectively. Finally, in subsection 4.3.6 we adapt the covariance test statistic to our problem.

4.3.1 The Objective Function

As has been showed in the previous section, the conditional log-likelihood is a concave function. Concavity is also carried in the penalty of the lasso [3]. Therefore we can either maximize the concave function $f(\beta, \lambda) = \ell(\beta) - \lambda \sum_{j=1}^{p} |\beta_j|$ using the cyclic coordinate ascent algorithm, or we can minimize the convex function $-f(\beta, \lambda) = -\ell(\beta) + \lambda \sum_{j=1}^{p} |\beta_j|$ using the cyclic coordinate descent algorithm, as have been said in subsection 3.3.3.

For the log-likelihood given in equation (4.3) one can derivate it with respect to one of the coefficients, but cannot solve the derivative with respect to that coefficient. This happens because the coefficient will stay in the exponential part of the log-likelihood even after the derivation. Therefore, a quadratic Taylor approximation of the log-likelihood around a given value $\tilde{\beta}$ of the coefficients vector would be useful. Friedman et al. [11] suggested to use an approximation of this log-likelihood by expanding it using Taylor series around a current estimate $\tilde{\beta}$ and then use the hybrid form of cyclic coordinate descent algorithm and iteratively reweighed least squares.

The Taylor expansion around a current estimate $\tilde{\beta}$ of the log-likelihood given in equation (4.3) is:

$$\ell_Q(\beta) = -\frac{1}{2} \sum_{i=1}^{N} w_i (z_i - X_i \beta)^2 + C(\beta)^2 \tag{4.6}$$

where z_i and w_i are the working response and the iterative weights, respectively, given in subsection 4.2.4 and applied on the current estimate $\tilde{\beta}$, and $C(\beta)^2$ is a constant. Therefore, we let the objective function be the following:

$$R(\beta, \lambda) = -f(\beta, \lambda) = -\ell_Q(\beta) + \lambda \sum_{j=1}^{p} |\beta_j| \tag{4.7}$$

which is a convex function, and we minimize this with respect to the β coefficients.

4.3.2 Partial Derivatives and the Soft Threshold

The partial derivatives of the objective function with respect to β_j are:

For $\beta_j > 0$:

$$\frac{\partial R(\beta,\lambda)}{\partial \beta_j} = \sum_{i=1}^{N}\left(w_i(z_i - x_{ij}\beta_j - \sum_{l\neq j} x_{il}\beta_l)(-x_{ij})\right) + \lambda = \\ = -\sum_{i=1}^{N}\left(w_i(z_i - x_{ij}\beta_j - \sum_{l\neq j} x_{il}\beta_l)x_{ij}\right) + \lambda$$

For $\beta_j < 0$:

$$\frac{\partial R(\beta,\lambda)}{\partial \beta_j} = \sum_{i=1}^{N}\left(w_i(z_i - x_{ij}\beta_j - \sum_{l\neq j} x_{il}\beta_l)(-x_{ij})\right) - \lambda = \\ = -\sum_{i=1}^{N}\left(w_i(z_i - x_{ij}\beta_j - \sum_{l\neq j} x_{il}\beta_l)x_{ij}\right) - \lambda$$

For $\beta_j = 0$: It does not exist.

Let us consider the first case where $\beta_j > 0$. Setting this equation equal to zero and solving with respect to β_j we get the following:

$$-\sum_{i=1}^{N} w_i x_{ij}(z_i - \sum_{l\neq j} x_{il}\beta_l) + \beta_j \sum_{i=1}^{N} w_i x_{ij}^2 + \lambda = 0 \Rightarrow$$

$$\Rightarrow \beta_j = \frac{\sum\limits_{i=1}^{N} w_i x_{ij}(z_i - \sum\limits_{l\neq j} x_{il}\beta_l) - \lambda}{\sum\limits_{i=1}^{N} w_i x_{ij}^2}$$

Respectively we get for $\beta_j < 0$ that:

$$-\sum_{i=1}^{N} w_i x_{ij}(z_i - \sum_{l\neq j} x_{il}\beta_l) + \beta_j \sum_{i=1}^{N} w_i x_{ij}^2 - \lambda = 0 \Rightarrow$$

$$\Rightarrow \beta_j = \frac{\sum\limits_{i=1}^{N} w_i x_{ij}(z_i - \sum\limits_{l\neq j} x_{il}\beta_l) + \lambda}{\sum\limits_{i=1}^{N} w_i x_{ij}^2}$$

Therefore we can define the estimating equation for β_j to be the following:

$$\hat{\beta}_j(\lambda) \leftarrow \frac{S\left(\sum_{i=1}^{N} x_{ij} w_i (z_i - \sum_{l \neq j} x_{il}\beta_l), \lambda\right)}{\sum_{i=1}^{N} w_i x_{ij}^2} \tag{4.8}$$

With the soft threshold function being:

$$S(A, \lambda) = sign(A)(|A| - \lambda)_+ = \begin{cases} A - \lambda & \text{if } A > 0 \text{ and } \lambda < |A| \\ A + \lambda & \text{if } A < 0 \text{ and } \lambda < |A| \\ 0 & \text{if } \lambda \geq |A| \end{cases}$$

where $A = \sum_{i=1}^{N} x_{ij} w_i (z_i - \sum_{l \neq j} x_{il}\beta_l)$.

4.3.3 Estimation of λ

This was the most important and most difficult part of the whole analysis. It was difficult both for computational reasons and for the complexity of the algorithms. It was important because the choice of the right λ value was crucial, as it has been discussed in subsection 3.3.5. However, we discuss those problems in subsection 4.3.3.4 that follows. In this subsection we first comment on the three methods that were initially implemented for finding the optimal λ, but they didn't work. Then we develop the main steps for the predictor-corrector algorithm discussed in subsection 3.4.1 and we adapt the algorithm to our problem. The reader should have in mind that bootstrap was also implemented (we will discuss this afterwards), and for each bootstrap sample we had to find the optimal λ.

4.3.3.1 The First Three Methods

The first method that was used was a constant λ value. This λ was constant and common to all bootstraps, as proposed by Tibshirani [28]. Many different λ values were chosen arbitrary for finding the one which gives the best results. A modified version of the constant λ was also used. This modification was that we first found the optimal λ from a sequence of λ values, using cross validation on the initial data matrix. Then we used this value as a constant in each bootstrap. Both approaches were time efficient and 1000 bootstrap samples were generated in less than 20 minutes. However, the results were not satisfactory. For both datasets only one coefficient was set different than zero and with an extremely big standard deviation. This could be the right result for the RD dataset, but not for NRD, since we know that in this dataset we have surely more than one coefficient that caused the MI event. Therefore, those approaches were completely non-informative and were skipped.

The second method was to compute a sequence of λ values ranging from λ_{max} to λ_{min} and choose the value of λ that gave us the minimum cross validation error.

Friedman et al. [11] suggested to use a minimum λ value of $\lambda_{min} = \epsilon\lambda_{max}$, where $\epsilon = 0.0001$ and then compute a sequence of length $K = 100$. This however did not give good results in our analysis. The reason was that the λ values did not give a piecewise increase in the active set of coefficients Ω, but they rather gave many gaps in the in-between values. An example can be seen in figure 4.5. According to Park and Hastie [22] and Lockhart et al. [15], this should be expected for a generalized linear model. Therefore, we could not choose that kind of sequence for the λ values, since they would not give us right results. Although, not choosing the exact optimal λ at each bootstrap, could be of not so great importance, but as we discussed in subsection 3.4.2, finding the right λ values (λ-knots), is indeed of great importance for assessing the significance of a model after the estimation of the parameters.

Finally, the third method was similar to the second, but this time we used a λ sequence which was generated according to the formula given by Simon et al. [26]. After we had found λ_{max}, we computed the sequence of λ_j from $\lambda_{min} = \epsilon\lambda_{max}$ according to the following formula: $\lambda_j = \lambda_{max}\,(\lambda_{min}/\lambda_{max})^{j/m}$ for $j = 1, ..., m$, where m was the length of the sequence. Different values of m were used and the best "approximate" was $m \sim 1000$, which gave smaller differences to the λ values since the m value was big enough to result to a more "detailed" sequence. This resulted to smaller gaps, but not small enough for finding the exact optimal[5] value of λ. Therefore this method was also skipped and we ended up with the predictor-corrector algorithm which is presented in the next subsection.

[5]By saying the exact optimal λ we argue with the following: Let us suppose that a value in the λ sequence, say λ_j, gives us a fairly low cross validation error and a number of k active coefficients, and the immediate next value, say λ_{j+1}, gives us also a fairly low cross validation error, but a number of $k + 5$ active coefficients. Because of that gap of 4 values in the active coefficients, we know that there has to be four more λ values in the interval $(\lambda_j, \lambda_{j+1})$ that would give us active sets of $k + 1, k + 2, k + 3$ and $k + 4$, respectively. But how do we know that one of those λ will not give a better minimization of the cross validation error than λ_j and λ_{j+1} do?

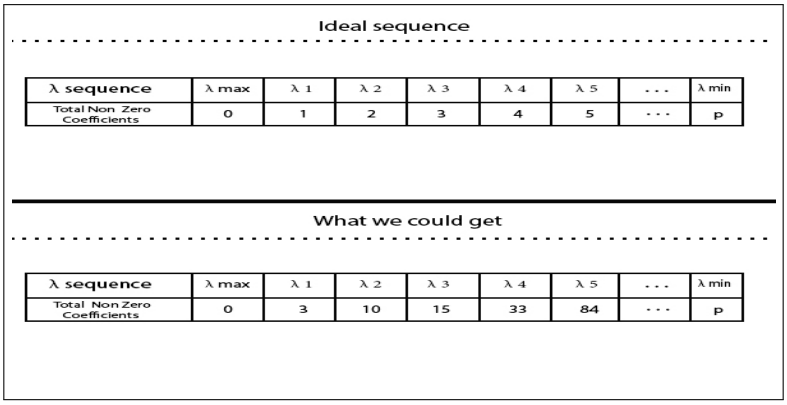

Figure 4.5: This figure is an example of the ideal λ sequence (top) and the λ sequence that we got from generating the λ sequences as discussed in the text (bottom). The ideal sequence sets one coefficient in the model, on each step, as we move from the λ_{max} to the λ_{min}. On the other hand, the sequence that we generated gives some gaps. For example, λ_{max} gives us 0 coefficients in the model (as expected) while λ_1 gives us 3 instead of 1. Since the λ sequence is in descenting order, there has to be two more λ values in the interval $(\lambda_{max}, \lambda_1)$ which will give us 1 and 2 coefficients in the model.

4.3.3.2 The Predictor-Corrector Algorithm

In contradiction with the other three methods previously discussed, the predictor corrector algorithm seemed to give the most appropriate results, both for the estimation of the parameters and for assessing their significance. However, some modifications of the algorithm had to be done for adapting it to our model. In this subsection, we briefly develop all the equations needed for implementing the predictor corrector algorithm.

The first step is to find the value of λ_{max}. Based in subsection 3.3.5, the maximum value of λ can be found by the soft threshold function. In our case, we have that $\lambda \geq |\sum_{i=1}^{N} x_{ij} w_i (z_i - \sum_{l \neq j} x_{il}\beta_l)|$. Therefore, if we set $\lambda_{max} = max_l(|\sum_{i=1}^{N} x_{ij} w_i (z_i - \sum_{l \neq j} x_{il}\beta_l)|)$ will all the β coefficients be equal to zero, and as we move from λ_{max} to λ_{min}, we would expect more and more coefficients to enter the model. If in the coordinate descent algorithm we give zero values for the initial coefficients, then for the first loop we will get: $p_i = 1/(1+e^{-0}) = 1/2$, $w_i = 1/2(1-1/2) = 1/4$ and $z_i = 0 + (1-1/2)/w_i = (1/2)/(1/4) = 2 \; \forall$ i. This leads to the maximum value of λ at:

$$\lambda_{max} = max_j(|\sum_{i=1}^{N} x_{ij} \frac{1}{4}(2-0)|) = max_j(|\frac{1}{2}\sum_{i=1}^{N} x_{ij}|)$$

Thus, finding the maximum value of λ is equivalent to finding the maximum absolute column sum of the data matrix and divide it by 2. By the time λ_{max} is computed, we can use the predictor-corrector algorithm for computing all the λ-knots, as discussed in subsection 3.4.1. The only difference with the predictor-corrector algorithm and our model is that the predictor-corrector algorithm defines λ_{max} as the maximum λ value for which only the intercept is in the active set

Ω. Our model, on the other hand, has no intercept term. Moreover, our λ_{max} is defined as the minimum λ value for which none of the coefficients is in the model and thus, Ω is empty. Note that, since λ_{max} is the minimum value for which Ω is empty, any value below it should let one or more coefficients to enter the model and give an increase in Ω. By testing this, we found out that even if we abstract *.Machine$double.eps* (the smallest possible value of a computer) from λ_{max} we get one coefficient in the model immediately. This lead to the following modification of the predictor-corrector algorithm:

> Since our model has no intercept and, since we need an initialization step for the predictor-corrector algorithm, it should be fair enough to treat the first entering coefficient as an "intercept", only for initializing the algorithm. Since we need a λ_{max} that gives an active set Ω which contains only one active coefficient, but we, on the other hand, have a λ_{max} which gives an empty set, we give this λ_{max} as an initialization to the algorithm, with an empty set Ω and a vector of zero coefficients. But before starting the estimation procedure, we immediately compute the first h step and thus, we jump to the first λ-knot and the Ω set gets one coefficient. Then, we restart the algorithm with the first λ-knot as λ_{max} and the coefficient who entered the model in the previous step takes the place of the "intercept" in the real predictor-corrector algorithm. Then the algorithm continues as usual.

The first coefficient who enters the model is not, of course, an intercept. The only reason for doing this small modification is for giving as input to the algorithm a λ_{max} for which the Ω set contains only one coefficient. If this hadn't be done and λ_{max} was the usual minimum value for which Ω is completely empty, then the algorithm would have stuck in the estimation procedure. However, this modification gives a "not so accurate" estimate for the first coefficient, but this is not a problem since the algorithm will run many circles after that and thus, the wrong estimation of the coefficient will be corrected (usually on the immediate next step). Furthermore, if the algorithm should start at a value smaller than λ_{max}, that is, somewhere in the sequence and not the beginning, then the same modification can be applied. The reason for not starting the algorithm at λ_{max} will be discussed later.

Next we develop the equations that were used in the algorithm, based on the equations in subsection 3.4.1 and on our model. A pseudo-code of the algorithm is also given. Finally, timing issues are discussed as well as a modification of the predictor-corrector algorithm. This modification is made mainly because of timing issues.

4.3.3.3 Equations for the Steps

Here we develop the equations that were used in the length step, the predictor step, the corrector step, as well as the active set step, that were discussed in subsection 3.4.1. Although the theory in subsection 3.4.1 is based on generalized linear models like ours, we used the Taylor approximation in the log-likelihood,

therefore the equations are slightly different. First of all, considering the objective function for our model, given by equation (4.7), we can find the corresponding H function given in equation (3.11), which is the following:

$$H(\beta,\lambda) = \frac{\partial R(\beta,\lambda)}{\partial \beta} = -\frac{\partial f(\beta,\lambda)}{\partial \beta} = -X'W(Z - X\beta) + \lambda sgn\begin{pmatrix}0\\ \beta\end{pmatrix} \tag{4.9}$$

Step Length

For the step-length part of the algorithm, we need the weighted correlations c. According to equation (3.12) and our model, the Taylor approximations of the weighted correlations are:

$$\hat{c}_j = X_j'W(Z - X_{-j}\beta_{-j}) \tag{4.10}$$

where X_j is the j-th column of the data matrix, Z and W are the working response vector given in equations (4.4) and the diagonal matrix of the weights given in equations (4.5), respectively. Moreover, X_{-j} is the data matrix with the j-th column removed and β_{-j} is the coefficients vector with the j-th entry removed. Note that the working response, the weights and the β coefficients are computed on the current λ-knot (λ_k when we are going to find the λ_{k+1}-knot).

As we can see from equation (4.10), the weighted correlations for our model are the A variable in the soft threshold given in equation (4.8). Which is actually reasonable, since our soft threshold in matrix form is $S(X_j'W(Z - X_{-j}\beta_{-j}),\lambda) = S(\hat{c}_j,\lambda) = S(A,\lambda)$ and, as have previously said, any coefficient with $\lambda \geq |A| \Rightarrow \lambda \geq |X_j'W(Z - X_{-j}\beta_{-j})| \Rightarrow \lambda \geq |\hat{c}_j|$ will be set to zero. In other words, it will belong to the Ω^c set[6]. This is something that joins together the theory in subsection 3.4.1.1 and the soft threshold function in equation (3.5).

For finding the h step which will give us the λ_{k+1}-knot, we need to compute the equations given by (3.14), that is:

$$c_j(h) = |\hat{c}_j - h\alpha_j| = \lambda_k - h$$

where, also for our model, $\alpha_j = X_j'\hat{W}X_\Omega(X_\Omega'\hat{W}X_\Omega)^{-1}sgn\begin{pmatrix}0\\ \hat{\beta}(\lambda_k)\end{pmatrix}$. Since the above equations can be written in the form given by equations (3.15), we can simply use those, that is:

$$h = min^+_{j\in\Omega^c}\left(\frac{\lambda_k - \hat{c}_j}{1-\alpha_j}, \frac{\lambda_k + \hat{c}_j}{1+\alpha_j}\right)$$

By the time we have found the λ_k-knot and its estimated $\hat{\beta}(\lambda_k)$ coefficients, we can compute the $\hat{c}$ vector and thus, find the h step and approximate the next λ_{k+1}-knot. When the λ_{k+1}-knot is approximated, one would expect that the coefficients

[6]Note that here we write $\lambda \geq |A|$ and not $\lambda = |A|$ as written in the theory part. This is because by simulation results we found out that the equality is not accurate. This probably happened because of numerical aspects which should be taken into account, when one applies any form of algorithms.

which give $|\hat{c}| \geq \lambda_{k+1}$ (the absolute correlations computed from the β on λ_k, against the new λ_{k+1}-knot) are going to be set in the active set Ω on the next round (after step 4: Active set)[7]. Therefore, $\hat{c}$ could be used as an indicator of what is probably going to happen next. If for example the coefficients that satisfy $|\hat{c}| \geq \lambda_{k+1}$ are exactly the same as those already in the active set, there is no need to use λ_{k+1} as a knot and estimate the parameters based on that, because probably we will result to the same active set and what we will have done would be to use unnecessary time. So if this was the case, we use the λ_{k+1} to re-estimate the h step on the same weights and coefficients and we repeat the procedure until the coefficients that satisfy $|\hat{c}| \geq \lambda_i$ (for $i > k+1$) have changed. Furthermore, corrector steps were used in all of those "internal" iterations for approximating better the current parameters on the current h step. Note however, that using the old absolute correlations $\hat{c}$ for checking if any extra coefficient is going to enter the model is not accurate, but rather an "approximation" that can save as time. This is because for actually checking which coefficient has an absolute value greater or equal to the new λ_{k+1}-knot, one has to compute the new absolute correlations $c_j(h) \forall j = 1, ..., p$ from the estimated coefficients on the new λ_{k+1}-knot, and must not use the old $\hat{c}$; This implies the estimation of the new β, which is what we try to avoid if necessary. However, we did this small modification only for saving time.

By the time we got a signal that the active set might change, we jumped to the third step, the corrector step, and not the second. According to Park and Hastie [22], predictor steps for small h changes are not crucial for big data problems. Therefore, when we were going to estimate the parameters on the next knot, we did not use a predictor step. On the other hand, as said on the previous paragraph, predictor steps were used on the internal iterations. This is because, according to Park and Hastie [22], when the differences between two knots are big, it should be wise to use a predictor step. By simulation results, we saw that when the internal loop was used, the changes between the knots were bigger than when we were going to test the active set on the steps three and four. Therefore, the predictor step was used in the internal loop.

The internal loop could sometimes be used and sometimes not. Whichever the case was, after the internal loop we move to step three, the corrector step, where we estimated the parameters based on the given λ-knot. If the resulting coefficients had given us the same active set Ω, it meant that nothing changed and that the h step was not big enough, therefore we returned to step one for re-computing the next step. If the resulting coefficients had given us one change in the active set (one coefficient had entered the model or had left it), then we moved to step four. Finally, if the resulting coefficients had given us a bigger increase in the active set than expected, it meant that the step was too big and we had to go back.

Park and Hastie [22] do not give any details of how the back step, an increase in the current λ, is done. However, based on the theory so far and the modified

[7] We actually know that the coefficients that will surely satisfy this inequality will allays be the ones that are already in the active set, and we need to reduce λ_{k+1} in such a way that at least one more from the elements in the $\hat{c}$ vector will satisfy the inequality. We know from the theory that the coefficients that are currently active will satisfy $|\hat{c}| \geq \lambda_k$, so they will also satisfy it for $\lambda_{k+1} < \lambda_k$. The question is which additional element of $\hat{c}$ will satisfy it.

algorithm, we used the following idea. Let λ_k and $\hat{\beta}(\lambda_k)$ be the previous knot and its estimates, respectively. Moreover, let λ_{k+1} and $\hat{\beta}(\lambda_{k+1})$ be the current knot and its estimates. If the current knot gives us a bigger increase (or decrease) in the active set than expected, we have to go back. Since we have the estimates for both knots, we can find the h_{back} step by using a corresponding procedure as we used for the h step (the forward one). Since the weighted correlations decrease as we go towards λ_{min}, they should increase as we go towards λ_{max}. That is, since we should have:

$$\hat{c}(\lambda_{k+1}) = \hat{c}(\lambda_k) - hX'\hat{W}X_\Omega(X'_\Omega\hat{W}X_\Omega)^{-1}sgn\begin{pmatrix}0\\ \hat{\beta}(\lambda_k)\end{pmatrix} \Rightarrow$$

$$\hat{c}(\lambda_k) = \hat{c}(\lambda_{k+1}) + hX'\hat{W}X_\Omega(X'_\Omega\hat{W}X_\Omega)^{-1}sgn\begin{pmatrix}0\\ \hat{\beta}(\lambda_k)\end{pmatrix} \Rightarrow$$

$$\hat{c}(\lambda_k) = \hat{c}(\lambda_{k+1}) + h_{back}X'\hat{W}X_\Omega(X'_\Omega\hat{W}X_\Omega)^{-1}sgn\begin{pmatrix}0\\ \hat{\beta}(\lambda_k)\end{pmatrix}$$

where $\hat{c}(\lambda_k)$ and $\hat{c}(\lambda_{k+1})$ are the weighted correlations on the corresponding λ-knots. Therefore, the h_{back} could be used as an approximation for the back step. This can be found by:

$$\hat{c}_j(\lambda_k) = |\hat{c}_j(\lambda_{k+1}) + h_{back}\alpha_j| = \lambda_{k+1} + h_{back} \Rightarrow$$
$$h_{back} = min^+\left(-\frac{\lambda_k + \hat{c}_j(\lambda_{k+1})}{1+\alpha_j}, \frac{\lambda_k - \hat{c}_j(\lambda_{k+1})}{\alpha_j - 1}\right)$$

where $\hat{c}_j(\lambda_{k+1})$ is, of course, calculated on the new knot, the new weights and working response from the $\hat{\beta}(\lambda_{k+1})$ estimates, but α is calculated on the previous knot. Thus, we can get an estimate of the h_{back} if needed. Then we add this estimate on the current λ_{k+1}-knot and we move again to step one. Note that, the minimum is taken among all positive values and that the h_{back} step does not need to be smaller than the previous h step. The reason for that is that the algorithm could have jumped many times for many h steps in a row, say h_1, h_2, h_3, and then realize that the last step was too big, but that we have to go in the middle of h_2 and h_3, so the h_{back} will be bigger than the last step, which is h_3 here.

Predictor Step

As has been already said, the predictor step was not often used in the algorithm. However, since it might have been used some times, we should provide the necessary equations. In subsection 3.4.1.2, we saw that the predictor β^{k+} can be found by the equation $\beta^{k+} = \hat{\beta}(\lambda_k) + (\lambda_{k+1} - \lambda_k)\frac{\partial\beta}{\partial\lambda}$, where $\frac{\partial\beta}{\partial\lambda}$ is found by solving $\frac{\partial H(\beta,\lambda)}{\partial\lambda} = \frac{\partial H}{\partial\lambda} + \frac{\partial H}{\partial\beta}\frac{\partial\beta}{\partial\lambda} = 0$. For our model, we have that for the j-th coefficient in the active set:

$$\frac{\partial H(\beta,\lambda)}{\partial\beta_j} = \sum_{i=1}^{N} w_i x_{ij}^2 + 0$$

which in matrix form is $\frac{\partial H(\beta,\lambda)}{\partial \beta} = X'_\Omega W X_\Omega$, and the weights are computed with respect to the coefficients from λ_k. Furthermore we have that:

$$\frac{\partial H(\beta,\lambda)}{\partial \lambda} = 0 + sgn(\lambda) = sgn\begin{pmatrix}0\\ \beta\end{pmatrix}$$

Therefore, one can compute:

$$\frac{\partial \beta}{\partial \lambda} = -\left(\frac{\partial H(\beta,\lambda)}{\partial \beta}\right)^{-1} \frac{H(\beta,\lambda)}{\partial \lambda} = -\left(X'_\Omega W X_\Omega\right)^{-1} sgn\begin{pmatrix}0\\ \beta\end{pmatrix}$$

and thus, the predictor step can be found by:

$$\beta^{k+} = \hat{\beta}(\lambda_k) + (\lambda_{k+1} + \lambda_k)\left(X'_\Omega W X_\Omega\right)^{-1} sgn\begin{pmatrix}0\\ \beta\end{pmatrix}$$

Corrector Step

After step one is done we can move to the corrector step. This step will use the coordinate descent algorithm discussed in subsection 3.2.1 for minimizing the convex objective function $R(\beta, \lambda_{k+1})$ with respect to the β coefficients. Here, according to Park and Hastie [22], one can either use β^{k+} if the predictor step has been done, or $\hat{\beta}(\lambda_k)$ if not, as warm starts for the coordinate descent algorithm (instead of initializing with a vector of zeros). This should lead to better estimates and it will also use less time. Note that Friedman et al. [11] also suggest a path-wise coordinate descent algorithm which uses warm starts of the previous estimates. After the corrector step is done, one should check if the active set should be updated or not. This is done in the next step.

Active set Step

This step should be done after each corrector step. According to Park and Hastie [22], the corrector and the active set step should run in a loop until the active set is no longer changing. After the corrector step, one should have the estimates $\hat{\beta}(\lambda_{k+1})$ on the λ_{k+1}-knot. We then simply calculate the weighted correlations $\hat{c}(\lambda_{k+1})$ based on the current estimates and we test which of the estimates in Ω^c give an absolute weighted correlation value greater than the current knot. Those coefficients are then added to the active set Ω. If the Ω is changed we run again a corrector step, if not, we first remove any zero coefficient that might have entered the active set and then, we move either to step one, if the active set has been increased by one or has remained the same, or we compute h_{back} if the active set has been increased by more than one. There is also an option on the algorithm suggested by Park and Hastie [22] were one can check if any of the current active coefficients reaches zero (goes into Ω^c) before another enters the active set. This however, was not seen in any of our simulations and therefore this step was skipped.

4.3.3.4 Time Issues and Bootstrap

For reducing the uncertainties of the lasso estimates, bootstrap was used. A total number of 1000 bootstrap samples of the same size as the total population were chosen. at each bootstrap, the optimal λ-knot had to be found via cross validation and the predictor-corrector algorithm. The time was a really big problem for both cross validation and the predictor-corrector algorithms. It would be completely time inefficient to first find all the λ-knots, estimate the coefficients, compute all the prediction errors for each knot and then find the optimal λ-knot which gives the minimum cross validation error. Even the use of parallel programming was not fast enough. Therefore, another algorithm was used, which combined cross validation, predictor-corrector and cyclic coordinate descent algorithm in a faster way, by skipping unnecessary λ values.

The algorithm is based on the whole theory discussed previously. It consists of the following four parts: (1) The first part creates a sequence of $m = 1000$ λ_j values, ranging from λ_{max} to $\lambda_{min} = \epsilon\lambda_{max}$, where $\lambda_j = \lambda_{max}\,(\lambda_{min}/\lambda_{max})^{j/m}$ for $j = 1, ..., m$ and $\epsilon = 0.00001$. (2) The second part computes the cross validation error for 20 of the λ values along the initial sequence. Those λ values have a distance of approximately 50 other values in-between them (it depends on the initial λ_{max}). That is, it computes $\lambda_1, \lambda_{50}, \lambda_{100}, ..., \lambda_{1000}$. The algorithm then finds the λ value which gives the minimum cross validation error among those 20 and it takes the right and left λ. If for example, the λ value which gives the minimum cross validation error is the 10-th, then the algorithm chooses the 9-th and 11-th. (3) The third step takes the right and left λ, which will have a distance of approximately 100 values in-between, and it again chooses 20 more values in that interval for computing their cross validation error. After that step, it finds again the λ which gives the minimum cross validation error and it takes the right and left values. (4) The fourth and final step of the algorithm, takes the right and left λ values found from the third step and it implements the predictor-corrector algorithm for finding the true optimal λ-knot in the interval between those two values. By this time, the same modification of the predictor-corrector algorithm was used, as discussed before. That is, since the λ value which will enter the predictor-corrector algorithm, do not correspond to λ_{max} (since it is skipped), before running the algorithm one have to compute the first h step.

A final modification of the algorithm was the early stops. For each step, we expect that as we compute the cross validation errors from a bigger value of λ to a smaller, the cross validation errors should become smaller and then finally start to increase again. By the time a cross validation error becomes greater than the previously computed one, the algorithm stops, because we know that the errors are going to increase after that step. Note that, the risk of getting stuck in a local minimum is not so big, because the λ values are not exactly next to each other;

There are many things that need to be commented here. First of all, in steps (2-3), the reason for choosing the right and left values of λ which gave us the minimum cross validation at the current step is because the initial λ sequence is not a λ-knots sequence but rather an approximate. As stated earlier, this kind of λ sequence gives us gaps. So by taking the right and left values we reduce the λ

sequence and at the same time we ensure that the true minimum lays somewhere in-between. Note that, percussions have to be taken if the λ value which gives the minimum cross validation at a given step is on the borders of the current λ sequence[8]. Furthermore, the way the reduction in the λ sequence is done, reduces the probability of getting stuck to a local minimum, although it does not eliminates it. By simulation results we did not stuck to any local minimum, but if the cross validation curve is not smooth, the probability becomes bigger.

This algorithm reduces the time it takes for finding the optimal λ-knot from seven hours to twenty minutes. However, twenty minutes for a bootstrap sample is still a long time when one needs to run 1000 bootstrap samples, but we had to be patient (it took ~ 7 days for each dataset). In figure 4.6, a visual example of how the algorithm works is given, and in algorithm 7 a pseudo-code of the algorithm is given.

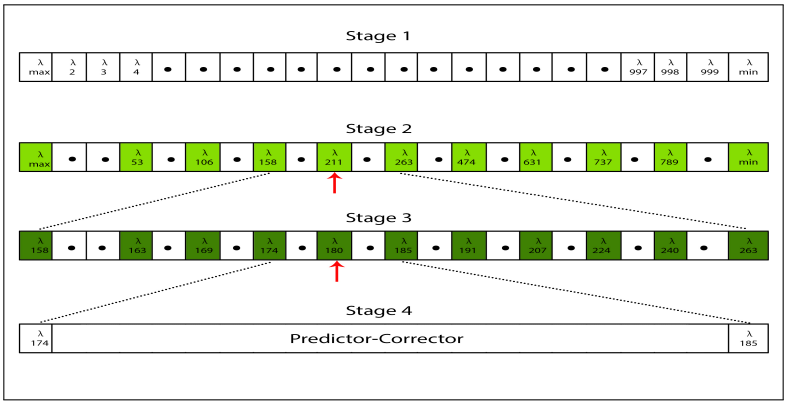

Figure 4.6: This figure is an example of how the modified bootstrap algorithm works in our problem. Each line corresponds to a stage of the algorithm. The coloured boxes are the values of λ for which the cross validation error is computed, the others are skipped. The red arrows correspond to the value of λ which gave the minimum cross validation error at the given stage. On stage 1 the sequence of the λ values is created, as described in the text. Those λ values are not actually knots, but it can be used as an initialization for finding the actual knots in a faster way. From those values, twenty values are chosen, with in-between distance ~ 53, and their cross validation error is computed on stage 2 (the light green boxes). Furthermore, the λ value which gives the minimum cross validation error on stage 2 is found and the two neighbour λ values are chosen to continue to stage 3. Note that, the two neighbour values are values for which the cross validation has been calculated, that is, they belong to the light green boxes and they are not the exact right and left neighbours. On stage 3 another reduction is taken place by the same way as on stage 2, but this time the interval between the λ values on the right and left corners, is smaller than before. Therefore, the algorithm aims with more precision this time. Finally, on stage 4 the right λ value (λ_{174}) is given as an initialization step in the predictor corrector algorithm, where the actual λ-knots are computed and the optimal λ-knot which gives the minimum cross validation error will be found. Note that since we choose neighbours of the cross validation minima, we ensure that the minimum cross validation error will be somewhere in the middle of the interval.

[8]If the minimum is on the last value, then take the last value and the previous one. If it is the first, then take the first and the second.

Data: Bootstrapped data matrix X.
Result: The estimated coefficients on the optimal λ-knot, as well as the cross validation error.
Initialization **(Stage 1)**: Create the λ sequence from λ_{max} to λ_{min};
Stage 2: Compute CV error for 20 of the values in the sequence, find minimum and take the right and left values;
Stage 3: Compute CV error for another 20 values in the interval between the right and left values chosen on stage 2. Find the one which gives the minimum error and take the right and left values again ;
Stage 4: Implement the predictor-corrector algorithm in the interval between the right and left values chosen on stage 3;

Algorithm 7: Modified Bootstrap Algorithm

4.3.4 AIC Criterion and the Estimates

After the bootstrap estimates had been generated, the AIC criterion was applied for finding the optimal frequency threshold. The reason for using the AIC criterion is that lasso tends to choose a slightly different set of coefficients to set equal to zero, at each bootstrap. By applying the AIC criterion we answer the question "What is the allowed number of times that a coefficient could have been set to zero, among the bootstrap samples". This was done by first counting the zero counts for each coefficient among the bootstraps. Those counts constructed a sequence of frequency thresholds that would create a various number of candidate models. Let us, for example, consider an arbitrary threshold somewhere in the threshold sequence, and say that it is 400. This means that some of the coefficients among the bootstraps, were set to zero 400 times. Using this threshold we find all the coefficients that were set to zero ≤ 400 times among the bootstraps. Those coefficients are allowed to enter the model by taking their mean among the bootstraps (for each coefficient), while the others that were set to zero > 400 times (that is, their frequency was above the current threshold), are simply set to zero. Then the AIC is computed for that model, where β is the vector of the current model coefficients and k is the number of non-zero coefficients in that vector. This is done for all the thresholds in the sequence and the one that minimizes equation (3.9) is chosen as optimal. When the optimal frequency threshold has been found, all the coefficients that were set different than zero less times than the threshold, are simply set to zero.

After that step, the final estimates were computed. The final estimates were computed by taking the mean among the bootstrap samples, for each coefficient. Of course, those who were above the optimal threshold were set to zero. Furthermore, the standard deviation for each coefficient was computed as proposed in subsection 3.2.2, although this was not the optimal way according to Lockhart et al. [15]. We shall revisit this issue afterwards. Finally, the non-zero estimated coefficients chosen by the bolasso were taken to the last part of the analysis, where their significance was tested.

4.3.5 Analysis of Interaction Effects

As has been previously mentioned, interaction effects were not tested. There were two reasons for that. The first and most important reason was time issues. Correcting and finalizing the algorithms for testing the main effects took really long time, because of the size of the datasets. Furthermore, running the algorithms for estimating the main effects for both datasets could take up to seven days. The second and final reason for not testing potential interactions was that the datasets were generated. Although weights had been placed in the NRD dataset for creating main effects, we could not be sure that interaction effects could have been occurred.

However, testing potential interactions can be done by exactly the same way as testing main effects for the lasso model. The only difference is how to define a window as exposed or unexposed and then treat those interacting drugs as one. The way by which we define a window as exposed or unexposed for the case of interactions is also described in figure 4.3.

4.3.6 Significance

In section 3.4 we referred to the paper by Lockhart et al. [15] about the significance testing of the lasso estimates. We implemented the covariance test statistic after we made some modification for adapting it to our problem. Furthermore, we implemented the covariance test statistic both on the bolasso estimates and on the lasso estimates (we revisit this afterwards). In this subsection we develop the covariance statistic of our generalized linear model and we discuss how the implementation was done.

4.3.6.1 Modified Covariance Statistic

When we tried to implement the covariance statistic (3.16), we noticed that the inner product of $I^{-1}S \times X\hat{\beta}(\lambda)$ (consider arbitrary λ) is not feasible; The reason for that is that for a generalized linear model, the information matrix is $I_{p\times p} = X'_{p\times N} W_{N\times N} X_{N\times p}$, and the score vector S is of dimensions $p \times 1$. Moreover, $X_{N\times p}\hat{\beta}_{p\times 1}(\lambda)$ is of dimensions $N \times 1$ [24]. Since $I^{-1}S$ is of dimensions $p \times 1$ while $X_{N\times p}\hat{\beta}_{p\times 1}(\lambda)$ is of dimensions $N \times 1$, the inner product is not feasible. We consider this as a writing error of the paper.

Assuming that by $z = \eta + I^{-1}S$, Lockhart et al. [15] actually mean the equation of the working response, we develop the covariate statistic for our model. The working response for our model was given in equation (4.4). This equation can be written in matrix form as:

$$z = \eta + W(Y - \mu) \tag{4.11}$$

where $\eta = X\beta$, Y is the response vector, μ is the vector of the fitted values and W is the diagonal matrix of the weights, which its (i, i) element is $d\eta_i/d\mu_i$. Furthermore, $W(Y - \mu)$ is of dimensions $1 \times N$ which makes the inner product with $X\hat{\beta}(\lambda)$ feasible. Therefore, the covariance statistic becomes:

$$T_k = \frac{\langle W(Y-\mu), X\hat{\beta}(\lambda_{k+1})\rangle - \langle W(Y-\mu), X_\Omega \tilde{\beta}_\Omega(\lambda_{k+1})\rangle}{2} \tag{4.12}$$

where $W(Y-\mu)$ are computed with respect to the active set Ω. We used this statistic to assess the lasso significance.

4.3.6.2 Border Specifications

Lockhart et al. [15] do not specify the covariance statistic for the first and last coefficient to enter the model, at least for the GLM case. We however, will test the first and last coefficients based on results from the whole theory so far and on what seems like a reasonable modification.

For testing the first coefficient who enters the model, one needs the Ω set at the previous value of λ_1. Since the Ω set is the active set of coefficients generated from λ_{k-1}, then for λ_1 (which gives the first coefficient to enter the model) the active set right before this value will be empty. That is, $\Omega = \emptyset$. Therefore, the covariate statistic becomes:

$$T_1 = \frac{\langle W(Y-\mu), X\hat{\beta}(\lambda_2)\rangle - 0}{2} \tag{4.13}$$

where X is a one column matrix corresponding to the $\Omega \cup \{\text{first to enter}\}$, and the weights are computed accordingly.

For testing the last coefficient that enters the model, we are missing an extra λ-knot. Since the covariate statistic uses the next λ_{k+1} for computing T_k, then for testing the last coefficient, which corresponds to λ_p (where p is the total number of coefficients), we need an extra λ value. By that time, however, we have that the active set Ω has length $p-1$ and the active set $\Omega \cup \{\text{last to enter}\}$ has length p. Since the final λ_p-knot is bounded by $\lambda_{min} = \epsilon\lambda_{max} > 0$, we can use any $\lambda < \epsilon\lambda_{max}$ as the next knot for computing T_p. Note that, for $\lambda = 0$ there is no penalty at the lasso and, therefore, the corresponding estimates are the usual maximum likelihood estimates. The covariance statistic for the last coefficient becomes:

$$T_p = \frac{\langle W(Y-\mu), X\hat{\beta}(\lambda)\rangle - \langle W(Y-\mu), X_\Omega \tilde{\beta}_\Omega(\lambda)\rangle}{2} \tag{4.14}$$

where λ is chosen by us, such that $0 \leq \lambda < \lambda_{min}$.

4.3.6.3 Implementation of the Covariance Statistic

We implemented the covariance statistic in two ways. The first way was to implement the statistic on the output of the bolasso algorithm. That is, on the coefficients that the bolasso estimated as non-zeros (after implementing AIC etc.). The second way was to use the initial datasets and run the significance test on it, that is, simple lasso on the initial data matrices. For both methods, the λ-knots sequence was computed by the predictor-corrector algorithm.

The reason for implementing the significance with two different ways, was for investigating how the covariance statistics works, and for investigating if the bolasso

and the simple lasso will give the same significant coefficients. On the one hand, the covariance statistics is a new method and the resulting p-values are not easy to interpret. On the other hand, according to Lockhart et al. [15], the usual p-values and confidence intervals for the lasso do not exist, therefore any confidence interval obtained by the bolasso would not be accurate. This is actually reasonable if we consider the way the intervals from a bootstrap are obtained. That is, by computing the standard deviation with respect to the bootstrapped estimates for each estimate (or by taking the 5-th and 95-th sorted value of the bootstrapped estimate for a 95% confidence interval). This is not so accurate for the lasso, because at each bootstrap sample the algorithm will choose the optimal λ-knot, via cross validation, and thus, the resulting λ-knot will not be the same at each bootstrap. Therefore, applying a different penalty to the same estimate throughout the bootstrap process will not result to an accurate confidence interval. Furthermore, if we do compute confidence intervals from the bootstraps, the usual standard deviation method for doing this, could be worse than just taking the 5-th and 95-th values (in a sorted sequence). Since lasso is a greedy way of estimating parameters, any normality assumptions needed for computing the standard deviation might have been violated.

We focus more on the output of the covariance test, rather than the actual interpretation of the p-values. For the bolasso estimates, we expect that all the estimates which are chosen not to be zero, will get a rather low p-value from the covariance test. Meaning that they are significant and correctly chosen by the bolasso. For the estimates from the complete dataset we expect that the covariance statistic will choose the same coefficients to be significant, as those chosen by the bolasso to be non-zero.

Chapter 5

Results of the Analysis

In this chapter we present the results of the analysis. The main analysis was based on the NRD dataset, but the results from the RD dataset are also given for comparison. The estimation methods and the significance assessing were the same for both datasets.

Both datasets were treated with two different ways. The first way was the bolasso estimating method. For this method, bootstrap was used for estimating the coefficients of the model. For each bootstrap, the optimal λ-knot was found via cross validation and the modified predictor corrector algorithm that we discussed in subsection 4.3.3.2. After the bootstrap, the optimal threshold was found using the Akaike's information criterion. The estimates and their confidence intervals were computed by the bootstrap samples. Furthermore, the estimates that were chosen as non-zero from the bolasso were used for significance testing.

The other method concerns a simple lasso application. For this method, the complete matrices from both datasets were used without any bootstrap process. Then the complete predictor corrector algorithm was used for finding the λ-knots. Furthermore, the significance of the coefficients was assessed by the covariance statistics, using all the coefficients this time.

This chapter begins with chapter 5.1 where the results from the bolasso are given. It then continues to chapter 5.2 where the results from the covariance statistic are presented. Chapter 5.3 discusses the results from the estimated log risk ratios. Furthermore chapter 5.4, gives the results from the replication of the bolasso for the NRD dataset. The asymptotic distribution of the covariance statistic is concerned in chapter 5.5. Finally, chapter 5.6 discusses the general results of the analysis.

5.1 Bolasso Results

In this section a presentation of the bolasso results is given. The results from both datasets are presented and commented. This section focuses on the comparison of the results between the two different datasets, as well as the investigation of characteristics of the bolasso method that significantly differ between the datasets.

5.1.1 Cross Validation and Lasso Paths

This section gives an illustration of the λ-knots lasso paths, as well as the cross validation error, for one bootstrap sample. One bootstrap sample was taken from the NRD dataset and one from the RD dataset. The entire λ-knots paths were computed for both datasets, using the predictor-corrector algorithm and warm starts when estimating the coefficients. Furthermore, the entire cross validation error curve was computed for both datasets. For reasons described in subsection 4.3.3.4, a modified version of the algorithms was used and therefore, we place the results of the optimal λ-knot chosen by our algorithm with the results of the complete predictor corrector algorithm together for comparison.

In figure 5.1 we see the complete lasso paths for the datasets. The two plots on the top corners correspond to the NRD dataset, while the two plots on the bottom corners correspond to the RD dataset (the left figures are the full scales, while the right are zoomed scales). The coloured lines in the plots correspond to the "ongoing" estimated coefficients, while the x-axis consists of the true λ-knots approximated by the predictor-corrector algorithm. For both datasets, we can see that for each λ-knot, we get a new coefficient in the model, as we move from λ_{max} to λ_{min}, although it is difficult to see this at the end.

For both NRD and RD datasets, it is clearly visible that we have gotten a rather "strange" first coefficient at the beginning. This is because, as said before, we do not have an intercept in the model and we therefore implemented the modified version of the algorithm. But this is not a problem since the coefficient clearly gains its true value later. Furthermore, the black vertical lines in each plot, correspond to the value of λ-knot which gives the minimum cross validation error, that is, the optimal λ-knot. The estimates at this λ-knot are those that would have been chosen for that bootstrap sample. Finally, we see that for both plots a big amount of coefficients tend to enter the model at the end. However, this should not be confused with the way that the lasso model works. Although more coefficients will enter the model as we move from λ_{max} to λ_{min}, which is what we expect from the lasso design, the way that the coefficients here enter so rapidly at the end is only because of the structure of the datasets.

By comparing the lasso plots for the two datasets we see some significant differences. Two things have to be commented here. The first, and most important one, is that for the NRD plots we see a motif in the way that the coloured lines of the estimates behave. Here, we are not interested in where the coefficients go (positive or negative direction), but rather on the fact that for the NRD dataset they tend to follow a motif to one direction, while for the RD dataset the lines are more uniformly placed. By that, it seems that the lasso has captured the differences

between the two datasets. And while in the NRD dataset some coefficients have a different behaviour than the rest, in the *RD* dataset they all seem to have a rather random behaviour. The second thing is the place where the black line is placed. For the NRD dataset, the line is placed more far to the right side, than for the *RD* dataset. This means that the lasso will set more estimates in the model for the NRD dataset and less for the *RD* dataset. Which is completely reasonable since the *RD* is random, while NRD is not. However, the total number of coefficients that are being set in the model is not common for each bootstrap. We shall discuss this later on.

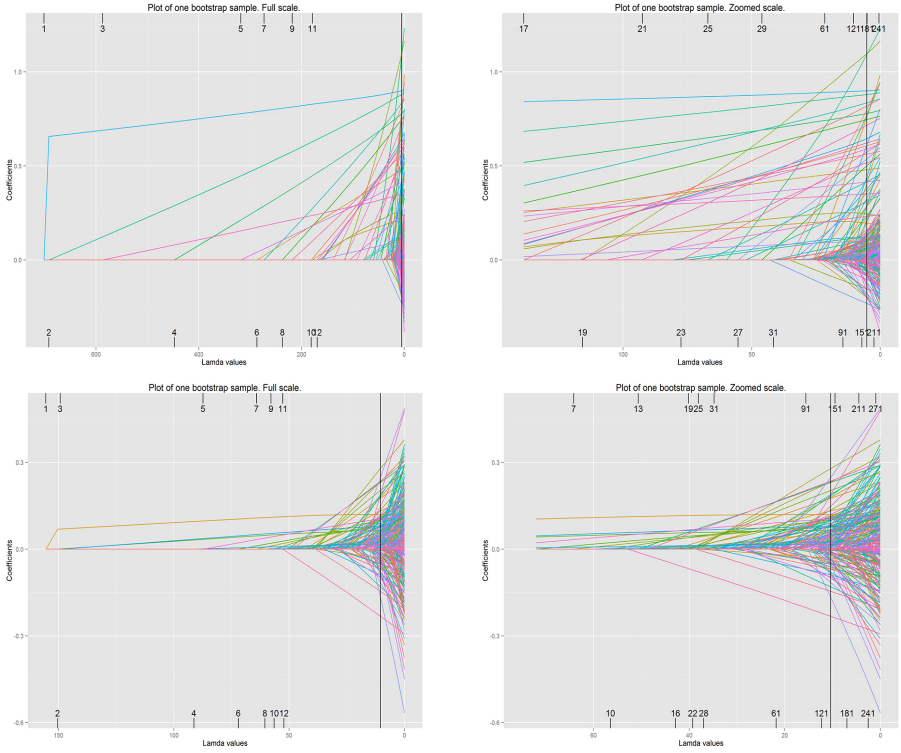

Figure 5.1: Those four plots are an illustration of how the lasso paths look like. The top plots are from the NRD dataset, while the bottom plots are from the RD dataset. The left plots are in full scale, while the right plots are on zoomed scale. Each coloured line in the plot corresponds to one coefficient. The y-axis is the value of the estimated coefficient and the x-axis consists of the λ-knots, computed by the predictor-corrector algorithm and placed in descenting order. The numbers on the top and bottom of the plots correspond to the total number of active coefficients on the specific λ-knot (they are given only for a few knots, not for all). It is obvious that the number increases as we go from λ_{max} to λ_{min}. Finally, the vertical black line in each plot corresponds to the optimal λ-knot which gives the minimum cross validation error. Comments on the plots can be found in the text.

In figure 5.2 we see an illustration of the cross validation error curves for the two datasets, computed on the same bootstrap samples as the ones in figure 5.1. First of all, we have to comment of the differences between the algorithm which computes the whole paths and the modified one. In all plots, the black curve corresponds

to the actual cross validation curve and the blue vertical line corresponds to the optimal λ-knot which gives the minimum error. As said before, computing the whole sequence via the predictor corrector algorithm is not time efficient.

Our algorithm is the blue dotted line. This is the path that our algorithm would have chosen and it clearly skips a lot of λ values. The dots correspond to each stage as they are coloured in figure 4.6. We can see that the algorithm skips a big portion of the λ sequence but it efficiently approximates the true optimal by taking into account more values as it gets closer to it. The vertical red line corresponds to the optimal value of λ that our algorithm would have chosen. Although the algorithm is based on a rather "fake" initial sequence, the final stage which is the predictor-corrector stage will compute the exact λ-knots.

The blue and the red lines do not overlap. This does not mean that our algorithm doesn't choose the same value as the initial algorithm. The reason is that the algorithms had to run separately and, since the predictor-corrector algorithm numerically approximates the λ-knots, the smallest modification could give slightly different results. The true predictor-corrector algorithm starts from the first λ-knot in the sequence of λ (the first to the left), while our modified version starts the predictor-corrector algorithm somewhere in the neighbourhood of the true minimum cross validation error (the dark-red dots). Because of the fact that the predictor-corrector algorithm uses warm starts for finding the next knot and estimating its coefficients, the initial λ value given in the algorithm would affect the estimates. This is the reason that the two vertical lines do not overlap. However, this effect is negligible and the total number of active coefficients is the same for both vertical lines. If the curves were to be computed at the same time and under the exact same conditions, the vertical lines would surely overlap. However, in the worst case scenario, where they don't overlap, they are very close to each other so we would not have any problem.

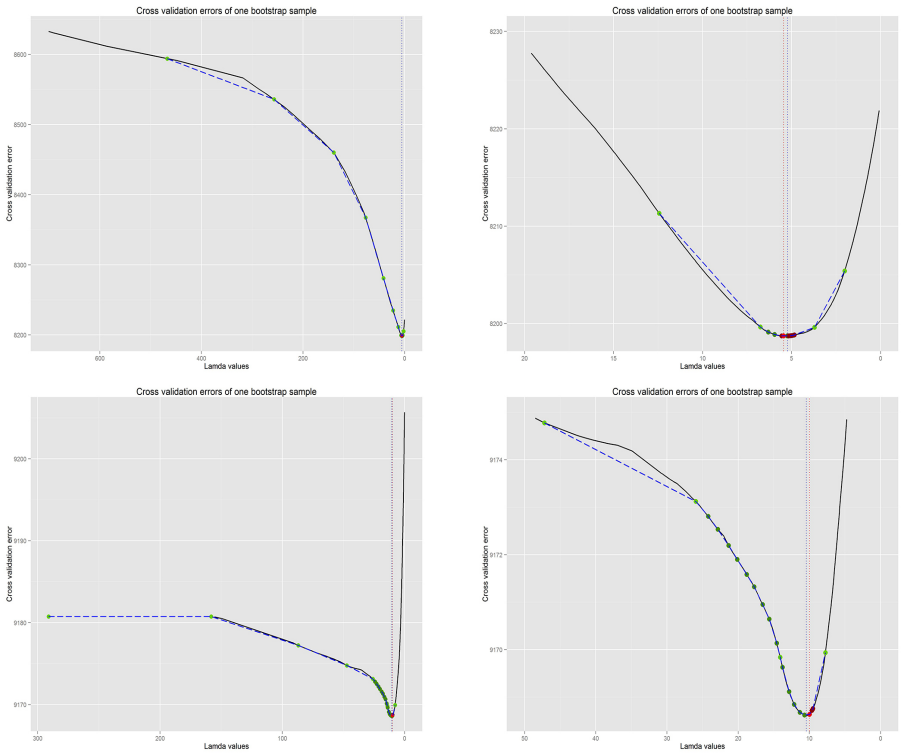

Figure 5.2: This figure is an illustration of the complete cross validation curve for the two datasets, as well as the the "path" that our algorithm follows for finding the optimal λ-knot. Those plots correspond to the same bootstrap samples that were used for plotting the lasso paths in the previous figure. The top corners of the figure correspond to the NRD dataset, while the bottom corners correspond to the RD dataset. Again, the left side plots are the full scales and the right side plots are the zoomed scales. The black line for both plots corresponds to the actual cross validation curve, while the blue dotted line corresponds to the path that our algorithm follows. Sometimes the lines overlap, sometimes not. The light green dots correspond to the second stage of the algorithm, the dark green dots correspond to the third, the dark red dots corresponds to the forth stage (the predictor-corrector stage) and the light red dot correspond to the optimal λ. Furthermore, the blue vertical line is where the true optimal is, while the red vertical line is the optimal λ that our algorithm would have chosen. More details can be found in the text.

By comparing the cross validation plots between the two datasets we see some differences. First of all, we see that the range of the cross validation errors for the RD dataset is much higher (the x-axis ranges from ~ 9160 to ~ 9210), than that for the NRD dataset (the x-axis ranges from ~ 8200 to ~ 8650). In general, the values of the cross validation errors depend on the dataset itself, therefore the comparison between the datasets would not be so accurate. Here however, we will give a rough comparison only for the matter of investigating the differences. The cross validation error is a measure of how well the model predicts for the specific value of λ-knot. For that reason we see that the NRD dataset is being better predicted from the lasso method, than the RD dataset, since it gives lower cross validation values. Secondly, we see that the optimal λ-knot from the NRD dataset is closer to zero, than the optimal λ-knot from the RD dataset. This means that

the total number of active coefficients for that bootstrap sample will be higher for the NRD dataset, than for the RD dataset. Which is something that coincides with the lasso plots commented previously. Note, however, that the λ values of the x-axis are also different between the datasets, but one cannot compare them. This is because λ_{max} depends on the maximum absolute column sum of the current matrix and therefore, it will clearly be different not only between NRD and RD, but also among the bootstrap samples of the same dataset.

Finally, we see that the cross validation curves for both datasets are smooth. One can hardly find a local minimum and therefore the concept of "divide and conquer" that our algorithm implements, works quite well. This however, will not necessarily be the case for other datasets. Other datasets might not have a smooth cross validation curve, but they could rather be full of local minima. In those cases, the version of the predictor-corrector algorithm that we implemented will probably stuck to a local minimum and is therefore not recommended.

5.1.2 Bolasso Behaviour on the Datasets

In this section we shall discuss the behaviour of the bolasso method on the two datasets. Since the one dataset is completely random, while the other one is not, we expect to see some differences on some of the bolasso characteristics. We chose to compare three characteristics of the bolasso procedure. The first one was the total number of non-zero coefficients which were chosen at each bootstrap. That is, the ones that were in the Ω active set for each bootstrap sample. The second one was the optimal λ-knots which were chosen at each bootstrap. Finally, the third one was the cross validation error on those optimal λ-knots. That is, the minimum cross validation error for each bootstrap.

Consider the left sub-figure in figure 5.3. This figure shows the total number of active coefficients included in the Ω set, for each bootstrap sample. The red points are from the NRD dataset, while the black are from the RD dataset. Clearly, the difference between the datasets is big. We can see that for the NRD dataset, the total number of active coefficients chosen in each bootstrap does not differ among the bootstraps, in a such big scale as the total number of active coefficients from the RD dataset. In other words, the red points tend to be more gathered and thus, giving smaller deviation than the black ones, which are more spread, resulting to bigger deviation of the total number of active coefficients among the bootstraps. This simply means that for the NRD dataset, the bolasso method seems to be more "determined" on the total number of coefficients that it will choose to enter the model, than for the RD dataset where the bolasso method cannot decide the total number of coefficients that should actually enter the model.

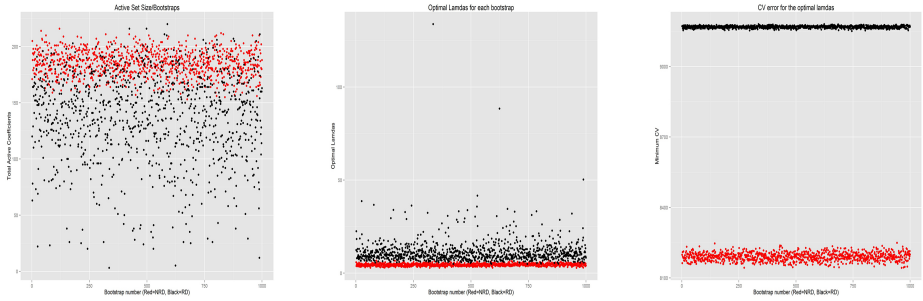

Figure 5.3: The three figures give an illustration of the different behaviour that the bolasso method had on the two datasets. For all three sub-figures, the x-axis is the number of the bootstrap ranging from 1 to 1000. Furthermore, the red points correspond to the NRD dataset and the black points correspond to the RD dataset. Three bolasso characteristics are shown in this figure. The left sub-figure shows the total number of non-zero coefficients chosen for each bootstrap. In other words, those numbers are the size of the active set Ω for each bootstrap. The middle sub-figure shows the optimal λ-knot chosen for each bootstrap. Finally, the right sub-figure shows the cross validation error of the chosen optimal λ-knot for each bootstrap.

A similar behaviour can be seen in the middle sub-figure of figure 5.3. In this figure we can see the values of the optimal λ-knots for each bootstrap. The red points correspond to the NRD dataset, while the black points correspond to the RD dataset. As we can see, the deviation of the black points is bigger than that of the red ones. This means again that the bolasso method acts more determined for the NRD dataset than for the RD dataset. As said previously, the values of the λ-knots are not be compared between the datasets nor between the bootstraps, because they depend on the columns of each bootstrap sample. However, here we are more interested in their behaviour rather their values. Note also, that the values of the λ-knots are connected with the first sub-figure for the total number of active coefficients. The higher the λ value in the middle figure, the lower the value in the left figure, since for high values of λ less coefficients enter the model.

Consider the right sub-figure of the figure 5.3. This figure shows the cross validation errors for the optimal λ-knots chosen at each bootstrap. That is, the λ-knots in the middle sub-figure. Again, the red points are from the NRD dataset, while the black points are from the RD dataset. Clearly the cross validation errors from the RD dataset are much higher than those from the NRD dataset. Something that coincides with figure 5.2. From that, it seems that the models chosen from the bolasso, are being better fitted for the NRD than for the RD dataset. However, as said before, since the cross validation error depends on the dataset, any direct comparison between the two datasets would not be accurate. We would rather consider the characteristic of the thickness of the values among the bootstraps. For the cross validation errors from the NRD dataset, there is no much to say. The fact that the red line, constructed by the points, is thicker than that for the RD dataset, does not give us any important information. Except from the fact that for different values of λ we get different values of cross validation error, which is something natural and expected. The important information from this figure is given from the black points of the RD dataset. It seems that even if the bolasso chooses optimal λ-knots in a big range, all of those knots give almost

the same high cross validation error. That is, all the optimal λ-knots are equally bad options. Bolasso, however, does the best it can for choosing the right λ-knot. The problem is only because of the RD dataset.

In conclusion, it seems that the bolasso gives important information for the structure of the datasets. Bolasso has captured the differences of the two datasets. Moreover, it seems to have captured the non-randomness of the NRD dataset and therefore, it acts more determined on the choice of the active coefficients that should enter the model. On the other hand, bolasso cannot determine the optimal λ-knots nor the total number of active coefficients for the RD dataset. The randomness of this dataset is easily seen from the left sub-figure. Because bolasso cannot easily determine the correct number of active coefficients for the RD dataset, the estimates from this dataset might not be trustworthy; Finally, it should be noted that we do not expect that the bolasso will choose the same number of active coefficients at each bootstrap, as has been previously said, lasso tends to forget some coefficients or take into account irrelevant ones and this is the actual reason for running bootstrap. However, the deviation of those numbers discussed in figure 5.3 is a way of understanding the bolasso behaviour.

5.1.3 Estimation of Log Risk Ratio

This section considers the results from the bolasso method. The estimation of the log risk ratios for the coefficients is given, as well as the AIC curves which chose the optimal threshold. Furthermore, the lasso paths for the non-zero coefficients are also presented, which correspond to a "reduction" of figure 5.1.

In figure 5.4 one can see the AIC curves for each dataset. The right sub-figure corresponds to the NRD dataset and the left corresponds to the RD dataset. The vertical red lines on both figures give the minimum Akaike's information criterion, which corresponds to the optimal threshold. From the figures one can see that the optimal threshold chosen for the NRD dataset is bigger (~ 920), than that of the RD dataset (~ 800). This means that the allowed number of times that each coefficient could have been set to zero among the bootstraps, is higher for the NRD dataset than that for the RD dataset. Remember that the coefficients which had been set to zero more times than the threshold will be simply set to zero for all bootstraps, while for the others that are below the threshold, their mean will be taken as an estimate. Since the threshold for the NRD dataset is bigger than that for the RD, the coefficients of the NRD are not so "restricted" as those from the RD. That is, they are allowed to have been set to zero more times than those from the RD dataset. Therefore, more coefficients from the RD will be set exactly to zero. Which means that for the model of the NRD dataset more coefficients are needed, than for the RD dataset.

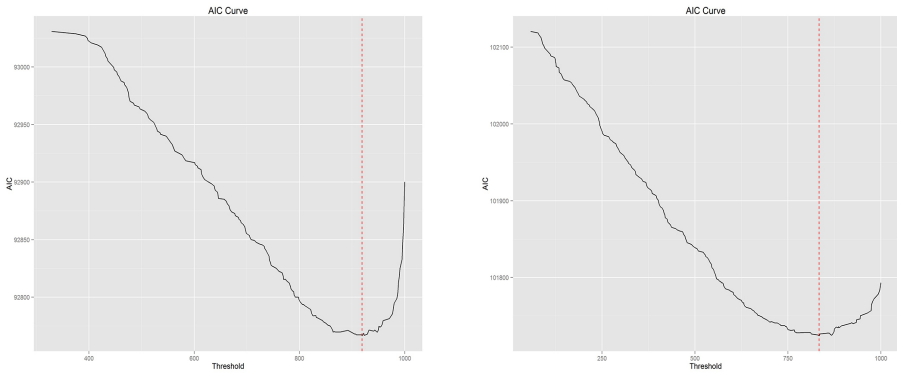

Figure 5.4: This figure shows the AIC curve for the two datasets (NRD left, RD right), which was used for finding the optimal threshold. The y-axis is the AIC criterion and the x-axis is the thresholds. By thresholds we mean the maximum total number that each coefficient could have been set to zero under the bolasso procedure. Finally, the red line corresponds to the minimum AIC, which corresponds to the optimal threshold.

In figure 5.5 one can see the estimated coefficients for each dataset. The left sub-figure is from the NRD dataset, while the right is from the RD dataset. The red dots are the estimates from the bolasso. Those are the coefficients that have successfully passed the Akaike's information criterion, and are estimated by taking their mean among all bootstrap samples. The red line on the plots is a line constructed by red dots. Those dots correspond to the coefficients that have not passed the Akaike's information criterion and thus, were set to zero. It is clear that the bolasso method set some coefficients exactly to zero and removes them immediately from the model.

The black lines which correspond to each estimate are their confidence interval. Those are computed on a 0.05 level of significance, by taking the 5-th and 95-th value of their sorted values. Obviously, some intervals include the zero value. Someone would reasonably wonder why those coefficients have not been set to zero by the bolasso. Well first of all, remember that according to Lockhart et al. [15], there is no inference for the lasso estimates. Therefore, those intervals are not accurate. They should not be interpreted as the usual confidence intervals because each value in the interval is computed by a different λ-knot (at each bootstrap). Thus a different penalty is applied to the same estimate. This procedure is greedy and leads to non-accurate inference. Secondly, the Akaike's information will allow some of the coefficients to take the zero value at least once, and this is what we see here. However, the reason for including the confidence intervals here, is only for showing the values that each coefficients took among the bootstraps.

Considering the same figure of the estimates, we see some differences between the datasets. First of all, the total number of estimates that are not exactly zero is higher for the NRD dataset, than for the RD dataset. This is something that coincides with the AIC plots commented before. It is also reasonable because the NRD is not random. However, since the RD dataset is indeed random, why does it have some coefficients that are not exactly zero? This might have happened

because even if the RD dataset is randomly generated, non-randomness could have been occurred. However, the confidence intervals for the RD estimates are much bigger than those for the NRD dataset. Which means that the estimates are not so accurate. Note that, this coincides with the results from the previous subsection, where we saw that the bolasso characteristics on the RD dataset varied in big scale among the bootstraps. That is, if the value of the optimal λ-knot is very different among the bootstraps, it will result to big variances for the estimates. Finally, most of those intervals include the zero value. One the other hand, the majority of the intervals for the NRD dataset do not include the zero at all.

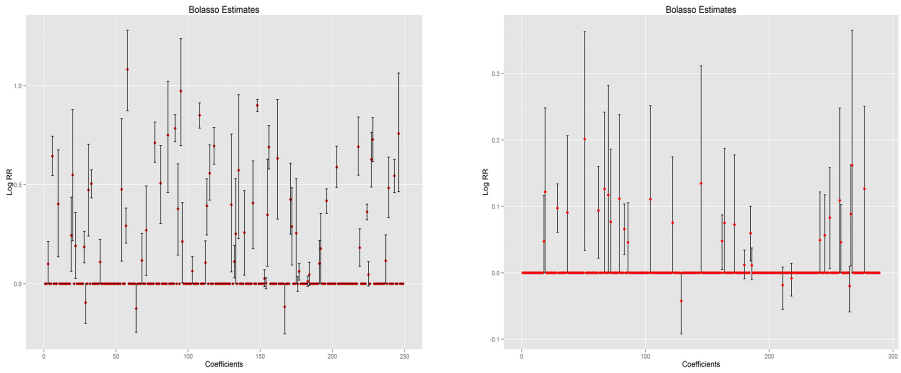

Figure 5.5: This figure shows the bolasso estimates of the log risk ratios for the two datasets (NRD left, RD right). The red points are the estimated coefficients from the bolasso, computed by taking the mean among the bootstrap samples. The black lines correspond to 95% confidence intervals of each estimate, which were obtained from the bolasso. That is, they are not optimal.

After that the estimates were computed, the non-zero estimates were only taken into account. With respect to those estimates, the lasso paths were again computed with the usual way of the predictor-corrector algorithm and the λ-knots. Those paths are given in figure 5.6. The top plot is for the NRD dataset, while the bottom plot is from the RD dataset. One can see that the total number of lines for the NRD plot is higher than that for the RD. This were also seen from the figures of the estimates. The remarkable thing here is that the two plots seem to have an approximately same structure! This is another success of the bolasso. Remember that even if NRD is not random while RD is, their background frequencies are indeed the same. Meaning that the datasets were generated by the same initial frequencies, as discussed in subsection 1.3.2. This is what we see here. Bolasso is an estimation method and not an inference method. One should expect that even if we see the same structure here, caused by the initial frequencies, the total number of significant estimates should be less for the RD dataset. In the next section we shall seek the significance of those estimates.

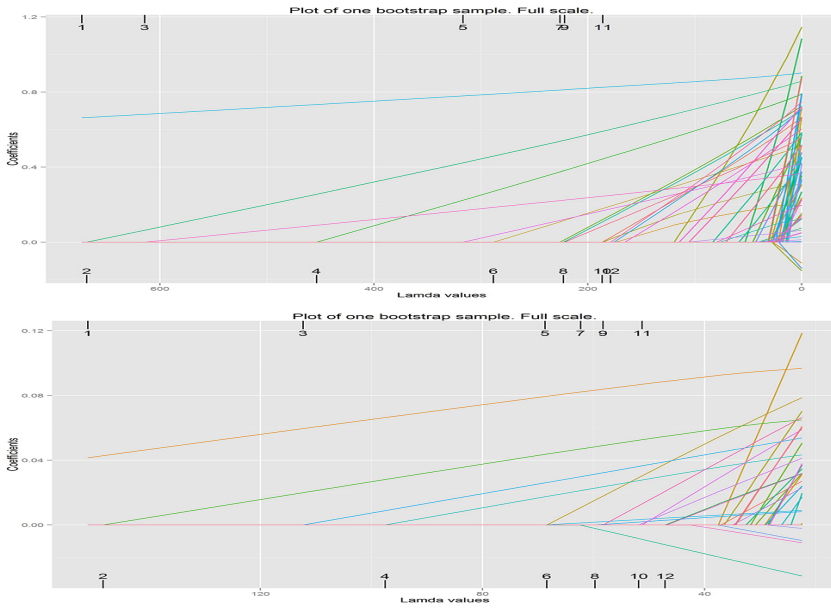

Figure 5.6: This figure can be considered as a reduced version of figure 5.1. It corresponds to the lasso paths of the coefficients chosen as non-zero from the bolasso method. The top figure corresponds to the NRD dataset while the bottom figure corresponds to the RD dataset. Each coloured line in the plot corresponds to one coefficient. The y-axis is the value of the estimated coefficient and the x-axis consists of the λ-knots, computed by the predictor-corrector algorithm and placed in descenting order. The numbers on the top and bottom of the plots correspond to the total number of active coefficients on the specific λ-knot (they are given only for a few knots, not for all).

5.2 Significance Testing

This section concerns the implementation of the covariance test statistics. The test was implemented both on the bolasso estimates and on the simple lasso estimates from the initial data matrices for both datasets. In subsections 5.2.1 and 5.2.2 we discuss and compare the results from the two datasets for their bolasso and simple lasso estimates, respectively. In the last subsection 5.2.3 we compare the results from the two methods for the same dataset. The last subsection is the subsection were the results of the significant coefficients are presented.

5.2.1 Significance of the Bolasso Estimates

The test was first implemented on the estimates from the bolasso method discussed in section 5.1. Those estimates that were assigned a value different than zero from the bolasso were the ones that their significance was assessed by the covariance test. Therefore for this step, the data matrices for each dataset included only the columns which corresponded to the non-zero estimates from the bolasso. Before applying the covariance test, the true λ-knots had to be computed again for those estimates, with respect to the new data matrices. This was again done by the predictor-

corrector algorithm. This time however, we did not need to run bootstrap and the data matrices were of smaller size. Therefore, the complete predictor-corrector algorithm was used and not the modified one.

The p-values from the covariance test are shown in figure 5.7. Considering a 0.05 significance level, the red line is placed on that value for the y-axis. Moreover, the order by which the p-values are given (according to x-axis), corresponds to the order by which the coefficients enter the model. That is, the first value is for the first coefficient which enters the model, etc. According to Lockhart et al. [15], the coefficients that have a p-value bigger than the given significance level, are considered non-significant given the others in the model. As we previously stated, there is a big ongoing discussion around the interpretation of those p-values. We however, shall consider them under the usual interpretation, by now.

Those coefficients that have a p-value smaller than 0.05 are considered to be significant. That is, they have to be in the model, while the others don't. Obviously, the total number of significant coefficients for the NRD dataset is higher than that for the RD dataset, which is completely reasonable. Finally, the reason that the RD dataset has two significant coefficients can be connected with possible non-randomness that were accidentally generated. This however, can not be investigated any further. The true outcome from the figure is that the NRD dataset has more significant coefficients and bolasso has indeed found them.

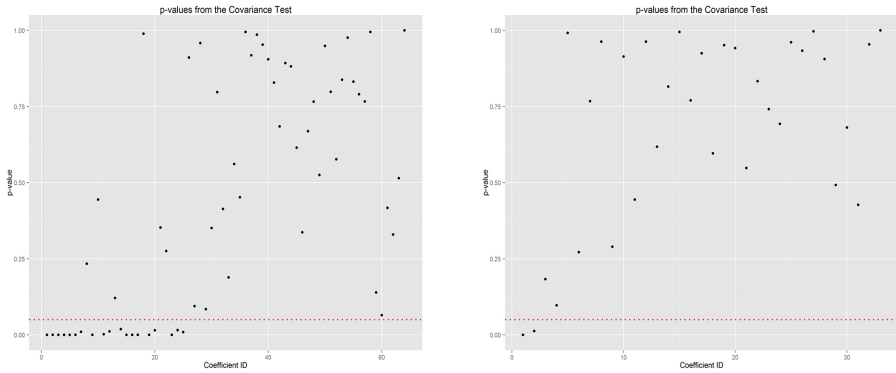

Figure 5.7: This figure shows the p-values from the covariance test, for the bolasso estimates. The right figure is from the NRD dataset, while the left is from the RD dataset. The red line in both figures is placed on $y = 0.05$, assuming an $\alpha = 0.05$ significance level. For the NRD dataset, the total number of coefficients chosen as non-zero from the bolasso is 64, while from those, 19 are considered significant from the covariance test. For the RD dataset, the total number of coefficients chosen as non-zero from the bolasso is 33, while only 2 are considered significant from the covariance test.

Taking into account only the significant coefficients, the lasso paths were again computed. In figure 5.8 we see the lasso paths from the significant coefficients of the bolasso. The left one is from the NRD and the right is from the RD dataset. Clearly, we now see some differences with the figure 5.6 and the background frequencies which were common to each dataset have gone. This was expected. Even if the background frequencies were the same for the two datasets, the coefficients from NRD had weights for causing the MI event and thus, they are significant.

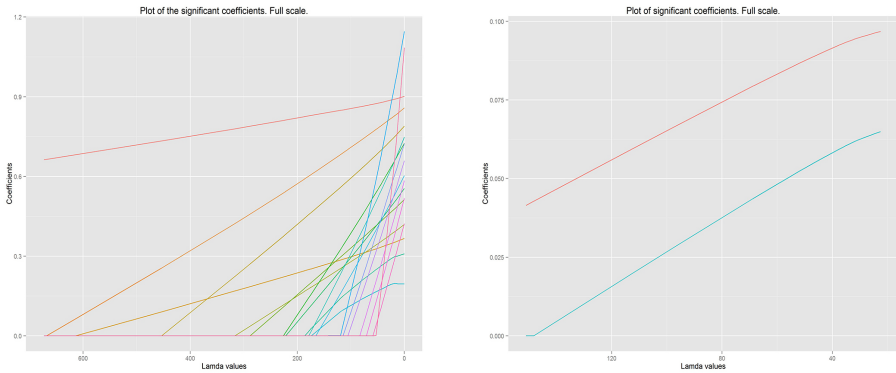

Figure 5.8: This figure gives a further reduction of the figures 5.1 and 5.6, where only the significant coefficients are taken into account. Again, the left figure is from the NRD dataset while the right is from the RD. Each coloured line in the plot corresponds to one coefficient. The y-axis is the value of the estimated coefficient and the x-axis consists of the λ-knots, computed by the predictor-corrector algorithm and placed in descenting order.

Figure 5.9 shows only the significant estimates of the bolasso method. The noteworthy here, is that none of the leftover coefficients has a confidence interval which includes the zero value. That is, all the estimates from the bolasso, which had taken the zero value at least once among the bootstraps, are not considered as significant at all. This applies to both datasets. Note however, that as said in section 4.1, 100 drugs were given weights for the NRD dataset. Here, not only the significant ones but also the ones from the bolasso output, are clearly less than 100. This has probably happened because of the reduction step in figure 4.2. On that step, we couldn't ensure that some of the 100 drugs will not leave the NRD dataset.

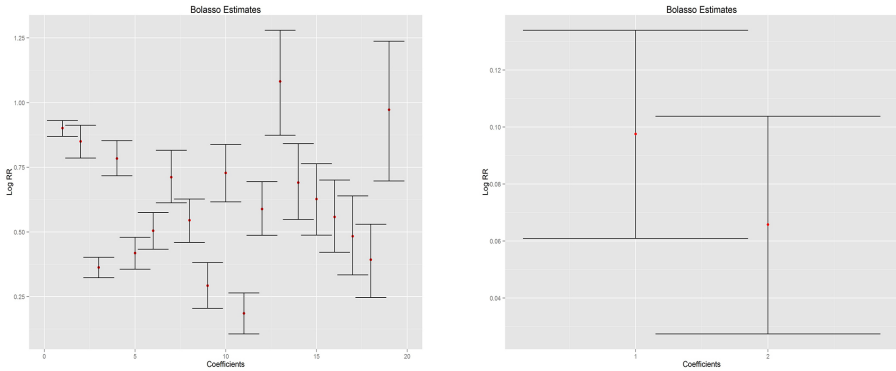

Figure 5.9: This figure is a reduced version of figure 5.5, where the non-significant estimates have been removed. The left figure corresponds to the NRD dataset, while the right figure corresponds to the RD dataset. The red points are the estimated coefficients from the bolasso, computed by taking the mean among the bootstrap samples. The black lines correspond to 95% confidence intervals of each estimate, which were obtained from the bolasso.

So far, it seems that the covariance test works quite well. Any confusion that could have been caused by the intervals which included the zero value, is now gone. Furthermore, it is quite remarkable the fact that not all of the bolasso estimates are significant. Since the lasso method, in general, sets some coefficients exactly to zero, one should expect that using bolasso, all the non-significant estimates will be set to zero. This however, seems not to be the case.

Finally, the significant estimates were chosen for each datasets and were re-estimated. The re-estimation was done using cyclic coordinate descent on the matrices which contained only the columns of the significant coefficients. Furthermore, the estimation was done using $\lambda = 0$ this time, which corresponds to the usual maximum likelihood [15]. Those estimates are the final estimates for the two datasets from the bolasso method and will be presented in section 5.2.3.

5.2.2 Significance of the Lasso Estimates

In this section we present the results from the covariance test, when applied to the complete data matrices without using bootstrap. That is, the lasso method was applied only once for each dataset and then the significance of the estimates was assessed by the covariance test. For that, we neither used bootstrap for estimating the coefficients, nor cross validation for finding the optimal λ-knot. The whole data matrices, for NRD and RD, were used in the predictor corrector algorithm for finding a sequence of true λ-knots, and the estimation of the coefficients was done on those knots by the cyclic coordinate descent algorithm. Based on the λ-knots sequence, the covariance statistics was implemented.

In figure 5.10 we can see the p-values calculated from the covariance test. We can see the same behaviour as before. The total number of significant coefficients for the NRD dataset is again higher than that for the RD dataset. Here however, the total number of p-values is greater than before because we used all the coefficients for the covariance test. Moreover, the p-values for the NRD dataset are not increasing so rapidly as they do for the other dataset. It seems that the covariance test tends to reject the coefficients of RD faster than those from the other dataset.

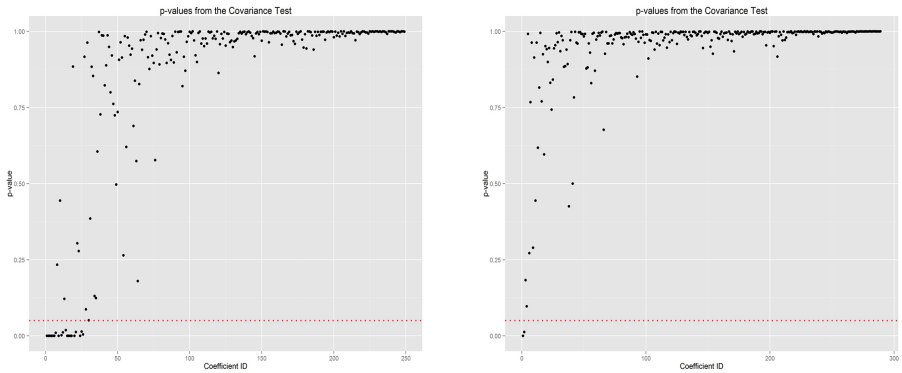

Figure 5.10: P-values from the covariance test, for the lasso estimates of the initial matrices. The right figure is from the NRD dataset, while the left is from the RD dataset. The red line in both figures is placed on $y = 0.05$, assuming an $\alpha = 0.05$ significance level. Finally, the points are the p-values. From the total number of 249 coefficients from the NRD dataset, 20 are considered significant from the covariance test. From the total number of 289 coefficients from the RD dataset, 2 are considered significant from the covariance test.

For a constant value of $\lambda = 0$, bootstrap with 1000 bootstrap samples was implemented for both datasets, only with the significant estimates in the model. Thus, confidence intervals could be again obtained. In figure 5.11 one can see the resulting estimates. Clearly none of them has a confidence interval which includes zero. Furthermore, in figure 5.12 one can see the lasso paths computed again by the usual way of the predictor-corrector algorithm. Their behaviour is the same as for the bolasso plots, as expected. Finally, those estimates will be considered as the final estimates from the lasso and will be presented in the next section. There, a comparison between bolasso and lasso will be given.

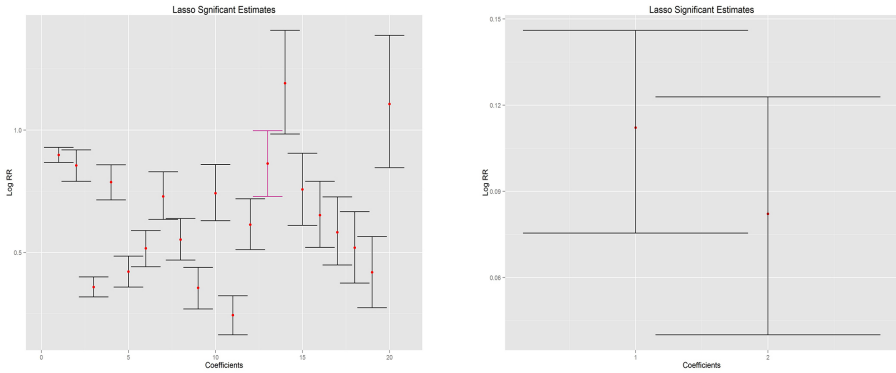

Figure 5.11: This figure shows the estimated coefficients that were considered significant, for the two datasets (NRD left, RD right). The red points are the estimated coefficients and the black lines are their standard deviation. Both the coefficients and their standard deviation were obtained using bootstrap of 1000 samples only on the coefficients considered significant by the covariance test. For the bootstraps, the estimation was done using no penalty, that is $\lambda = 0$. Note that, for the right figure, the lines are big because of scaling. Finally, the purple confidence interval in the left figure, will be discussed in the next subsection.

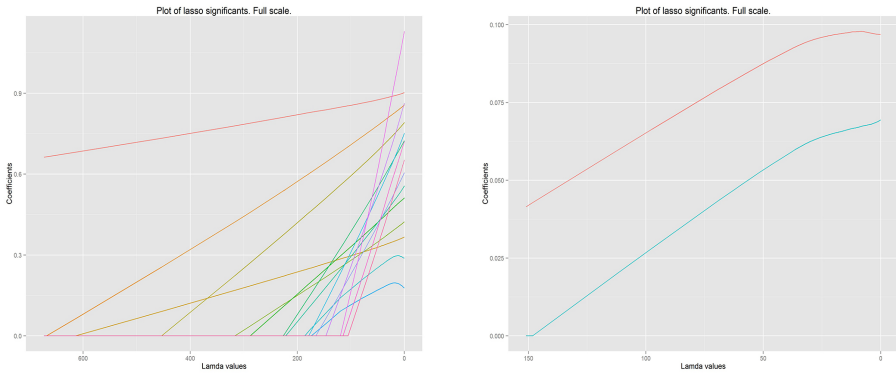

Figure 5.12: This figure shows the lasso paths for the two datasets (left NRD, right RD), for the estimates that were considered significant. The x-axis corresponds to the λ-knots. Each coloured line in the plot corresponds to one coefficient. The y-axis is the value of the estimated coefficient and the x-axis consists of the λ-knots, computed by the predictor-corrector algorithm and placed in descenting order.

5.2.3 Comparison of the Two Methods

This subsection concerns the comparison of the bolasso and the simple lasso methods for the two datasets, after the significance testing. Therefore, we consider only the figures given in subsections 5.2.2 and 5.2.1. This section is connected with appendix C, where the tables of the estimates are given. We shall consider the results from one dataset at a time and will compare the potential differences between the two methods that were previously implemented. Note that, the actual comparison here is between the results of the bolasso and simple lasso, after the covariance test. In other words, this section will help us understand the covariance test by comparing its different results from the same dataset.

Results from RD

Consider first the right sub-figure of figure 5.7, which corresponds to the p-values of the RD dataset from the bolasso method. This looks fairly the same as the right sub-figure of figure 5.10 for the p-values from the simple lasso method. Moreover, the same similarity can be seen in the right sub-figure of figures 5.8 and 5.12, and 5.9 and 5.11. It therefore seems like bolasso and simple lasso provided the same results after the significance test.

Consider table C.2 of the estimates given in appendix C. This table shows the significant estimates from the bolasso and the lasso method, as well as their names, their confidence intervals (CI) and their p-values. The simple lasso method gives bigger estimates and bigger confidence intervals, however the differences are not large. Furthermore, we see that the p-values from the covariance test are exactly the same for the two methods;

It seems like the simple lasso method is enough for this dataset. Taking into account the fact that the covariance test gave exactly the same results for the two methods, and that the bolasso confidence intervals are not the optimal ones,

the simple lasso method combined with the covariance test, should be the optimal method to use.

This is also a big computational advantage. The bolasso method took almost seven days to run, while the simple lasso took only three hours. Since the results are the same after the implementation of the covariance test, and the bolasso confidence intervals are not trustworthy, the simple lasso method seems to be enough if one wishes to assess the significance of the estimates via the covariance test. The current dataset, however, is completely random. Therefore, one should not trust it without taking the results from the other dataset into account.

Results from *NRD*

The differences are quite significant, considering the results from the bolasso and simple lasso for the *NRD* dataset, after implementing the covariance test. Although the left sub-figure of figures 5.7 and 5.10, 5.8 and 5.12, and 5.9 and 5.11 look fairly the same, they are not;

By comparing the left sub-figure of figures[1] 5.9 and 5.11 we can see that the simple lasso method has inserted one extra coefficient in the model (the purple one), while the bolasso has not. That is, the covariance test chose one extra coefficient as significant for the simple lasso. This can be reasonable since the simple lasso method tends to set some irrelevant coefficients in the model, and this is the reason for running the bolasso method: for excluding the irrelevant coefficients as much as possible.

Consider table C.1 in appendix C. This is the table of the estimates and all their information. We can see that all the results, except for the last line, are almost the same (sometimes exactly the same) from the two methods. According to those results, it seems again that for testing the significance and estimating the coefficients, the simple lasso is indeed enough. Therefore, we could skip running bolasso which takes too long time. Here however, the simple lasso gave us one extra coefficient, namely "S03CA04". Therefore, even if the estimates, the confidence intervals and the p-values are the same for both methods, the fact that one extra coefficient is in the model makes it difficult to choose the best method. This is because we don't actually know which method was right. That is, we don't know if this extra coefficient should be in the model or not.

There is one bigger problem here. If we assume that the simple lasso method is wrong and that it accidentally chose one extra coefficient, we should naturally expect that this extra coefficient has a rather high p-value. However, this is not the case. The p-value of this coefficient is not only low, but it is almost zero. This indicates that the coefficient is very important for the model; For that, we did a check for seeing where the bolasso "missed" this coefficient. The results were surprising. The bolasso method excluded this coefficient at the Akaike's information criterion. This coefficient did not pass this criterion and therefore its significance hadn't been tested at all. Akaike's information criterion is a form of model significance. Therefore, it is very strange that this coefficient is not

[1] We shall compare only those two figures, because in the figures for the p-values and the lasso paths is not easy to see the differences.

significant at all according to bolasso, but it is one of the most significant ones according to lasso.

This was an important difference between the methods. For investigating this further, we run the bolasso again. This time we used 1500 bootstrap samples. The results of the replicated bootstrap should give an insight of which method is the best one. Note that the simple lasso method does not need to be replicated. This method is not based on simulations of datasets like the bolasso, therefore its results would be the same. In section 5.4 we present the results of the replication and we comment on them.

5.3 Interpretation of the Results

This section concerns the interpretation of the estimated coefficients, by the way they should be interpreted under a real analysis. Only the results from the bolasso for the NRD dataset are considered here, given in table C.1 in appendix C.

For our likelihood under the case-crossover design, the estimates are the log risk ratio of the coefficients [14]. Let $log(\frac{\beta}{1-\beta})$ be the output of a coefficient from the bolasso method, then $e^{log(\frac{\beta}{1-\beta})} = \frac{\beta}{1-\beta}$ is the estimated risk ratio for that coefficient. By exponentiating the confidence interval we can also find the confidence interval for the risk ratio of a coefficient.

We are more interested in the the risk ratios rather than the values of the true coefficients. The risk ratio of each coefficient shows the risk of taking the corresponding drug. If the risk ratio is greater than one, then the drug has an influence on the MI event. If the risk ratio is less than one then the drug has prevented the MI even. However this cannot happen in our case-crossover design. As said in subsection 2.1.1, if we think that a trigger might have prevented the event, we treat the interruption of that drug as a trigger in the model. Something that we did not do for any of the drugs here. Finally if the risk ratio is one, no information is obtained about the effect of the drug.

Table C.3 in appendix C shows the transformed version of the estimates. That is, the risk ratios and their confidence intervals, as well as the initial frequencies of those drugs. As we can see, all the drugs have a risk ratio greater than one, with drug "C10AA05" (red) giving the highest and "B01AC06" (blue) giving the lowest. This means that all those drugs have an effect of causing MI. Moreover, "C10AA05" and "B01AC06" have the biggest and smallest effect, respectively, of causing MI among the patients who took them.

By checking again the corresponding p-values and the drug frequencies we see that the results are contradicted. The p-values should be associated with the risk ratios and the frequencies. High risk ratio in combination with high frequency should result to a lower p-value, than a drug having low risk ratio and low frequency. However, by checking the drugs "R01AD05" (green) and "C10AA05" (red), we see that this seems not to be the case. "R01AD05" has lower p-value than "C10AA05", even if its risk ratio and frequency are lower than those for "C10AA05". Considering also the problems discussed in subsection 5.2.3, it seems like the covariance test

is not preferable, but rather misleading; For drugs with low risk ratio and high frequency (or the other way around), like the drug "B01AC06", we cannot decide if the resulting p-values seem to be right.

In general, for investigating which drug is the most risky one, does not necessarily need any significance testing. However, bolasso can be used without the significance testing of the covariance test. Having that in mind, we decided to check the coefficients from the bolasso that were not considered as significant from the covariance test. Ten of those estimates with the highest risk ratios are given in table C.4.

The three first coefficients in table C.4 have risk ratio higher than two. This value might be considered significant, meaning that those drugs do indeed affect the MI event. However, their p-values assigned by the covariance test are big enough for rejecting them as significant. Taking also this result into account it seems that the covariance test has a weakness in identifying the true significant coefficients. We shall discuss this further after the replication.

5.4 Replication for NRD

This section considers the replication of the bolasso for the NRD dataset. A total number of 1500 bootstrap samples was run this time. Akaike's information criterion was again implemented for finding the optimal threshold. Finally, the covariance test was implemented on the non-zero bootstrapped estimates. Just like before.

Figure 5.13 shows the results from the replication. The top left figure is the Akaike's information criterion which gives an optimal threshold of $\sim$ 1377 (remember that we have 1500 bootstraps this time). The right top figure shows the bolasso estimated coefficients as well as their confidence intervals. Moreover, the left bottom figure shows the p-values for the bolasso estimates, computed again by the covariance test. Finally, the right bottom figure shows only the significant estimates of the bolasso. All figures are the same as the ones for the bolasso with 1000 bootstrap samples, except from the right top figure of the estimates.

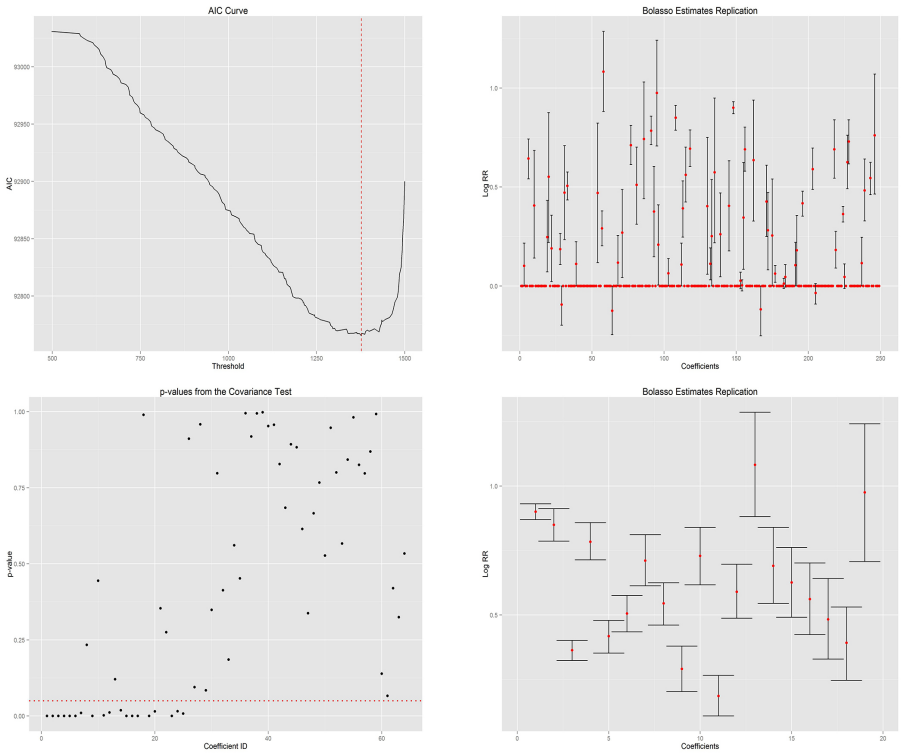

Figure 5.13: This figure shows the outputs from the replication of the NRD dataset. The figure on the top left corner is the Akaike's information criterion for the bootstrap samples. The figure on the top right corner is the estimates of the bolasso. The left bottom figure is the p-values of the bolasso estimates and finally, the right bottom figure shows the significant estimates for the bolasso after the covariance test was applied.

Consider first the differences between the estimates of NRD for the 1000 and 1500 bootstrap samples. The top right figure of figure 5.13 (the replicated estimates) and the left figure of figure 5.5 (the initial estimates), are not exactly the same. Because it is difficult to observe the differences from the plots, and because it is tiring to compare 64 estimated values and drug IDs, we plotted the estimates in figure 5.14. In that figure, the black dots correspond to the estimates that were chosen as non-zero both from the 1000 and the 1500 bootstrap samples of the NRD. Each black dot is actually double (one dot for the initial estimates of the bolasso and one for the replicated bolasso). Clearly the estimates of those dots are overlapping (with extremely small deviation some times). Moreover, the red triangle corresponds to an estimate from the replicated bootstrap, while the green triangle corresponds to an estimate from the initial bootstrap. Although the triangles are close to each other, they belong to different drugs. That is, the initial bolasso and the replicated one did not choose exactly the same coefficients to be in the model. This means, that the uncertainties of the lasso method are not completely eliminated by running bootstrap. However, both of those estimates are

very close to zero, which means that their risk ratios will be almost one and thus, non-informative. Finally, it is remarkable the fact that the significant coefficients, chosen by the covariance test, are exactly the same for both the initial bolasso and the replicated one. This can be seen be the right bottom figure of figure 5.13 and the left figure of figure 5.9, as well as the table which we discuss in the next paragraph.

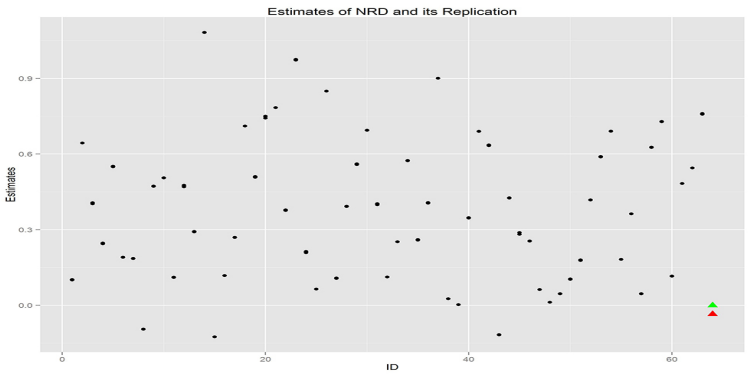

Figure 5.14: This figure shows the estimated coefficients that the initial bolasso and its replication chose to enter the model. The y-axis contains the values of the estimates and the x-axis corresponds to drug IDs. The black dots are estimates that were in common for both procedures. Furthermore, the red triangle corresponds to an estimate that the replicated bolasso chose to enter the mode, while the initial bolasso did not. Finally, the green triangle corresponds to an estimate that the initial bolasso chose to enter the model, while the replicated bolasso did not.

Consider now only the results from the replicated bolasso, and the p-values from the simple lasso. Table C.5 in appendix C shows the replicated bolasso estimates that were considered significant by the covariance test, placed again side by side with the simple lasso estimates. As we can see, we got exactly the same results as before. The replicated bolasso did not choose the extra coefficient to be in the model. The rest of the results are the same as those in table C.1.

The last thing that one could do is to investigate the behaviour of the ”S03CA04” coefficient, with respect to the others. Therefore, we computed the correlations of that coefficient against all the others. This should explain if the coefficient belongs to a correlated group, or if it is truly irrelevant. Figure 5.15 shows those correlations. The red point on the top right corner is the correlation of ”S03CA04” against itself, which is of course one. The rest of the points are the correlations of ”S03CA04” against the other coefficients. Clearly, all those correlations lay around zero. None of those correlations is significant enough, for explaining the strange behaviour of ”S03CA04”. Thus, this coefficient is indeed irrelevant and does not belong to a group of correlated coefficients. This proves that the bolasso method has correctly excluded this coefficient from the model. On the other hand, the reason that the covariance test chooses the coefficient, will remain a mystery.

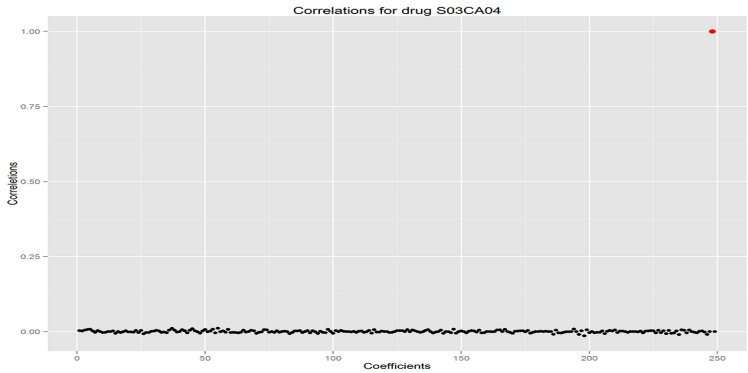

Figure 5.15: Correlations plot for the "S03CA04" drug vs the rest of the drugs. The red point on the top right corner is the correlation of the drug with itself. The black points are the correlations with the other drugs.

For the covariance statistic we were more interested in the total number of significant coefficients. That is, we would like to investigate if we could get the same significant coefficients when using simple lasso combined with the covariance statistic and when using bolasso. The fact that the estimated coefficients, shown in the tables, are almost the same, is only because on every method we re-estimated them without penalty. Finally, the results from this section question the validity of the covariance test, but also show the importance of the replication procedure. The replication did not give exactly the same results as the the initial bolasso. However, the difference is negligible.

5.5 The Asymptotic Distribution of T_k

This section considers the asymptotic distribution of the covariance statistics T_k. According to Lockhart et al. [15], T_k follows asymptotically the $Exp(1)$ distribution under the null hypothesis. Lockhart et al. [15] however, have not proven this assumption for the case of generalized linear models, but they support this conclusion on simulation results. We however, did not see any straightforward results in our analysis.

For some values of T_k the quantile-quantile plots were created. In appendix D one can see four figures with the Q-Q plots. Figure D.1 shows the Q-Q plots from the bolasso estimates for the NRD dataset, figure D.2 shows the Q-Q plots from the bolasso estimates for the RD dataset, figure D.3 shows the Q-Q plots from the simple lasso estimates for the NRD dataset and figure D.4 shows the Q-Q plots from the simple lasso estimates for the RD dataset. The top row in each figure shows the Q-Q plots for some of the covariance statistics that resulted to a p-value< 0.05 for the corresponding coefficient. That is the null hypothesis was rejected for them and thus, T_k should not asymptotically follow the $Exp(1)$ distribution. Moreover for each figure, the bottom row contains the Q-Q plots

for some of the covariance statistics which resulted to a p-value> 0.05 therefore, the null hypothesis could not be rejected and thus, their asymptotic distribution should be $Exp(1)$. The way by which the plots were created was by simulating 100 values from the $Exp(T_k)$ distribution (each plot has a different T_k), and then plot the quantiles against the theoretical quantiles of the $Exp(1)$ distribution.

For all plots the results are similar, so we comment the results in general. The distribution of T_k seems in many cases to be similar to $Exp(1)$, since the points are on the line, but one cannot tell if the results are from the null hypothesis or not. Most of the plots seem to have an $Exp(1)$ distribution, irrespective of whether the null hypothesis is rejected (top plots), or not (bottom plots). For the top plots of each figure, the points should not be on the line, but as we can see, this is not always the case. The points of some plots show that the corresponding statistic gives a good fit. On the other hand, the points for the bottom plots should always be on the line, or at least most of them. But this is again not the case for all plots. Some plots show a rather bad fit and thus they do not follow the $Exp(1)$ distribution, even if they should.

The conclusion from the Q-Q plots is that the distribution of T_k is indeed $Exp(1)$ most of the time, but the result seems to hold for both the null and the alternative hypothesis. In the paper of Lockhart et al. [15], they also apply the covariance statistic to the case of the elastic net[2] and they state that for that model, the covariance statistic under the null hypothesis, multiplied by a factor, follows by distribution the $Exp(1)$ distribution. They specifically state that for an orthogonal data matrix [15]:

$$(1+\gamma)\, T_k \xrightarrow{d} Exp\,(1)$$

As we can see for the case of the elastic net, the covariance statistic multiplied by a factor which contains the penalty, follows an $Exp(1)$ distribution under the null. A such modification maybe also should have been applied to the case of generalized linear models, and one maybe should not use just the usual $T_K \to Exp\,(1)$. If this should be the case, then the results from the covariance statistic might have been different.

[2]The Elastic net can be considered as another form of the lasso. The only difference is that the Elastic net, except from the λ penalty, penalizes the likelihood also by $\frac{\gamma}{2}||\beta||_2^2$. There $\gamma \geq 0$ is a constant [15].

5.6 Conclusion of the Analysis

The current analysis was an implementation of bolasso and simple lasso methods as well as the newly proposed covariance test statistic. According to the theory discussed in this thesis, simple lasso tends to forget some coefficients, which in our case were the log risk ratios, or take into account some irrelevant ones. Moreover, a reduction of those uncertainties can be done via bolasso in combination with the Akaike's information criterion. The reason for running both simple lasso and bolasso was mainly for applying the covariance test on both methods and then compare the results. As a bi-product of the analysis, replication need to be run for the NRD dataset for confirming the results of the initial bolasso. In this section we shall discuss the main founding of the analysis.

Bolasso seems to be an efficient estimation and variable selection method with only small uncertainties. First of all, we saw in subsection 5.1.2, that bolasso has the ability of capturing the characteristics of a dataset. Since each bootstrap sample corresponds to a single lasso, bolasso does indeed reduce the uncertainties that the simple lasso could have caused. This can be seen by the fact that if we take two bootstrap samples into account, we should have gotten a different set of coefficients. Bolasso on the other hand "averages" the active sets and with the help of Akaike's information criterion, effectively chooses the correct coefficients to be in the model, taking always into account all the other observations from the bootstrap samples. Although the replication did not give exactly the same results as the initial bolasso, most of the coefficients were the same. Only two coefficients were not in common from the two procedures but the differences were not significant since their log risk ratios were almost zero. This shows that bolasso does not completely eliminate the uncertainties of the lasso but it significantly reduces them.

Bolasso has two negative aspects. First of all, it is extremely time consuming. Although bolasso has indeed proven its efficiency in reducing the set of coefficients, it cannot easily be implemented to high dimensional problems. This however, becomes even worse for the case of generalized linear models. The main problem is not the bootstrap itself, but the combination of cyclic coordinate descent, cross validation and the predictor-corrector algorithm which should be used for finding the exact optimal λ-knot. The current case of GLM combined with high dimensionality, lead to the fact that the true λ-knots were extremely close to each other, especially at the end of the λ sequence, as we saw from the lasso-paths plots. This had as a result that finding the optimal λ-knot could take up to $20 \sim 30$ minutes for each bootstrap sample. However, this was also the result of inefficiency in the predictor-corrector algorithm. By simulated observations we saw that the step of absolute correlations in the algorithm, would not necessary predict the next true active set correctly, probably because of the high dimensionality of the data. This had as a result that the algorithm would run unnecessary steps many times. Those problems however, concern the time that the bolasso method needs. Its efficiency remains unaffected. Secondly, the fact that bolasso cannot provide trustworthy confidence intervals is an important problem, directly connected to the lack of inference for the lasso. Although one could use bolasso for finding the most relevant coefficients and then use only those in a simple bootstrap without penalty, for

finding confidence intervals, those intervals might not be correct because the initial choice of the coefficients would have been done by the lasso.

The covariance test statistic is not so straightforward, yet promising. The results from the covariance test statistic of bolasso and simple lasso, when applied to the RD dataset, were exactly the same. This was our first indication that the covariance statistic does indeed a good job. We thus concluded that no bolasso needs to be run. This was a naive conclusion according to the results from the NRD dataset. This dataset was not random and thus its results were more important. As we saw from those results, the covariance statistic was misleading on this case.

We found three differences of the covariance test when applied to bolasso and simple lasso, for the NRD dataset. The first one was the fact of one extra coefficient in the model of simple lasso. Although we know that simple lasso tends to select some irrelevant coefficients, the fact that the coefficient's p-value was almost zero was very misleading. Moreover, this extra coefficient was indeed irrelevant and not correlated with any other coefficient. The second one was the fact that not all the bolasso non-zero coefficients were significant (which actually happened also for the RD dataset, but by that time it seemed that bolasso chose wrong.). Since bolasso used Akaike's information criterion and since, according to the theory, the reason for using bolasso is for reducing the uncertainties of simple lasso, one would expect that most of the non-zero coefficients from the bolasso are significant. However, this was not the case according to the covariance statistic. When the covariance statistic was applied to the bolasso estimates, a further reduction of the total number of significant coefficients took place. More specifically, bolasso chose a total number of 64 coefficients to be in the model and if we remove the one that was not in common with the replication, we can say that bolasso chose 63. The covariance test reduced this number to 19 coefficients in the model, which is a big difference, and 18 of those coefficients were in common with the bolasso. The third and final one was the fact that the coefficients of the bolasso that were rejected by the covariance statistic, had indeed high risk ratios. That contradicts with the interpretation of the risk ratios. Risk ratios that are significantly higher or lower from the value of one, might be considered as significant, or at least have p-values that could explain their results (bigger p-values for those risk ratios close to one, lower for those far from one.).

Furthermore, we showed in section 5.5 that the asymptotic distribution of T_k is $Exp(1)$, but not always under the correct hypothesis. We specifically saw that sometimes the distribution under the null hypothesis is indeed $Exp(1)$, but sometimes it isn't. Respectively we saw that for the alternative hypothesis, most of the T_K followed an $Exp(1)$ distribution, but they shouldn't.

Moreover, two positive aspects of the covariance test were found, but their results are still difficult to understand. We saw that even if the initial bolasso for the NRD and the replicated one gave slightly different results, the covariance test chose exactly the same coefficients as significant for both procedures. Moreover, we saw that the covariance test removed all the coefficients that were set to zero at least once among the bootstraps. From those results we see that the covariance statistic captures something about the estimates, but those results are not easy to

interpret.

The article by Lockhart et al. [15] is very inspiring but it lacks precision at some places. First of all, the article states that the case of generalized linear models is not yet proven. Therefore, the authors have not written any detailed analysis for that case. Furthermore, the authors do not make clear if the covariance statistic should take into account the non-significant coefficients, when testing for others. That is, when we test one coefficient by one, in a sequence of λ-knots, should we stop when one coefficient is not significant, exclude the coefficient and then continue, or should we just continue until the end, using the non-significant coefficient in the estimations? For our analysis, we just continued without excluding any non-significant coefficient. Furthermore, if we should have excluded the non-significant, then the λ-knots sequence should be re-approximated again, and again, each time we exclude a non-significant coefficient. This is because this sequence is directly connected with the coefficients in the model (the columns of X matrix). Doing that, would have resulted to a great increase of time that the covariance test would use. Moreover, the interpretation of the p-values is not straightforward at all. Lockhart et al. [15] do not comment on the interpretation of the p-values and Bühlmann et al. [23] warn the users to not naively interpret those p-values as they would usually be interpreted for any other test. The reason for that, is that the hypothesis being tested from the covariance statistic is conditional, given the other coefficients in the active set Ω.

The final conclusion of the analysis is that the bolasso method is still the optimal method to use. Although it is time consuming, its results are more accurate and more easy to interpret. However, its confidence intervals are not trustworthy and probably other methods have to be applied for reducing that problem. On the other hand, work has to be done for the covariance test statistic. The interpretation of the resulting p-values is not easy and we cannot be sure about the asymptotic distribution of T_k for the GLM case. Probably therefore we got strange results about the significance of the risk ratios. Because clearly, the bolasso is based on a non-conditional testing (by using AIC), while the covariance statistic is based on a conditional one. However, the use of covariance test statistic shows that in the future, avoiding bolasso and its time inefficiency, might be achieved by a proper use of this statistic. That would be a great computational advantage;

Chapter 6

Discussion and Further Analysis

6.1 Discussion

The current thesis concerned the investigation of different aspects of the lasso method and its characteristics. Moreover, of secondary priority was the investigation of probable triggers that could have caused myocardial infarction to a set of patients. The whole analysis was based on the case-crossover design. Finally, the implementation of the newly proposed covariance statistic has also concerned.

The analysis concerned two different datasets generated from a small amount of data. The data given for generating the datasets was constituted by a set of 775 drug names and their usage frequencies in a period between $2008 - 2012$. Both datasets included a total number of 75.000 patients IDs and were generated based on the usage frequencies. The generated information that was used in the analysis were the date of the myocardial infarction event for each patient and their drug intakes. Furthermore, the one dataset was completely random, whereas the other dataset had some weights on 100 randomly chosen drugs, such that the myocardial infarction event had a bigger probability for occurring right after an intake of one of those drugs. The investigation of which drug could have caused myocardial infarction was of secondary priority. The reason for that was that both datasets were not real. However, those datasets were used for investigating the lasso method and its strong and weak sides.

The layout of the analysis was based on the case-crossover design. The reason for that was mainly for eliminating internal characteristics of the corresponding likelihood. The case-crossover design has the ability of removing internal characteristics of the subjects, such as age and sex, by allowing each subject to be the case and the control of itself on different time periods. The potential triggers of the study were the drugs. Moreover, one case and one control window were used, both of seven days length and placed seven days apart from each other. The reason for using only one case window was because each patient's follow up period was

assumed to have stopped by the time of the first myocardial infarction event. The reason for using only one control window was for reducing possible seasonal bias. Moreover, the windows were placed using a unidirectional non-localizable design, which resulted in biased estimates. However, the magnitude of the bias was not investigated, but it should be small, according to Lumley and Levy [16]. A simple design was applied for the hazard period of the drugs by allowing no place for minimum induction time and a constant effect period of seven days. Furthermore, the reason for using seven days distance between the case and control windows was because a washout period of seven days was assumed for each drug intake. Based on that layout the likelihood function was created and modified to a form of generalized linear model for estimating the log risk ratios of the coefficients.

The estimation of the log risk ratios was done by the lasso method and the cyclic coordinate descent algorithm. The reason for using the lasso method was because of its ability of setting some of the coefficients exactly to zero by penalizing the likelihood model with a λ factor. This was a great advantage for our case because the total number of coefficients was very big and thus, finding the best model with means of forward or backward selection would be time inefficient. Moreover, the usage of cyclic coordinate descent to the model's convex objective function reduced the total time of the coefficient estimation by estimating the coefficients in a cyclic way, avoiding the inversion of the Hessian.

The lasso method has the disadvantage of selecting irrelevant coefficients to enter the model or selecting a part of a group of correlated coefficients. Those uncertainties were reduced by the use of bolasso, a bootstrap version of lasso. For each bootstrap sample, the estimation of the coefficients was done with the lasso method. Furthermore, coefficients that were frequently set to zero among the bootstraps, were excluded from the model. Finally, the frequency threshold for the allowed number of zero occurrences for each coefficient was found with the Akaike's information criterion.

The most important and most difficult part of the bolasso method was finding the optimal λ for each bootstrap sample. It was the most important part because the λ penalty plays a crucial role on the estimation of the coefficients. This is because the value of λ decides how much each coefficient will be shrunk and probably set to zero. Choosing the wrong λ would result to wrong number of coefficients in the model. The choice of the optimal λ was done by cross validation at each bootstrap sample. A sequence of λ values was generated and the value that gave the lower cross validation error was chosen for estimating the coefficients. However, for the case of generalized linear models finding the optimal λ was not straightforward. The reason for that was that gaps between the total number active coefficients were created with the use of an arbitrary λ sequence. Therefore, the predictor-corrector algorithm was used, before applying cross validation, for approximating a sequence of λ-knots which gave an increase of one, of the total number of active coefficients, for each λ value in it. This however, was very time inefficient and therefore, a modified version of that algorithm was used at each bootstrap, which provided a faster way of choosing the optimal λ at each bootstrap.

Both datasets were used for investigating the characteristics of the bolasso

method. The characteristics that were investigated were the total number of active coefficients, the value of the optimal λ and the cross validation error on that λ. Those characteristics were observed for each bootstrap and were then compared between the datasets. The conclusion from that part of the analysis was that the bolasso method identified the differences of the datasets. The total number of active coefficients and the optimal λ-knot, for the NRD dataset, that were chosen from bolasso, did not differ a lot among the bootstraps. This resulted to more stable estimates with small variance. On the other hand, bolasso gave big variation for both the optimal λ-knot and the total number of active coefficients among the bootstraps, for the RD dataset. Therefore, the confidence intervals of the estimates from that dataset were big.

The second part of the analysis was the implementation of the covariance statistic both on the bolasso estimates and on the simple lasso estimates. Although the covariance statistic is a new test for the lasso inference, and is not satisfactorily developed for the case of generalized linear models, no other inference method exists for the lasso. Therefore, this statistic was used for assessing the significance of the coefficients. However, the results of the statistic turned out to be misleading, under the usual interpretations of significance testing. The results from the statistic when applied to the bolasso estimates were not as expected. The non-zero coefficients of the bolasso method should be significant, assuming that the bolasso should have removed all the non-significant coefficients. However the implementation of the statistic on those estimates gave a further reduction of the significant estimates. Furthermore, the p-values of the coefficients and their estimated risk ratios were contradictory. If the p-values were to be interpreted in the usual way, low p-values should have resulted in risk ratios much higher or lower than the value of one, if the corresponding drug frequencies were also high. However, this was not the case. Moreover, the covariance statistic did not seem to have the ability of correcting the irrelevant choices of the simple lasso method. The conclusion from that result was not that the p-values are incorrect, but they rather test different hypotheses than the bolasso and the Akaike's information criterion do. In general, the conditional hypothesis that the covariance statistic tests seems not to be easy to interpret.

Finally, the distribution of the covariance statistic is $Exp(1)$, but not only under the null hypothesis as expected. It seemed to be difficult to separate the null form the alternative hypothesis by only looking at the plotted quantiles. This means that regardless the true hypothesis, the covariance statistic T_k can asymptotically follow the $Exp(1)$ distribution. This is a major problem because the resulting p-values might not be correct. One cannot reject the null hypothesis since the alternative might have the same distribution, or the other way around;

The final conclusion from the analysis was that bolasso is still the safest way to use. The fact that both the initial bootstrap and the replicated one gave fairly close results, tells us that the bolasso method is stable and trustworthy. Although its confidence intervals are not trustworthy because of the different penalties applied to each bootstrap, the bolasso method effectively chooses the total number of coefficients that should be in the model. On the other hand, the covariance statistic is not well developed for the case of the generalized linear models, but it

is very promising.

6.2 Further Analysis

6.2.1 Use of the Current Thesis

The results of the current thesis will be used on a real and more accurate dataset. This thesis served as a "background" analysis for finding the best method that one should use under the case-crossover design. When the real dataset is available, the bolasso method will be used for estimating the risk ratios of the coefficients. For that case, we will be interested in the risk ratios and the confidence intervals of the estimates. However, finding trustworthy confidence intervals for the bolasso is a research that needs to be done.

Researchers are often concerned about the adverse effects of drugs. This concern is not only about drugs under development, but also about drugs that are already in the market. Ensuring that a drug in a market will not cause any adverse effect is, of course, of great importance. However, for ensuring that a drug can be safely taken, tests have to be done in population samples. This is not always enough though, because every one of us is different and thus, different problems can arise to each one of us when taking a drug. The problem becomes even bigger when two or more drugs react together.

There are known examples for cardiovascular side effects which were discovered after the drugs came to market. One example is rofecoxib (Vioxx), a non-steroidal anti-inflammatory drug used mainly against acute pain conditions, which was withdrawn from the market in 2004 after over 80 million people were using it. It was estimated that between $88.000 \sim 144.000$ heart attacks as side effects were attributed to this drug. Another example is sibutramine (Reductil), an anti-obesity medication, which was withdrawn from the market in 2010 due to the higher risk of stroke and myocardial infarction as side-effects of the drug. By systematically analysing all drugs needing prescription in Norway, we might expect similar but yet unknown adverse side effects for some drugs, but we may also find yet unknown protective effects as well.

By applying methods that allow systematic examination of all potentially existing associations between drugs, their combinations and possible health outcomes, we might reveal potentially important side effects and drug-drug interactions that otherwise might go undetected by classical hypothesis-driven approach. Therefore, no specific hypothesis will be used in the real analysis, but only a global one: drugs have adverse or positive side effects on cardiovascular outcomes.

6.2.2 Modifications for the Methods Used in the Thesis

It would be interesting to use different control or case window's length for each patient. In the current analysis we used a simple hazard model for the effect of the drugs. We assumed that all drugs have minimum induction time equal to zero and seven days effect period. This might be a rather naive assumption. Not all

drugs might have an equal minimum induction time or effect period. This might also be the case for the washout period. Taking those information into account, the windows might be of different length for each drug. None of the articles that the current thesis was based on, raised the issue of windows with different length.

One of the main challenges of the current thesis was the implementation of the algorithms. The bolasso method is extremely time inefficient for high dimensional data. The problem arises when one wishes to approximate the λ-knots by the predictor-corrector algorithm and then use cross validation for finding the optimal one. This could take up to 7 hours for each bootstrap sample and even parallel programming was not efficient enough. Therefore, effective algorithms should be created or the currently used ones should be modified. A small modification was created in the current thesis which reducted the time for each bootstrap to 30 minutes, however the algorithm is greedy and not encouraged to use. Moreover, 30 minutes for each bootstrap sample is still a long time.

Finally, corrections and modifications can be done on the covariance statistic. This statistic is indeed promising and maybe in the future might be used for avoiding bolasso. This however seems not to be an easy task.

Glossary

C | M

C

Coronary artery

The left and right coronary arteries are the only ways by which the heart tissues are being supplied with blood [31]. 35

M

Myocardial tissue

or myocardium, is the thick muscle layer of the heart [31]. 35

Bibliography

[1] Q. Acton, *Myocardial Infarction: New Insights for the Healthcare Professional: 2013 Edition*. ScholarlyEditions, 2013. [Online]. Available: http://books.google.no/books?id=N5Kw5vs_NP8C

[2] M. Avalos, Y. Grandvalet, N. D. Adroher, L. Orriols, and E. Lagarde, "Analysis of multiple exposures in the case-crossover design via sparse conditional likelihood." *Stat Med*, 2012. [Online]. Available: http://www.biomedsearch.com/nih/Analysis-multiple-exposures-in-case/22419612.html

[3] M. Avalos, L. Orriols, H. Pouyes, Y. Grandvalet, F. Thiessard, and E. Lagarde, "Variable selection on large case-crossover data: application to a registry-based study of prescription drugs and road traffic crashes." *Pharmacoepidemiol Drug Saf*, 2013. [Online]. Available: http://www.biomedsearch.com/nih/Variable-selection-large-case-crossover/24136855.html

[4] P. Breheny and J. Huang, "Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection," *The Annals of Applied Statistics*, vol. 5, no. 1, pp. 232–253, 03 2011. [Online]. Available: http://dx.doi.org/10.1214/10-AOAS388

[5] T. T. Cai and M. Yuan, "Comments on "a significance test for the lasso"," 2013. [Online]. Available: http://www.imstat.org/aos/future_papers.html

[6] G. M. Clarke, C. A. Anderson, F. H. Pettersson, L. R. Cardon, A. P. Morris, and K. T. Zondervan, "Basic statistical analysis in genetic case-control studies," *Nat Protoc*, vol. 6, no. 2, pp. 121–33, 2011, clarke, Geraldine M Anderson, Carl A Pettersson, Fredrik H Cardon, Lon R Morris, Andrew P Zondervan, Krina T eng 064890/Wellcome Trust/United Kingdom 081682/Wellcome Trust/United Kingdom 085235/Wellcome Trust/United Kingdom WT91745/Z/10/Z/Wellcome Trust/United Kingdom Research Support, Non-U.S. Gov't England 2011/02/05 06:00 Nat Protoc. 2011 Feb;6(2):121-33. doi: 10.1038/nprot.2010.182. Epub 2011 Feb 3. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/21293453

[7] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap*, ser. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis, 1994. [Online]. Available: http://books.google.no/books?id=gLlpIUxRntoC

[8] J. Fan, F. Han, and H. Liu, "Challenges of Big Data Analysis," *ArXiv e-prints*, Aug. 2013.

[9] J. Fan and Z. T. Ke, "Discussion on "a significance test for the lasso"," 2013. [Online]. Available: http://www.imstat.org/aos/future_papers.html

[10] J. Friedman, T. Hastie, H. Höfling, and R. Tibshirani, "Pathwise coordinate optimization," *The Annals of Applied Statistics*, vol. 1, no. 2, pp. 302–332, 12 2007. [Online]. Available: http://dx.doi.org/10.1214/07-AOAS131

[11] J. H. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent," *Journal of Statistical Software*, vol. 33, no. 1, pp. 1–22, 2 2010. [Online]. Available: http://www.jstatsoft.org/v33/i01

[12] S. Hernández-Díaz, M. A. Hernán, K. Meyer, M. M. Werler, and A. A. Mitchell, "Case-crossover and case-time-control designs in birth defects epidemiology," *American Journal of Epidemiology*, vol. 158, no. 4, pp. 385–391, 2003. [Online]. Available: http://aje.oxfordjournals.org/content/158/4/385.abstract

[13] H. Janes, L. Sheppard, and T. Lumley, "Overlap bias in the case-crossover design, with application to air pollution exposures," *Statistics in Medicine*, vol. 24, no. 2, pp. 285–300, 2005. [Online]. Available: http://dx.doi.org/10.1002/sim.1889

[14] S. Li, B. Mukherjee, S. Batterman, and M. Ghosh, "Bayesian analysis of time-series data under case-crossover designs: Posterior equivalence and inference," *Biometrics*, vol. 69, no. 4, pp. 925–936, December 2013. [Online]. Available: http://ideas.repec.org/a/bla/biomet/v69y2013i4p925-936.html

[15] R. Lockhart, J. Taylor, R. J. Tibshirani, and R. Tibshirani, "A significance test for the lasso," 2013. [Online]. Available: http://www.imstat.org/aos/future_papers.html

[16] T. Lumley and D. Levy, "Bias in the case – crossover design: implications for studies of air pollution," *Environmetrics*, vol. 11, no. 6, pp. 689–704, 2000. [Online]. Available: http://dx.doi.org/10.1002/1099-095X(200011/12)11:6<689::AID-ENV439>3.0.CO;2-N

[17] J. Lv and Z. Zheng, "Discussion : "a significance test for the lasso"," 2013.

[18] M. Maclure, "The case-crossover design: a method for studying transient effects on the risk of acute events," *Am J Epidemiol*, vol. 133, no. 2, pp. 144–53, 1991, maclure, M eng HL 41016/HL/NHLBI NIH HHS/ Comparative Study Research Support, U.S. Gov't, P.H.S. 1991/01/15 Am J Epidemiol. 1991 Jan 15;133(2):144-53. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/1985444

[19] M. Maclure and M. A. Mittleman, "Should we use a case-crossover design?" *Annu Rev Public Health*, vol. 21, pp. 193–221, 2000, maclure, M Mittleman, M A eng Review 2000/07/08 11:00 Annu Rev Public Health. 2000;21:193-221. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/10884952

[20] J. Marin and C. Robert, *Bayesian Core: A Practical Approach to Computational Bayesian Statistics*, ser. Springer Texts in Statistics. Springer, 2007. [Online]. Available: http://books.google.no/books?id=Lx74oEz5pmsC

[21] W. Pan, "Akaike's information criterion in generalized estimating equations." *Biometrics*, vol. 57, no. 1, pp. 120–5, 2001. [Online]. Available: http://www.biomedsearch.com/nih/Akaikes-information-criterion-in-generalized/11252586.html

[22] M. Y. Park and T. Hastie, "L1-regularization path algorithm for generalized linear models," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 69, no. 4, pp. 659–677, 2007. [Online]. Available: http://dx.doi.org/10.1111/j.1467-9868.2007.00607.x

[23] L. M. Peter Bühlmann and S. van de Geer, "Discussion of "a significance test for the lasso"," 2013. [Online]. Available: http://www.imstat.org/aos/future_papers.html

[24] G. Rodríguez, "Lecture notes on generalized linear models," 2007. [Online]. Available: http://data.princeton.edu/wws509/notes/

[25] S. Schneeweiss, T. Sturmer, and M. Maclure, "Case-crossover and case-time-control designs as alternatives in pharmacoepidemiologic research," *Pharmacoepidemiol Drug Saf*, vol. 6 Suppl 3, pp. S51–9, 1997, schneeweiss, S Sturmer, T Maclure, M eng England 2004/04/10 05:00 Pharmacoepidemiol Drug Saf. 1997 Oct;6 Suppl 3:S51-9. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/15073755

[26] N. Simon, J. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for cox's proportional hazards model via coordinate descent," *Journal of Statistical Software*, vol. 39, no. 5, pp. 1–13, 2011. [Online]. Available: http://www.jstatsoft.org/v39/i05/

[27] K. Thygesen, J. S. Alpert, H. D. White, and on behalf of the Joint ESC/ACCF/AHA/WHF Task Force for the Redefinition of Myocardial Infarction, "Universal definition of myocardial infarction," *Circulation*, vol. 116, no. 22, pp. 2634–2653, 2007. [Online]. Available: http://circ.ahajournals.org/content/116/22/2634.short

[28] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society, Series B*, vol. 58, pp. 267–288, 1994.

[29] R. J. Tibshirani and R. Tibshirani, "A bias correction for the minimum error rate in cross-validation," *The Annals of Applied Statistics*, vol. 3,

no. 2, pp. 822–829, 06 2009. [Online]. Available: http://dx.doi.org/10.1214/08-AOAS224

[30] K. Tretyakov, S. Laur, G. Smant, J. Vilo, and P. Prins, "Fast probabilistic file fingerprinting for big data," *BMC Genomics*, vol. 14 Suppl 2, p. S8, 2013, tretyakov, Konstantin Laur, Sven Smant, Geert Vilo, Jaak Prins, Pjotr eng Research Support, Non-U.S. Gov't England 2013/03/06 06:00 BMC Genomics. 2013;14 Suppl 2:S8. doi: 10.1186/1471-2164-14-S2-S8. Epub 2013 Feb 15. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/23445565

[31] A. Weinhaus and K. Roberts, *Anatomy of the Human Heart*, P. Iaizzo, Ed. Humana Press, 2005. [Online]. Available: http://dx.doi.org/10.1007/978-1-59259-835-9_4

[32] T. T. Wu, Y. F. Chen, T. Hastie, E. Sobel, and K. Lange, "Lange k: Genomewide association analysis by lasso penalized logistic regression," *Bioinformatics*.

Appendix A

Programming and Big Data Challenges

This appendix concerns personal experiences and programming difficulties along the way of the analysis of the current thesis, as well as references of the created algorithms and in-built packages that were used.

A.1 Big Data Challenges

A.1.1 Definition of Big Data

Technology has been developed efficiently in the last years, providing useful and more accurate tools which are used in many research areas. Moreover, the more developed technology becomes, the cheaper the research methods become, giving researchers the opportunity of creating more and more data with easier ways. This leads to a massive amount of data (Big Data), which is rapidly increasing. On the one hand, sciences, engineering and social science are positively affected by this significant technological development. Lagging, exchanging and comparing data have become easier and faster, leading to more accurate results. On the other hand, this big amount of data yields many challenges that a researcher has to take into account in order to run a research which will provide efficient results [8].

Big Data is a globally used term, which describes the *massive amount*[1] and *high dimensionality*[2] data. Although Big Data gives an extremely helpful privilege to all kinds of science, it has some major difficulties, which can be real challenges for a researcher willing to use some sort of Big Data. Some of those challenges are noise accumulation, spurious correlation and identical endogeneity, computational and algorithmic issues, and homogeneity [8].

Hypothesis testing and decision-making depend on the high dimensionality of Big Data. Thus, a large number of parameters have to be simultaneously estimated

[1]The massive amount of data comes from the big sample sizes used on researches [8].

[2]High dimensionality means that a big number of sub-populations are used for researches [8].

and tested. Errors can occur because of the large number of those parameters. This is called noise accumulation effect and is highly common in high dimensional data. This effect has to be handled with great caution because it can result to wrong parameter estimation and thus misleading hypothesis judgement. Moreover, spurious correlation can be caused by the high dimensionality, meaning that random variables, which are in fact uncorrelated, may have high sample correlation. This can also lead to false decision-making and, therefore, has to be treated carefully. Finally, another issue of high dimensionality is the identical endogeneity. Consider, for example, that we have a regression model of the form $Y = \sum_{j=1}^{d} \beta_j X_j + \epsilon$. Endogeneity means that some of the regression predictors are correlated with the error term. Regression models assume that the error term and the regression predictors are uncorrelated with each other. This assumption is crucial for various statistical procedures that are going to take place in the later analysis (variable or model selection for example) [8].

Furthermore, the huge amount of data combined with the high dimensionality makes not only the analysis but also the transferring and lagging of data extremely difficult and time costly [8], [30]. The most used approach for solving those problems is to divide the problem into many sub-problems (smaller datasets). Then store them as different-smaller units and run separate analyses to those sub-units. At the end, the results from those analyses in the sub-units are combined to form a global result for the main problem [8].

Finally, homogeneity is another challenge of Big Data. Homogeneity comes from the fact that Big Data is created from many resources and sub-populations. Each of those has their unique characteristics that are not in common to the rest of the sub-populations. Those characteristics can serve as outliers and careful attention should be placed on their analysis. However, due to the big amount of data in the case of Big Data, there are more effective ways for identifying and understanding those outliers, than in small sample data sets [8].

Likely, according to Fan [8] there are some statistical techniques that can be implemented to Big Data and help researchers analyse the data sets with a more effective way, which can actually result to trustworthy results. Finally, according to Tretyakov et al. [30] there are also many algorithmic databases and programs that solve adequate a big part of the computational and data-transferring problems.

A.1.2 Challenges During this Thesis

Because of the massive data that is being daily collected, all kinds of science get more accurate results in every kind of research. For example, in statistical science, more accurate estimates can be obtained, which lead to more accurate predictions. However, in my personal experience from the current analysis, it was extremely difficult to access both datasets, because each of them contained more than $\sim$ 67.500.000 rows. The challenge was not only a matter of time, but also a matter of space; As a master student in statistics I had never encountered so massive datasets in any of the obligatory projects, as those used here. Many different algorithms, packages and methods (including parallel programming) were used for trying not to reduce the datasets. I didn't want to reduce the datasets because I

know that in the future I will most probably work with datasets of that size, and I wanted to have an experience. However, what I gained from this, was the fact that not all currently used statistical algorithms (like cross validation) are the optimal choices if one works with massive datasets. Therefore, mathematical modifications should be done to some of the commonly used algorithms and new methods for faster programming would be really helpful (which might already exist and I don't know.).

Finally, the other thing that I gained from this work, was that is not easy to extract data from the sources. What I mean is that, the reason that I didn't get the real dataset in time, was because it takes time to get data and at the same time ensure that the any patient's personal information will remain unknown. This is something that I completely understand, but I had never thought about.

A.2 Programming

A.2.1 Created Algorithms for the Thesis

In this subsection an overview of the created algorithms is given. Those algorithms consist both of main modules and functions. Moreover, many of them can be found in slightly different forms which were modified for satisfying different needs (like plots, estimation etc.). Here only the names and function of those algorithms is given, more details can be found in the algorithms.

Main Modules

- *NRD_generator_part1, NRD_generator_part2, RD_generator_part1, RD_generator_part2:* Modules for generating the datasets.
- *Exposures_matrices_I:* Module for counting the exposures.
- *Lasso_CV_Step_plots:* Module for plotting the lasso paths and the CV error.
- *Bolasso_main, Bolasso_Part_1, Bolasso_Part_2:* Modules for running the bolasso in parallel.
- *Bolasso_plots:* Module for plotting the results from the bolasso, like the active set, the λ-knots and the CV errors.
- *Significance, Significance_lasso:* Modules for the significance testing for the bolasso and the lasso respectively.

Functions

- *Chi_squared_generator:* Function used for generating information for the datasets.

- *Matrix_creation_I,data_manipulation, Model_matrix:* Functions for modifying the datasets and creating the data matrices X.
- *Bootstrap_simple:* Function for running bolasso with constant λ.
- *ST, CCD:* Functions for the soft threshold and the cyclic coordinate descent, respectively.
- *P_E, P_E_gap:* Different forms of the cross validation function.
- *Activated, Pathwise, Pathwise_gap, Find_optimal, First_step:* Functions for finding the optimal λ-knot via the predictor-corrector algorithm.

A.2.2 Packages for Latex and R-studio

This subsection serves as a reference section for the packages used on this thesis. The reason is that the creators of the those packages wish that any user will refer to them, and we have to respect that.

Packages used for the LaTeX: amsmath, algorithm2e, graphicx, adjustbox, caption, enumitem, subcaption, float, ifthen, xkeyval, xfor, amsgen, etoolbox, longtable, supertabular, array, multicol, glossaries, geometry, cite, multirow, url, fontspec, lmodern, afterpage, xcolor, comment, tocbibind, babel, datatool.

Packages used for R-studio: Matrix, doParallel, doSNOW, ggplot2, plyr.

Appendix B

Proof of Equation (3.4)

In subsection 3.3.3 we showed that if the log-likelihood function $\ell(\beta)$ is differentiable, then the forward and backward directional derivatives of the objective function $f(\beta, \lambda) = \ell(\beta) - \lambda \sum_j |\beta_j|$ are equal for every β, except for $\beta = 0$ where it does not exist. In this appendix we will show that the concave function f and the convex function $-f(\beta, \lambda)$ will give the same soft threshold function.

Consider first the concave function $f(\beta, \lambda) = \ell(\beta) - \lambda \sum_j |\beta_j|$. When we derivate with respect to β_j and then set the derivative equal to zero we get accordingly:

If $\beta_j > 0$**:**

$$\frac{\partial \ell(\beta)}{\partial \beta_j} - \lambda = 0 \Leftrightarrow \hat{\beta}_j = \frac{-A - \lambda}{-B} = \frac{A + \lambda}{B}$$

If $\beta_j < 0$**:**

$$\frac{\partial \ell(\beta)}{\partial \beta_j} + \lambda = 0 \Leftrightarrow \hat{\beta}_j = \frac{-A + \lambda}{-B} = \frac{A - \lambda}{B}$$

where A is a function of the explanatory variables x_{ij} and the coefficients $\beta_{l \neq j}$, and B is a function of the explanatory variables x_{ij}. If we now consider the convex function $-f(\beta, \lambda) = -\ell(\beta) + \lambda \sum_j |\beta_j|$, we get accordingly:

If $\beta_j > 0$**:**

$$-\frac{\partial \ell(\beta)}{\partial \beta_j} + \lambda = 0 \Leftrightarrow \hat{\beta}_j = \frac{-A + \lambda}{-B} = \frac{A - \lambda}{B}$$

If $\beta_j < 0$**:**

$$-\frac{\partial \ell(\beta)}{\partial \beta_j} - \lambda = 0 \Leftrightarrow \hat{\beta}_j = \frac{-A - \lambda}{-B} = \frac{A + \lambda}{B}$$

As we can see, for the concave function with $\beta_j > 0$ we get the same result as for the convex function with $\beta_j < 0$, and respectively for the other option. This

however still lead us to the same soft threshold function because it also depends both on the way the λ penalty is used ($+\lambda$ or $-\lambda$), and on the minimization or maximization of the convex or concave function, respectively.

Appendix C

p-values and Estimates

Table C.1: Estimates for the NRD Dataset.

Drug ID	Bolasso Estimates			Simple Lasso Estimates		
	Estimates	CI	p-value	Estimates	CI	p-value
N02AA59	0.8974	$(0.8694, 0.9308)$	0	0.8982	$(0.8675, 0.9294)$	0
J01CA08	0.8531	$(0.7856, 0.9123)$	0	0.8556	$(0.7905, 0.9186)$	0
R06AE07	0.3611	$(0.3231, 0.4019)$	0	0.3591	$(0.3183, 0.4)$	0
G03CA03	0.7834	$(0.7168, 0.8531)$	0	0.787	$(0.7149, 0.8583)$	0
N07BA03	0.4187	$(0.3559, 0.4788)$	$4.1e-10$	0.4214	$(0.359, 0.4853)$	$4.051e-10$
C01DA02	0.515	$(0.4334, 0.5752)$	0	0.5169	$(0.4419, 0.5896)$	0
D07AC13	0.7264	$(0.6126, 0.8155)$	0.01007	0.7288	$(0.6348, 0.8294)$	0.01007
S01GX02	0.552	$(0.4593, 0.627)$	$2.887e-15$	0.5527	$(0.469, 0.6391)$	$2.887e-15$
C10AA01	0.3565	$(0.2052, 0.3819)$	0.002219	0.3555	$(0.2694, 0.4393)$	0.002219
R06AX27	0.7366	$(0.616, 0.8385)$	0.01143	0.7421	$(0.6297, 0.8594)$	0.01146
B01AC06	0.2423	$(0.1058, 0.2646)$	0.01892	0.2439	$(0.1636, 0.3233)$	0.01904
R01AD05	0.6082	$(0.487, 0.6945)$	0	0.6135	$(0.5116, 0.7195)$	$3.523e-09$
C10AA05	1.19	$(0.8738, 1.279)$	0.0001212	1.191	$(0.9839, 1.407)$	0.0001159
R05CB01	0.7583	$(0.5483, 0.8414)$	$1.687e-05$	0.7577	$(0.6113, 0.905)$	$3.943e-05$
R06AX26	0.6558	$(0.4879, 0.7637)$	$1.048e-10$	0.6522	$(0.521, 0.7906)$	$2.029e-10$
J01FA09	0.5746	$(0.4218, 0.7007)$	0.01491	0.5828	$(0.4489, 0.727)$	0.01235
S01EE01	0.5217	$(0.3341, 0.6396)$	$3.354e-05$	0.5193	$(0.3751, 0.6665)$	$4.814e-05$
J01EE01	0.4138	$(0.2463, 0.5301)$	0.01586	0.4191	$(0.2743, 0.565)$	0.01426
G04BE03	1.106	$(0.6968, 1.237)$	0.00871	1.106	$(0.8462, 1.387)$	0.004698
S03CA04	X	X	X	0.863	$(0.7292, 0.9972)$	$1.11e-16$

This table contains the bolasso estimates (left), that were considered significant from the covariance test, as well as the significant estimates from the simple lasso (right). CI stands for the confidence intervals of the estimates. The estimates of this table are the log risk ratios of the β_j coefficients from the likelihood. The red X on the last line is a missing coefficient for the bolasso that was considered as significant for the simple lasso.

Table C.2: Estimates for the RD Dataset.

Drug ID	Bolasso Estimates			Simple Lasso Estimates		
	Estimates	CI	p-value	Estimates	CI	p-value
B01AC06	0.09755	$(0.06088, 0.1339)$	$3.04e-06$	0.1121	$(0.07548, 0.146)$	$3.04e-06$
C10AA01	0.06577	$(0.02741, 0.1037)$	0.0128	0.08213	$(0.04003, 0.1229)$	0.0128

Table for the significant coefficients from the bolasso (left), and simple lasso (right), after the implementation of the covariance test. The estimates of this table are the log risk ratios of the β_j coefficients from the likelihood.

Table C.3: Risk Ratio Estimates for the NRD Dataset.

Drug ID	Bolasso Estimates			Frequencies
	Risk Ratio	CI	p-value	
N02AA59	2.453	(2.386, 2.537)	0	546.65
J01CA08	2.347	(2.194, 2.49)	0	230.21
R06AE07	1.435	(1.381, 1.495)	0	214.95
G03CA03	2.189	(2.048, 2.347)	0	137.98
N07BA03	1.52	(1.427, 1.614)	4.1e-10	87.28
C01DA02	1.674	(1.543, 1.777)	0	113.85
D07AC13	2.068	(1.845, 2.26)	0.01007	48.92
S01GX02	1.737	(1.583, 1.872)	2.887e-15	46.40
C10AA01	1.428	(1.228, 1.465)	0.002219	689.44
R06AX27	2.089	(1.851, 2.313)	0.01143	34.13
B01AC06	1.274	(1.112, 1.303)	0.01892	809.36
R01AD05	1.837	(1.627, 2.003)	0	45.08
C10AA05	3.288	(2.396, 3.593)	0.0001212	262.42
R05CB01	2.135	(1.73, 2.32)	1.687e-05	172.88
R06AX26	1.927	(1.629, 2.146)	1.048e-10	21.18
J01FA09	1.776	(1.525, 2.015)	0.01491	52.99
S01EE01	1.685	(1.397, 1.896)	3.354e-05	21.76
J01EE01	1.513	(1.279, 1.699)	0.01586	49.92
G04BE03	3.024	(2.007, 3.445)	0.00871	70.50

Transformed risk ratio estimates from the C.1 table, for the bolasso estimates. The estimates of this table are the risk ratios of the β_j coefficients from the likelihood. The frequencies column is the initial frequencies per 1000 people in the period of five years (2008-2012).

Table C.4: Part of Non-significant Risk Ratios for the NRD Dataset.

Drug ID	Bolasso Estimates		
	Risk Ratio	CI	p-value
S02BA07	2.134	(1.591, 2.898)	0.4134
D10AD01	2.118	(1.585, 2.779)	0.1886
J01MA02	2.003	(1.828, 2.201)	0.2338
N02CC01	1.993	(1.785, 2.222)	0.1211
A03FA01	1.904	(1.726, 2.107)	0.4445
N03AF01	1.883	(1.386, 2.535)	0.3367
M01AC06	1.774	(1.255, 2.597)	0.1392
A10BB12	1.732	(1.241, 2.408)	0.5251
D07BC01	1.662	(1.356, 2.009)	0.09434
C09DA03	1.61	(1.121, 2.302)	0.06482

Ten risk ratios from coefficients of the bolasso, that were considered as non-significant from the covariance test. The estimates of this table are the risk ratios of the β_j coefficients from the likelihood.

Table C.5: Replicated Estimates for the NRD Dataset.

Drug ID	Bolasso Estimates			Simple Lasso Estimates		
	Estimates	CI	p-value	Estimates	CI	p-value
N02AA59	0.9008	(0.8698, 0.9312)	0	0.8982	(0.8675, 0.9294)	0
J01CA08	0.85	(0.7865, 0.9126)	0	0.8556	(0.7905, 0.9186)	0
R06AE07	0.3629	(0.3233, 0.4015)	0	0.3591	(0.3183, 0.4)	0
G03CA03	0.7842	(0.714, 0.8583)	0	0.787	(0.7149, 0.8583)	0
N07BA03	0.4176	(0.3525, 0.4786)	$4.094e-10$	0.4214	(0.359, 0.4853)	$4.051e-10$
C01DA02	0.506	(0.4347, 0.5757)	0	0.5169	(0.4419, 0.5896)	0
D07AC13	0.7111	(0.6132, 0.8116)	0.01007	0.7288	(0.6348, 0.8294)	0.01007
S01GX02	0.545	(0.4607, 0.625)	$2.887e-15$	0.5527	(0.469, 0.6391)	$2.887e-15$
C10AA01	0.291	(0.2028, 0.3795)	0.002219	0.3555	(0.2694, 0.4393)	0.002219
R06AX27	0.7294	(0.6168, 0.8398)	0.01146	0.7421	(0.6297, 0.8594)	0.01146
B01AC06	0.1856	(0.1083, 0.2653)	0.01902	0.2439	(0.1636, 0.3233)	0.01904
R01AD05	0.5902	(0.4871, 0.6968)	0	0.6135	(0.5116, 0.7195)	$3.523e-09$
C10AA05	1.083	(0.8817, 1.286)	0.0001216	1.191	(0.9839, 1.407)	0.0001159
R05CB01	0.6904	(0.5447, 0.8396)	$1.541e-05$	0.7577	(0.6113, 0.905)	$3.943e-05$
R06AX26	0.6258	(0.4911, 0.7618)	$1.048e-10$	0.6522	(0.521, 0.7906)	$2.029e-10$
J01FA09	0.5613	(0.4242, 0.7019)	0.01527	0.5828	(0.4489, 0.727)	0.01235
S01EE01	0.483	(0.329, 0.6416)	$3.354e-05$	0.5193	(0.3751, 0.6665)	$4.814e-05$
J01EE01	0.3919	(0.2468, 0.5306)	0.01582	0.4191	(0.2743, 0.565)	0.01426
G04BE03	0.9755	(0.7074, 1.242)	0.008043	1.106	(0.8462, 1.387)	0.004698
S03CA04	**X**	**X**	**X**	0.863	(0.7292, 0.9972)	$1.11e-16$

The significant estimates from the replicated bolasso (left), after the implementation of the covariance test. The right panel is the same as for the C.1 table. The estimates of this table are the log risk ratios of the β_j coefficients from the likelihood.

Appendix D

Q-Q Plots

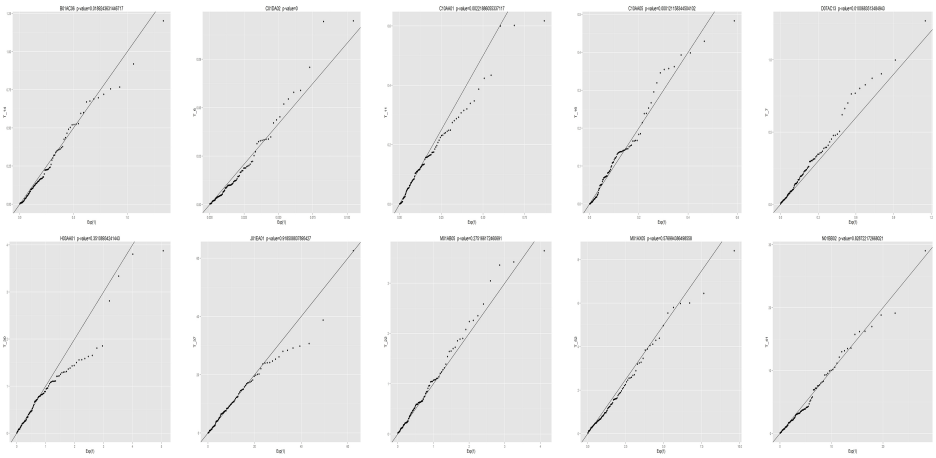

Figure D.1: Quantile-Quantile plots for the covariance statistic T_k against the quantiles of $exp(1)$ distribution, for some of the bolasso coefficients of the NRD dataset. The top row belongs to coefficients with p-value< 0.05, while the bottom row belongs to coefficients with p-value> 0.05.

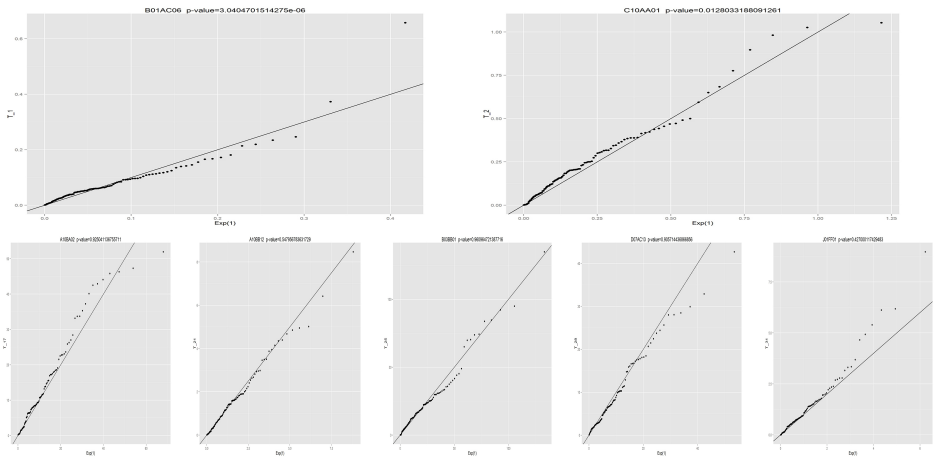

Figure D.2: Quantile-Quantile plots for the covariance statistic T_k against the quantiles of $exp(1)$ distribution, for some of the bolasso coefficients of the RD dataset. The top row belongs to coefficients with p-value< 0.05, while the bottom row belongs to coefficients with p-value> 0.05. Note that we had only two significant coefficients on that case.

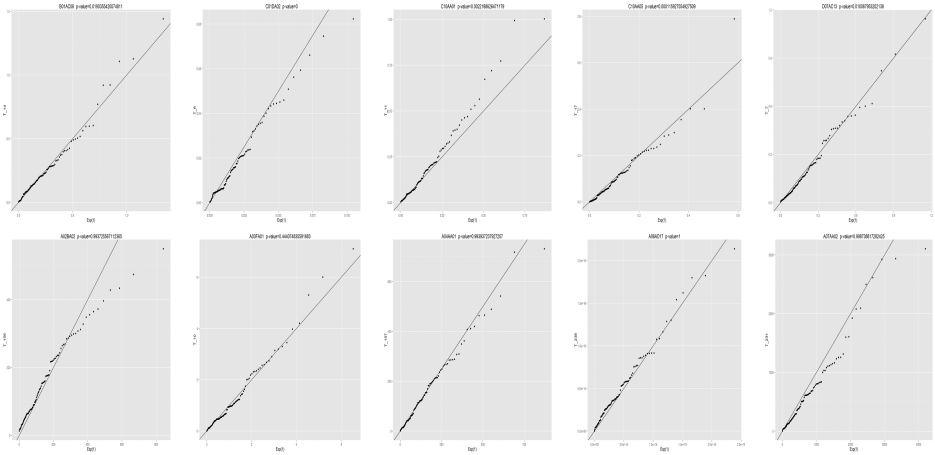

Figure D.3: Quantile-Quantile plots for the covariance statistic T_k against the quantiles of $exp(1)$ distribution, for some of the simple lasso coefficients of the NRD dataset. The top row belongs to coefficients with p-value< 0.05, while the bottom row belongs to coefficients with p-value> 0.05.

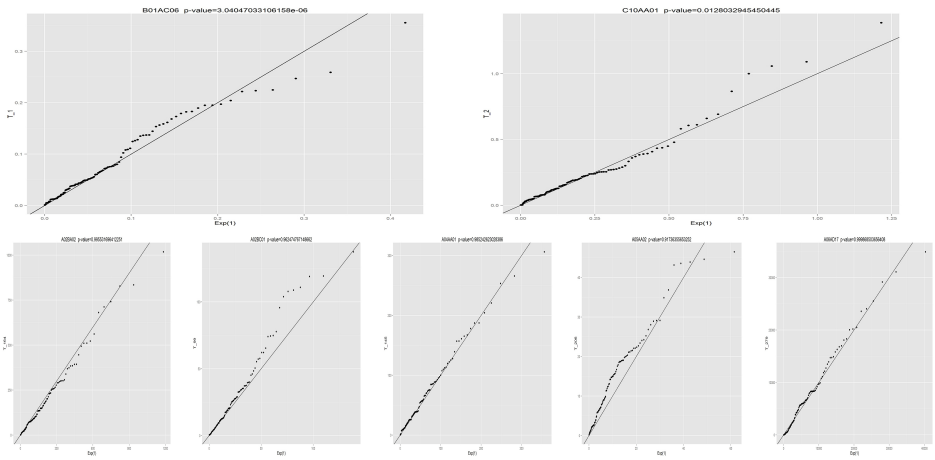

Figure D.4: Quantile-Quantile plots for the covariance statistic T_k against the quantiles of $exp(1)$ distribution, for some of the simple lasso coefficients of the RD dataset. The top row belongs to coefficients with p-value< 0.05, while the bottom row belongs to coefficients with p-value> 0.05. Note that we had only two significant coefficients on that case.