The Instability of Cross-Validated Lasso

by

Kine Veronica Lund

THESIS

for the degree of

Master of Science

(Master i Modellering og dataanalyse)

Faculty of Mathematics and Natural Sciences
University of Oslo

December 2013

Det matematisk-naturvitenskapelige faultet
Universitetet i Oslo

Abstract

In a situation where the number of available covariates greatly exceeds the number of observations, the fitting of a regression model to explain the connection between the response and the explanatory variables can be a challenging task. The problem can be compared to a set of equations with more unknowns than there are equations and requires application of a regularisation method to result in a useful solution. There are several such methods, with different properties. This thesis focuses on one such method: the Lasso in combination with cross-validation (CV) to determine the level of regularisation. Specifically, we consider the method when applied on survival data where the covariates are thousands of gene expression levels.

The combination of Lasso and CV proves to be unstable in the sense that repeated application of the standard R implementation often give varying results. This study's main focus is to investigate what the causes of this instability may be.

Data was simulated to map the factors that affect the stability. The simulated data sets' properties are easy to control and the effects on the regularisation results are easily observed.

The tests show that the CV process cause marked instability (varying results) when the division into training and test sets involve test sets with size larger than one. Moreover, the stability of the regularisation depends on the properties of the data set.

A unique prediction result is preferable to easily choose a prognostic gene signature. However, a range of signatures from repeated regularisations can be utilised to indicate the accuracy of the suggested signature.

This thesis maps several factors that affect the stability of Lasso and CV, and will hopefully contribute to caution - be a warning flag - when utilising the Lasso method to find a prognostic model.

Acknowledgments

This thesis is the completion of more than five years of a great time at the University of Oslo. First, I would like to thank my supervisors Knut Liestøl and Ole Christian Lingjærde. Thank you for taking the time to give great advice while having busy schedules, and for the useful feedback given continuously from the very beginning of the project. The thesis of their previous master student, Hege Størvold, has been of great use throughout my work, especially during the first weeks when everything seemed a bit greek to me.

The lively environment in the study room during my next-to-last semester prepared me well for the final semester. Thanks for the coffee, the race for being first there in the morning and the positive study influence.

There are many people who have contributed to the many happy happenings and memorable memories during my time at UiO. Without HRH Ursus Major and his Realistforeningen, these years would have been a drag. RF has also supplied me with two of my best friends, Ida and Helena. Without your support and our frequent tea breaks, I would have consumed less tea and been less sane.

Thanks to my family for always being there for me and for supporting my choices.

Last but not least, thank you, Øyvind. Your patience, motivation, pep talks and focus have been essential these last months and always are.

Contents

List of Figures

List of Abbreviations

λ penalty parameter, page 3

ρ correlation, page 31

k the k'th has gene effect 0.5, page 32

n number of observations, page 8

p number of genes, page 8

CRC colorectal cancer, page 2

CV cross-validation, page 3

CVL cross-validated likelihood, page 20

DNA deoxyribonucleic acid, genetic molecules, page 1

FPR false positive rate, page 56

GLM generalised linear model, page 27

Lasso least absolute shrinkage and selection operator, page 11

LOOCV leave-one-out cross-validation, page 17

mRNA messenger RNA, page 1

PSE predicted squared error, page 17

RNA ribonucleic acid, genetic molecules, page 1

ROC receiver operating characteristic, page 56

TPR true positive rate, page 56

Chapter 1

Introduction

Both scientists and non-scientists are fascinated by the blueprint of life, our DNA. The information stored in the DNA tells much about you beyond what is visible on the surface. There are for example numerous conditions that have been established as being completely or partially genetically determined, ranging from rare diseases such as sickle-cell anemia, to common and harmless properties such as baldness. Investigation of our genome, in order to reveal links between diseases and the DNA, is important to be able to predict future illness. Furthermore, the progress and response to a particular treatment may depend on the patient's genotype (the complete set of genes).

Some diseases, including cancer, are not normally hereditary, even though they are genetically caused. A mutation may occur in a cell, e.g. because of external factors such as radiation. All cells deriving from that cell will inherit the mutation. Some mutations may be tumourigenic and increase the chance that the cells multiply rapidly and uncontrolled, resulting in a tumour.

If a disease is completely or partially genetically determined, one faulty gene or a larger group of genes may account for some of the reason. These genes may be linked to each other, or they may be independent. Since genes (or gene products) interact, a change in one gene may cause altered expression of several genes; genes express themselves in mRNA (messenger RNA). The DNA string of the gene is the recipe for a protein through protein synthesis. The messenger that brings the recipe from the DNA string to the production of proteins is the mRNA. The quantity of mRNA (and therefore also the protein product) for a gene may vary over time and is referred to as the gene's expression. (For more motivation and introduction to biology, see Section 2.1.)

Changed expression levels may signal malignant changes. Humans have more than 20,000 genes and there is usually a limited number of patients with the illness or property in interest. The great number of genes compared to the number of patients brings on a challenge for every biologist when trying to find features separating malignant cells from healthy ones. Statistical methods may reduce the number of candidate cancer genes until the few genes supposedly most relevant

to the patient's condition are left.

Several methods exist for this purpose. One, the Lasso method, reduces the supposed significance of certain genes to zero such that a strict subset of genes is left. Other methods, like ridge regression, reduces the supposed significance close to zero, but not quite, such that it is harder to rule out which genes affect the disease. The focus in this thesis is on the Lasso alternative because of the useful variable selection property when coefficients are set to zero and because it is the method of choice in a recent and well acknowledged paper[1]. In this paper the Lasso is applied on colorectal cancer (CRC) data[1] and discusses which genes make up a prognostic gene signature. The study concluded with a seemingly useful answer, a prognostic 7-gene signature, despite the fact that the repeated application of Lasso gave different answers. It also raised several questions:

- Why is the result so different when the input is identical every time? The instability is worrying.
- Is the method dependent on details of the data set?
- Should other, or modified, methods be applied to verify the result?

In this thesis, the reader is thought of as a person with a certain interest and understanding of basic statistic topics such as linear regression and probability models. These topics are well explained in [2]. No prior knowledge about biology or genetics is required to understand and find interest in the study. There will be few biological details, but the necessary information will be explained.

1.1 Motivating example

Globally, about 1.2 million people are diagnosed with CRC and about 600,000 die of the disease every year. It is the fourth most common cause of death by cancer, more present in developed countries than in less developed countries[3].[1] Four out of five CRC patients are treated with surgery where the tumour is removed[5].

It is common to use cancer stage as a prediction for the success of treatment of CRC[6]. This is an unreliable prediction, and it would be useful if genomic data could be utilised to see if the patient is likely to respond well to treatment.

A paper published in 2012 by Sveen et al.[1] uses such data from 95 patients to find a connection between treatment success and gene expression. Specifically, a gene signature (i.e. a combination of genes) predicting the treatment result is sought. The authors' aim was to put the patients in a high risk group and in a low risk group. The paper uses the R package `penalized` in order to do the cross validation and Lasso regularisation. There is also another package called

[1]The disease is strongly correlated with a western diet consisting of e.g. red meat, fat and alcohol[4].

`glmnet`, which is applied on the same sort of problems as `penalized`, authored by Friedman, Hastie and Tibshirani, the latter known for proposing the Lasso, see Section 2.2.2. `glmnet` has been used among others by [7][8]. For more information about the R packages, see Section 2.3.

When approaching the two methods with the same data and parameters, the results may be different, as described below and in Section 3.1.

1.1.1 Rough summary of the method used in Sveen's paper

Filtering. The data set used in the CRC paper[1] initially contains information about the expression level of about 20,000 genes in 95 patients. Genes with p-values from Cox regression greater than 0.5 or variance less than 0.2 were considered non-interesting, see Figure 1.1. Such p-values suggest that the genes linked to them probably are irrelevant to any difference between the patients. Note that the null hypothesis here is that each gene is *not* relevant to the survival of patients.

Similarly, low variance genes are unlikely to be relevant to the response. Consequently, genes that differ between the patients are sought.

The R package *penalized*. After filtering based on variance and p-value, Sveen et al.[1] were left with a little more than 3,000 genes. Starting with these genes, the challenge was to end up with a predictive model with just a few explanatory variables: the genes relevant to the response. In order to achieve this, they used the `optL1` function in the R package `penalized`. `optL1` is an implementation of the Lasso method, see Section 2.2.2 and can perform Cox regression, as well as other types of regression.

Lasso. Lasso introduces a penalty parameter $\lambda > 0$ which decides how strict the punishment is for including more genes in the model. Varying λ will vary the number of genes included. The `optL1` function basically works like this:

1. Divide the data in subsets, e.g. 10 groups of size 9-10 (95 patients in total).

2. For a fixed λ, use the Lasso on the data from all patients, except those from one of the subsets to fit a (Cox) model.

3. Find the error on the excluded subset.

4. Repeat step 2 and 3 until all subsets have been removed once.

5. Sum the errors.

This process is called **cross-validation** (CV), see Section 2.2.7, and is repeated on a sequence of reasonable λ-values. The λ that gave the smallest error is noted and used to make a predictive model based on the complete data set.

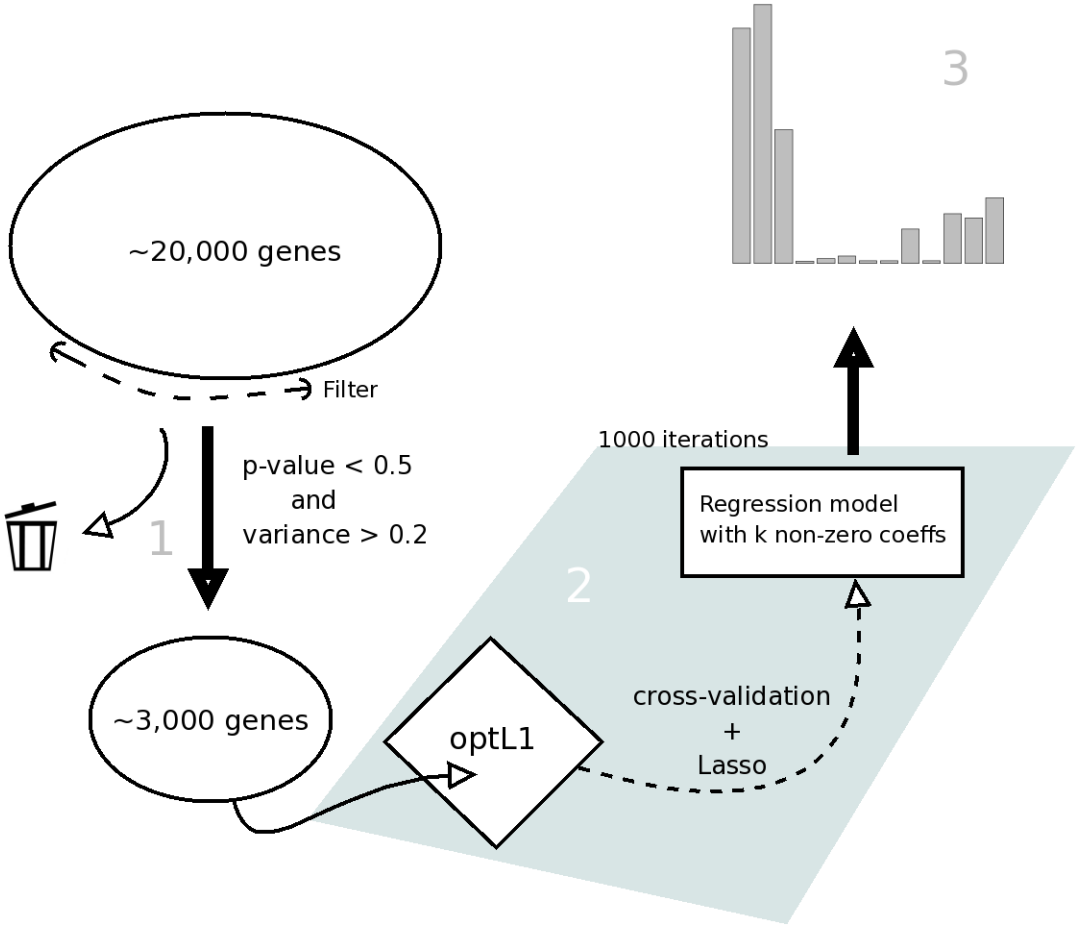

Figure 1.1: Flow of `optL1`. *1. The original gene set consisting of nearly 20,000 genes is filtered to include only those that vary the most amongst the individuals and with small Cox p-values. 2. These are then run through CV and the Lasso method 1000 times, taking note of the size of the supposedly relevant gene signature (group) each time. 3. The result is presented using a bar diagram.*

Variation in results. Ideally, there is just one solution which concludes with a suitable prediction model. However, for every new call to `optL1` the result varied, similarly to what is shown in Figure 1.2.

To obtain a conclusive answer, Sveen et al. repeated Lasso a thousand times and took note of the signature size (the number of coefficients different from zero) each time. Eight gene expression signatures of size zero to twelve showed up more than fifty times each. These signatures were then run on a first validation series; the five larger signatures were significant here. The 7-gene expression signature gave a prognosis closest to the actual survival outcome. Note that the smaller signatures were always contained in the larger ones. A third run on a second validation series verified that the 7-gene signature gave a reasonable prognosis.

Reproduction. In the attempt to reproduce the results, the same recipe was followed and nearly the same result was achieved. The seven genes that make up the prognostic gene expression signature in the paper, are the same seven that were left in the 7-gene signature in my calculations. There were fewer hits on the 7-gene signature, however. It was not the one with the most hits in the paper either, but the next steps in [1] were used to justify the result.

Unstable Lasso results permeate my calculations, as well as those performed by the paper's authors. Thus, in this thesis, the reasons for instability are attempted unraveled and an answer to the question "Is a result based on such Lasso calculations reliable?" is sought.

1.2 Chapter overview

Chapter 2 gives a general introduction to the required biological and statistical concepts. Lasso and CV have already been mentioned in the introduction, but the motivation for choosing these methods and the theory behind them will be given more attention there. Also, there is an introduction to the R packages `penalized` and `glmnet` which performs Lasso regularisation and CV, as well as other types of regularisation. **Chapter 3** discusses the methods of investigation and how to display the results. Data is simulated to mimic real data so that the properties of the data set can be controlled and the effect of these can be measured. Simulated data sets also ease the process of testing CV parameters on different data sets. **Chapter 4** presents the results from the investigations described in Chapter 3. Then a conclusion on the question "Is the algorithm described in Sveen et al.'s paper[1] reliable?" is approached. This is discussed in **Chapter 5**. In **Chapter 6** the results are further discussed and suggestions for further work are presented.

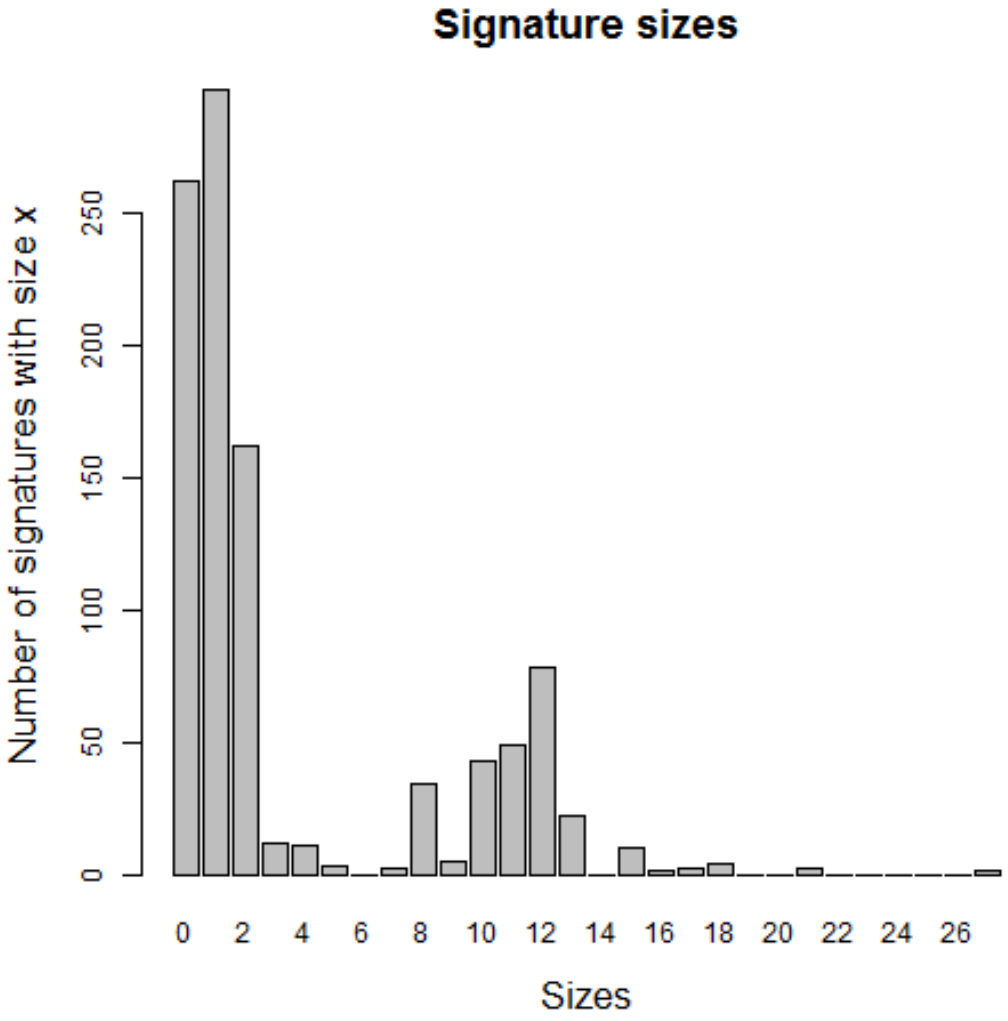

Figure 1.2: Signature size bar diagram. *The numbers along the x-axis are the sizes of the gene signatures (the number of genes included by Lasso). The height of each bar describes the number of times a signature with the given number of genes is found. The 0-, 1- and 2-signatures are the most frequent, but the 8-, 10- 11- and 12-signatures are also interesting. Judging from this diagram, obtained by the author (and deviating slightly from the one obtained by Sveen et al.[1]), the signatures of size 3 to 7, 9 and larger than 12 are less likely to be of any interest.*

Chapter 2

Background

Genes, gene expression, mRNA and other biological terms were mentioned in the introduction. In this chapter these terms are further explained. The statistical terms Lasso, Cox, cross-validation, etc. are also described.

2.1 Biological background

The genetic material in humans is stored as 46 chromosomes: 22 pairs of non-sex chromosomes (autosomes) and two sex chromosomes (XX for females and XY for males)[9], see example of a chromosome map in Figure 2.1. Each chromosome consists of a DNA molecule complexed with proteins and winds up to form chromatin. Most DNA, about 97% in humans, is non-coding[1]. The rest of the DNA is coding and form genes. A gene is a subset of the chromosome DNA string and is the recipe of (a part of) a protein. The DNA string includes nucleotides where each nucleotide includes one of the nucleobases guanine (G), adenine (A), thymine (T) and cytosine (C). A DNA string is recognised by its order of nucleobases.

Through protein synthesis, the gene is read and instructs the creation of a protein. If a gene is missing, some proteins may not be made, or if the gene has been altered from the original, a protein may function differently. In cancer, there is a fault that promotes non-normal growth or reduces a cell's ability to carry out suicide (apoptosis) when malfunctioning. Usually, cells multiply and die in balance, at varying speed in different part of your body. In fact, during a period of a few years your whole body will consist of new cells, with some exceptions such as the brain.

The translation of DNA to proteins happen via mRNA, messenger RNA. mRNA is the complementary string of a DNA string (gene), i.e. the mRNA have complementary nucleobases to the DNA string. The mRNA is the **gene**

[1]Non-coding DNA was earlier called "junk DNA" because it was unclear if it had any biological function. However this has been disproved, and the term is no longer used.

expression. Some genes provide many copies of mRNA, whereas other genes generate few. This can vary between individuals and also over time in a cell. The number of mRNAs from a gene is called the gene expression level.

A missing or incorrect gene is caused by a mutation. If the mutation causes the creation of a tumour, the mutation is tumourigenic.

2.1.1 Data structure

In a gene expression study involving p genes and n individuals, the gene expression levels may be expressed as an $n \times p$ matrix.

$$X = \begin{array}{r|cccc} & \text{gene 1} & \text{gene 2} & \dots & \text{gene } p \\ \text{patient 1} & 4400 & 340 & \dots & 40 \\ \text{patient 2} & 4450 & 2 & \dots & 41 \\ \vdots & & & & \\ \text{patient } n & 4349 & 358 & \dots & 43 \end{array} \tag{2.1}$$

The survival times of all observations may be expressed in a column vector:

$$(y_1, y_2, \dots, y_n)^T = (8, 3, \dots, 10)^T$$

For more information about survival times and other survival analysis terms see Section 2.2.3.

Suppose the patients should be divided into two or more groups, for example a high risk group and a low risk group. In the above example, gene 1 is expressed at about the same level in patients 1, 2 and n, and say this is the case for patients 2,3, ..., $n-1$ as well. Then the gene most likely has nothing to do with the response in interest. (It may have something to do with the survival time, but it is the differences between the patients that are interesting. That way the observations can be split into different groups which means that common factors can be ignored.)

The difference in gene 2 is interesting however. The gene is hardly expressed in patient 2, relative to patients 1 and n. It is now interesting to look upon:

- What is the normal gene expression level?
- Which patient(s) differ from this?
- Does it have anything to do with the response, or is the high/low gene expression level harmless? Say that patient 2 did not survive as long as the other patients. Can this be explained by the difference in gene 2?

Working with the CRC data[1], interesting gene expression levels are sought to see if they are linked to the response. Can genes that predict the survival time of the patient be found?

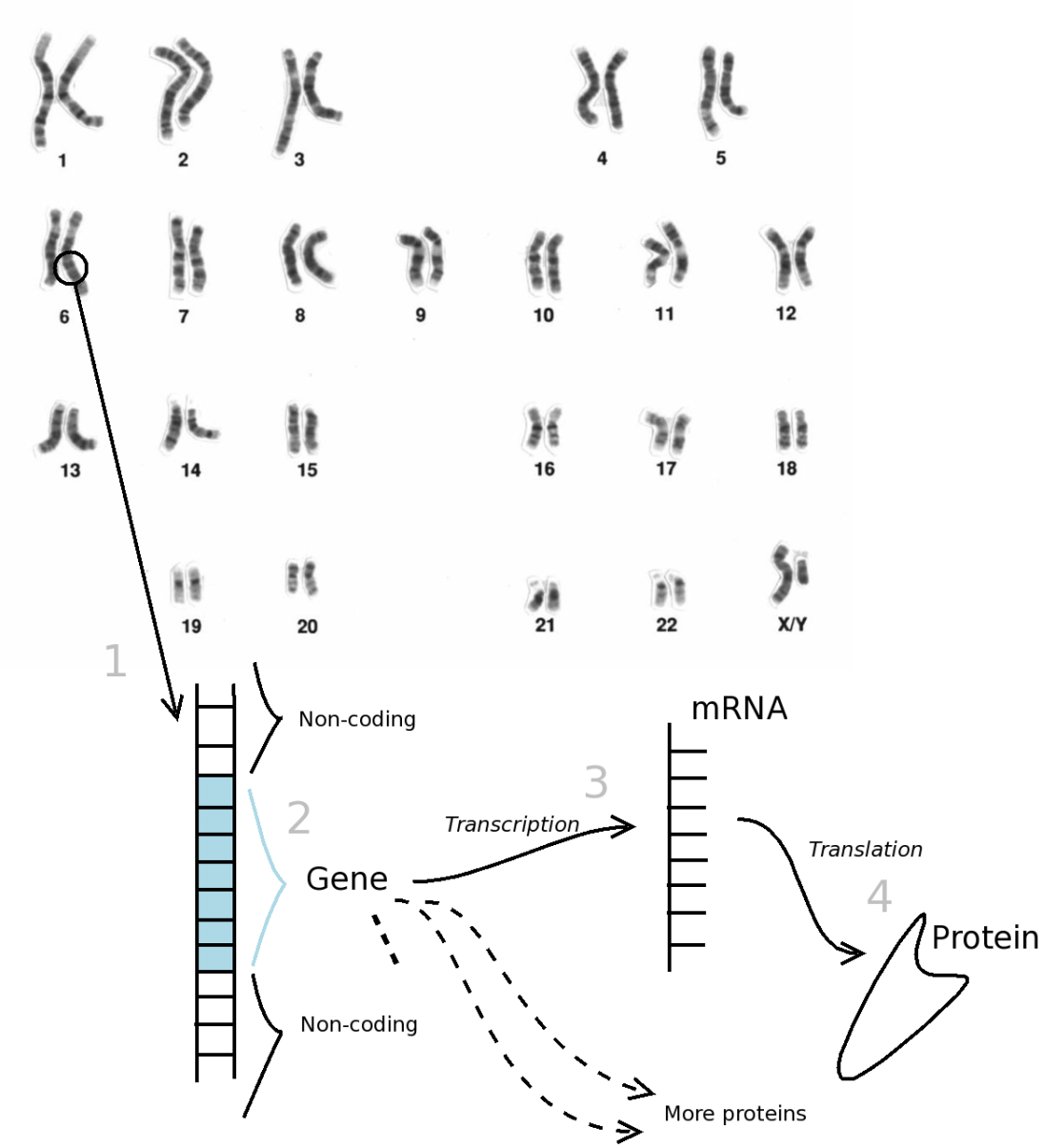

Figure 2.1: Sketch of the protein synthesis. *1. Each chromosome contains a long DNA string. 2. Some parts of the chromosome are called genes; they code for proteins. 3. The gene (DNA substring) is transcribed to mRNA, a complementary string. 4. The mRNA is translated to a protein consisting of a chain of amino acids. Chromosome map from [10].*

2.2 Statistical background

Survival analysis is the underlying theme of this study. In survival analysis the data set contains information about a set of patients: a number of covariates (in our case the expression level of one of more genes), an observation time and whether the observation time reflects time to a specific event (such as death) or time to censoring (e.g. due to the subject being alive at the end of the study period).

The challenge is to find out if there is a link between the covariates and survival, and if so find the link. In order to perform the analysis, a presentation of some statistical tools discussed follows in this section.

2.2.1 Introduction

A regression model is a way of explaining the connection between the covariates and the response: in our case the gene expression levels and survival time. A **linear regression model** looks like this

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p + \epsilon = \beta_0 + \boldsymbol{\beta}^T \mathbf{x} + \epsilon. \tag{2.2}$$

The y represents the response, e.g. some clinical variable measured on a continuous scale. The β_is are the weights of the explanatory variables, the xs, e.g. gene expression levels. Often, the β_0 is included in $\boldsymbol{\beta}$ by adding an element, 1, in $\mathbf{x}$, making $y = \boldsymbol{\beta}^T \mathbf{x} + \epsilon$. ϵ is the error, or the noise term. When there is more than one observation available, the β_is can be estimated using the least squares method.

2.2.2 The Lasso method

When dealing with genetic data there will often be information about many more genes than observations, i.e. $p >> n$. The problem of fitting a regression model such as the Cox model may then be compared to solving a system of equations with more unknowns than equations, which makes it possible to choose the values of the unknowns in infinitely different ways. If no precautions are taken, this process may result in a fit adapted to the noise instead of, or in addition to, the variables that are actually related to the response. The resulting model may be overfitted, and will be useless for the prediction in a general case[11]. For further motivation, see Section 2.2.7.

In order to settle on a few well-chosen variables, the explanatory variables that are irrelevant to the response should be removed. There are several ways to do this; some include shrinking the regression coefficients until they approach zero, while others actually reduce them to zero. For more information, see Section 2.2.6. Lasso is one of the latter methods.

Lasso

The Lasso (Least Absolute Shrinkage and Selection Operator) estimate for ordinary continuous data is defined by[12][13]:

$$\hat{\boldsymbol{\beta}} = \min_\beta \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2, \qquad \text{s.t.} \sum_{j=1}^{p} |\beta_j| \leq t, \tag{2.3}$$

where y_i is response i and x_{ij} is the jth covariate of observation i. The difference $y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j$ is actually the difference between the observed response y_i and the predicted response $\hat{y}_i = \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j$. So, the solution $\hat{\boldsymbol{\beta}}$ to the problem, is the $\boldsymbol{\beta}$ that minimises the error, under the constraint $\sum_{j=1}^{p} |\beta_j| \leq t$.

By choosing t very large, there will be no constraint at all and the fit will approach that of an ordinary linear regression model. But, by letting $t \to 0$ some of the coefficients β_j will move towards zero which is the intent. In the extreme case of $t = 0$, the β_js will all be forced to be zero, and the model has no variables with non-zero coefficients.

Notice that Lasso may be written in the Lagrangian form

$$\hat{\boldsymbol{\beta}} = \min_\beta \left(\frac{1}{2} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right). \tag{2.4}$$

The solution to this is equivalent to (2.3) by application of the Lagrange multiplier theorem[13]. Decreasing the t is the same as increasing the λ; $\lambda \to \infty \Leftrightarrow t = 0$ because a large λ penalty will lead the minimisation problem to choose $\beta_j = 0$ for all js such that the term $\lambda \sum_{j=1}^{p} |\beta_j|$ is as small as possible, zero. The problem, $\lambda = 0 \Leftrightarrow t \to \infty$, represents having no penalty at all. The term $\lambda \sum_{j=1}^{p} |\beta_j|$ is called the **L1 penalty**.

To find $\hat{\boldsymbol{\beta}}$, methods based on quadratic programming may be used[12]. They are iterative, and in every iteration a least squares problem is solved.

There exists variations of Lasso, such as adaptive Lasso. The adaptive Lasso method adds weights to the restrictions $\sum_{j=1}^{p} |\beta_j| \leq t$, so instead it becomes $\sum_{j=1}^{p} |\beta_j|\tau_j \leq t$, $\tau_j > 0$ for all j. The choice of weights is very important, and Zhang and Lu[14] propose using $\tau_j = 1/|\tilde{\beta}_j|$, which is just one of many possible choices. They motivate the choice by the consistency of the $\tilde{\beta}_j$s; the $\tilde{\beta}_j$s reflect the importance of the covariates.[2]

2.2.3 Survival analysis

Survival models in cancer usually describe the risk of death or relapse; from here on death will be used as an example, as in [1]. The **survival function** is the

[2] $|\beta_j|/|\tilde{\beta}_j| \to I(\beta_j \neq 0) = \{1 \text{ when } \beta_j \neq 0;\ 0 \text{ when } \beta_j = 0\}$ when $n \to \infty$.

risk of death occurring later than a given time t:

$$S(t) = P(T > t), \tag{2.5}$$

where T is a random variable representing time of death[15]. Note that the survival function S is decreasing, approaching zero as no person can live forever. However, survival models may be applied in other areas such as mechanical engineering where it makes more sense to talk about eternal life. Also, $S(0)$ is assumed to be 1, except if there is a chance of immediate event.

The **hazard rate** reflects the death intensity in a small time interval $[t, t+dt)$, given that the individual has survived up until the time t. Here dt is approaching zero, so the hazard rate represents the instant death rate. This function is often denoted λ:

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{P(t \leq T < t + \Delta t | T \geq t)}{\Delta t} \tag{2.6}$$

which can be rewritten by following[13]:

$$\begin{aligned} P(t \leq T < t + \Delta t | T \geq t) &= 1 - P(T \geq t + \Delta t | T \geq t) \\ &= 1 - \frac{P(T \geq t + \Delta t)}{P(T \geq t)} \\ &= 1 - \frac{S(t + \Delta t)}{S(t)} \end{aligned}$$

By putting both parts in the same fraction,

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{1}{\Delta t} \cdot \frac{S(t) - S(t + \Delta t)}{S(t)} = -\frac{1}{S(t)} \lim_{\Delta t \to 0} \frac{S(t + \Delta t) - S(t)}{\Delta t} = -\frac{S'(t)}{S(t)}, \tag{2.7}$$

where the definition of the derivative $S'(t) = \lim_{\Delta t \to 0}(S(t + \Delta t) - S(t))/\Delta t$ was applied in the last transition.

The relation between the hazard rate $\lambda(t)$ and the survival function $S(t)$ is useful. By the chain rule, we know that $\frac{d}{dt} \log S(t) = S'(t)/S(t)$, and

$$\lambda(t) = -\frac{S'(t)}{S(t)} = -\frac{d}{dt} \log S(t), \tag{2.8}$$

which implies that

$$S(t) = \exp(-\Lambda(t)), \qquad \Lambda(t) = \int_0^t \lambda(s) ds. \tag{2.9}$$

$\Lambda(t)$ is called the cumulative hazard and it follows from (2.8) and (2.9) that the relation between the cumulative hazard and the survival function is $\Lambda(t) = -\log S(t)$.

Another useful relation is that the lifetime distribution function $F(t)$ describes the negative survival function: $F(t) = 1 - S(t)$. By this, the density function is $f(t) = -S'(t)$. This is the foundation of an equivalent version of the hazard rate:

$$\lambda(t) = \frac{f(t)}{S(t)} \tag{2.10}$$

Survival models are based on survival data, i.e. information about a set of observations, usually patients. This includes a censor indicator which tells you if a patient has been withdrawn from the experiment for some reason. Death by other causes than the illness of interest, a patient failing to turn up for check-ups or voluntarily withdrawing from the experiment may all be causes for censoring. Usually the events are denoted with ones and zeros: A one represents event (death) and a zero represents censoring.

In addition to this information, there usually is information about gender, age, gene expression levels, as described in Section 2.1, and the time until withdrawal or event (death), called survival time. The survival time is usually the time from the entering of the patient in the study until event or censoring. A patient may be entered in a study at the time the tumour was discovered. Other survival times, such as age, may also be applied.

2.2.4 The Cox model

The most common regression model for survival data is the **Cox model** because it makes no assumptions about the baseline hazard $\lambda_0(t)$, it is semi-parametric, as described in this section. According to this model, the hazard function is

$$\lambda(t|x_1, \ldots, x_p) = \lambda_0(t) \exp(\beta_1 x_1 + \ldots + \beta_p x_p)$$

or

$$\lambda(t|\mathbf{x}) = \lambda_0(t) \exp(\boldsymbol{\beta}^T \mathbf{x}). \tag{2.11}$$

Here, $\mathbf{x} = (x_1, \ldots, x_p)$ are the p covariates values for each individual. $\lambda_0(t)$ is the **baseline hazard**. Notice that if $\boldsymbol{\beta} = \mathbf{0}$, then $\lambda(t|\mathbf{x}) = \lambda_0(t)$. Consider two individuals with covariate vectors $\mathbf{x_a}$ and $\mathbf{x_b}$. Then

$$\frac{\lambda(t|\mathbf{x_a})}{\lambda(t|\mathbf{x_b})} = \frac{\lambda_0(t) \exp(\boldsymbol{\beta}^T \mathbf{x_a})}{\lambda_0(t) \exp(\boldsymbol{\beta}^T \mathbf{x_b})} = \exp\left(\boldsymbol{\beta}^T (\mathbf{x_a} - \mathbf{x_b})\right). \tag{2.12}$$

Thus the ratio of the hazards of two individuals is independent of time. Suppose the ith covariate increases by 1 and all other covariates are identical in the two individuals, i.e. $x_{a,i} = x_{b,i} + 1$ and $x_{a,k} = x_{b,k}$ for $k \neq i$. Then, by inserting into (2.12):

$$\frac{\lambda(t|\mathbf{x_a})}{\lambda(t|\mathbf{x_b})} = \exp(\beta_i), \tag{2.13}$$

which is called the **hazard ratio** for the ith covariate.

The partial likelihood

It is a complicated task to estimate the regression coefficients $\boldsymbol{\beta}$ from the hazard function (2.11) as it is a function of more than one parameter. Cox observed that with the independence of time it is possible to estimate $\boldsymbol{\beta}$ without modelling the hazard function, and instead maximise the **partial likelihood**[16]. This approach is called **Cox proportional hazards model**[15].

The partial likelihood is a function of $\boldsymbol{\beta}$:

$$L(\boldsymbol{\beta}) = \prod_{i:C_i=1} \frac{\theta_i}{\sum_{j:Y_j \geq Y_i} \theta_j}, \quad \theta_k = \exp(\boldsymbol{\beta}^T \mathbf{x}_k). \tag{2.14}$$

where $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are the covariate vectors (column vectors) for the n observations, each consisting of p covariates. Y_k is the observed time for the kth individual and is censored if $C_k = 0$. $C_i = 1$ means that individual i is non-censored, i.e. an event has taken place. The notation $\delta_i \in \{0, 1\}$ is also common for censoring status.

The partial likelihood is, in words: For all non-censored individuals i, multiply the hazard ratio (for i) divided by the sum of the hazard ratios of all individuals still alive at the time of event i (both censored and non-censored).

The log partial likelihood is given by

$$l(\boldsymbol{\beta}) = \sum_{i:C_i=1} (\boldsymbol{\beta}^T \mathbf{x}_i - \log \sum_{j:Y_j \geq Y_i} \theta_j) = \sum_{i:C_i=1} \varphi_i(\boldsymbol{\beta}), \tag{2.15}$$

which we may maximise to find the maximiser of (2.14).

2.2.5 Lasso and the Cox model

So far, the general linear regression case of Lasso has been described. In Section 2.2.4, the Cox model was introduced which may also be combined with Lasso.

Tibshirani[16] proposes to estimate $\hat{\boldsymbol{\beta}}$ by

$$\hat{\boldsymbol{\beta}} = \text{argmax}_\beta \; l(\boldsymbol{\beta}), \qquad \text{s.t.} \sum_{j=1}^{p} |\beta_j| \leq t, \tag{2.16}$$

where $l(\boldsymbol{\beta})$ is the log partial likelihood given in (2.15). To find $\hat{\boldsymbol{\beta}}$, Tibshirani[16] proposes to first fix t and initialise $\hat{\boldsymbol{\beta}} = 0$, then minimise $(\mathbf{z} - \boldsymbol{\eta})^T \mathbf{A} (\mathbf{z} - \boldsymbol{\eta})$, where $\mathbf{z}$, $\boldsymbol{\eta}$ and $\mathbf{A}$ are based on our existing knowledge of $\mathbf{X}$, $\boldsymbol{\beta} = \hat{\boldsymbol{\beta}}$ and l, subject to $\sum_{j=1}^{p} |\beta_j| \leq t$ through a quadratic programming procedure. Repeat the minimisation until $\hat{\boldsymbol{\beta}}$ converges.

Tibshirani was not the first person to consider quadratic programming as a solution to such a problem, or even representing the problem with an L1 penalty

(Lasso). The first use of linear programming was by Leonid Kantorovich in 1939 who used it to plan effective moves in World War II.

A basic linear programming problem can look like this:

$$\max f(x,y) = \max(2x+3y), \quad \text{s.t. } g(x,y) = 4x + 0.5y \leq 10, \quad x,y \geq 0. \tag{2.17}$$

This problem can be solved by the simplex algorithm.

Application of the Lagrange multiplier theorem, as mentioned in Section 2.2.2, can transform the problem into an equivalent problem:

$$\Lambda(x,y,\lambda) = f(x,y) + \lambda \cdot (g(x,y) - c) = 2x + 3y + \lambda(4x + 0.5y - 10) \tag{2.18}$$

The minimisation points (x,y) of this problem give the solution to (2.17). Note that this is similar to the Lasso formulae in (2.3) and (2.4). The Lasso problem forms a quadratic programming problem instead, because the function in (2.16) is quadratic.

2.2.6 Other regularisation methods

The Lasso is not always the best choice. In a situation where variables are correlated within groups, Lasso tends to pick one of these and discard the rest. A proposed zero-coefficient therefore does not necessarily mean that the variable is irrelevant; it could just be correlated with another variable which has a non-zero coefficient.

In the case where $p > n$, i.e. the number of genes is greater than the number of observations, Lasso includes a maximum of n (number of observations) variables. Zou and Hastie[17] argue that this is a limiting feature of the method.

There are other methods that avoid these issues, such as ridge regression[18] and the elastic net[17].

Ridge regression

Ridge regression maintains the number of variables in the model, it just shrinks the coefficients. All the variables are included in the final model. However, it is not necessarily easy to see which are more relevant to the survival response. The coefficient sizes may say something, but must be seen in relation to the covariate value. A large coefficient combined with a small covariate may make only a small contribution to the overall predicted effect and vice versa.

The corresponding expression to (2.4) is:

$$\hat{\boldsymbol{\beta}} = \min_\beta \left(\frac{1}{2} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right) \tag{2.19}$$

The term $\lambda \sum_{j=1}^{p} \beta_j^2$ is called the **L2 penalty**.

Elastic net

The elastic net is one attempt to combine the L1 and L2 penalties of Lasso and ridge regression. If some variables are correlated, more variables than in Lasso will tend to be included in the final suggested fit. Still, like Lasso, it reduces the number of variables in the model[17].

(Naïve) elastic net is defined by:

$$\hat{\boldsymbol{\beta}} = \min_\beta \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2, \qquad \text{s.t. } (1-\alpha)\sum_{j=1}^{p} |\beta_j| + \alpha \sum_{j=1}^{p} \beta_j^2 \leq t, \tag{2.20}$$

where α is a constant defined by the restrictions laid upon the penalties defined by $\alpha = \lambda_2/(\lambda_1 + \lambda_2)$. λ_1 is the Lasso penalty parameter and λ_2 is the ridge penalty parameter. Leaving one of them zero leads to the other regularisation method:

- $\lambda_2 = 0 \Rightarrow \alpha = 0 \Rightarrow$ Lasso
- $\lambda_1 = 0 \Rightarrow \alpha = 1 \Rightarrow$ Ridge regression.

Comparison to (2.3) shows the similarity to Lasso and ridge regression.

The naïve elastic net performs poorly when it differs greatly from either Lasso or ridge regression. To eliminate this disadvantage, a suggestion is to scale $\hat{\boldsymbol{\beta}}$ by a factor[17].

2.2.7 Cross-validation

When fitting a model on a training set, we wish to minimise the difference between the prediction and the actual observation, i.e. $y_i - \hat{y}_i$. One way to do this would be to minimise the mean squared error (MSE):

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \frac{1}{n}\sum_{i=1}^{n} e_i^2, \qquad e_i = y_i - \hat{y}_i$$

In a situation where $p >> n$, however, the MSE would be zero for several different values of $\boldsymbol{\beta}$.

Such a procedure would be less suitable for any other data. Adding a penalty, such as Lasso (see Section 2.2.2), will restrict the model. However, we can just choose $\lambda = 0$ in (2.4), and we are left with exactly the same problem as before.

Even if we did not have a possibly overfitted model, training the model on a specific data set may make it a poor model for a general case. Having a separate test set would be beneficial to test the model's general usefulness. That way, we

could set the model parameter λ to a specific positive value, leading to $f(\lambda, \mathbf{x}_i)$, try it out on a test set by estimating the MSE

$$MSE = \frac{1}{n}\sum_{j=1}^{n}(f(\lambda, \mathbf{x}_j) - Y_j)^2, \tag{2.21}$$

where $(\mathbf{x}_j, Y_j)$ are the observation pairs in the test set. Set λ to a new value, make a new model, run this in the test set, find the MSE. Iterate this until the λ that gives the minimum of MSE has been identified.

The aim of this is not to use the same data that we trained the model on to test it. Unfortunately, the general solution is not yet obvious. The test set may not be representative for a general case, in fact usually they are not. And just optimising the model on the one data set, would make it suitable for the test set, but not generally. The dream scenario would be to have infinitely many different data sets to test the model on and find the one that fits most. With that information, the **predicted squared error** (PSE) could be calculated. PSE is the error of the model tested on future observations. Then we would have a model that performs well on any data set.

Having infinitely many data sets is impossible. Instead, we can use the one data set available to both train and test the model in several iterations, so called **cross-validation** (CV).

The CV process

An illustration of the CV flow can be found in Figure 2.2. The general idea of CV is to separate m observations from the data set, make a model based on the remaining $n - m$ and test it on the m observations outside the data set. The left-out group of size m is called the **test set**, while the remaining group of size $n - m$ is called the **training set**. Repeat this such that all observations are used once as the test set.

Say $m = 1$, which means there are $K = n$ groups of size 1. This is called **leave-one-out cross-validation** (LOOCV) . (Later, K-fold CV will be introduced.) Say we have a penalised model that depends on a parameter λ:

$$f(\lambda, \mathbf{x}_i) = \boldsymbol{\beta}(\lambda)^T \mathbf{x}_i, \tag{2.22}$$

which is similar to the regression model presented in Section 2.2.1. CV can be used to find the λ that, inserted in f, minimises the CV error,

$$CV(\lambda; X, \mathbf{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - f^{(-i)}(\lambda, \mathbf{x}_i))^2. \tag{2.23}$$

$f^{(-i)}$ is the model trained on $\mathbf{x}_j$ for $j = 1, \ldots, n$ except for i. Fix λ, leave out observation i, train a model on the remaining data, and test on i. Do this for

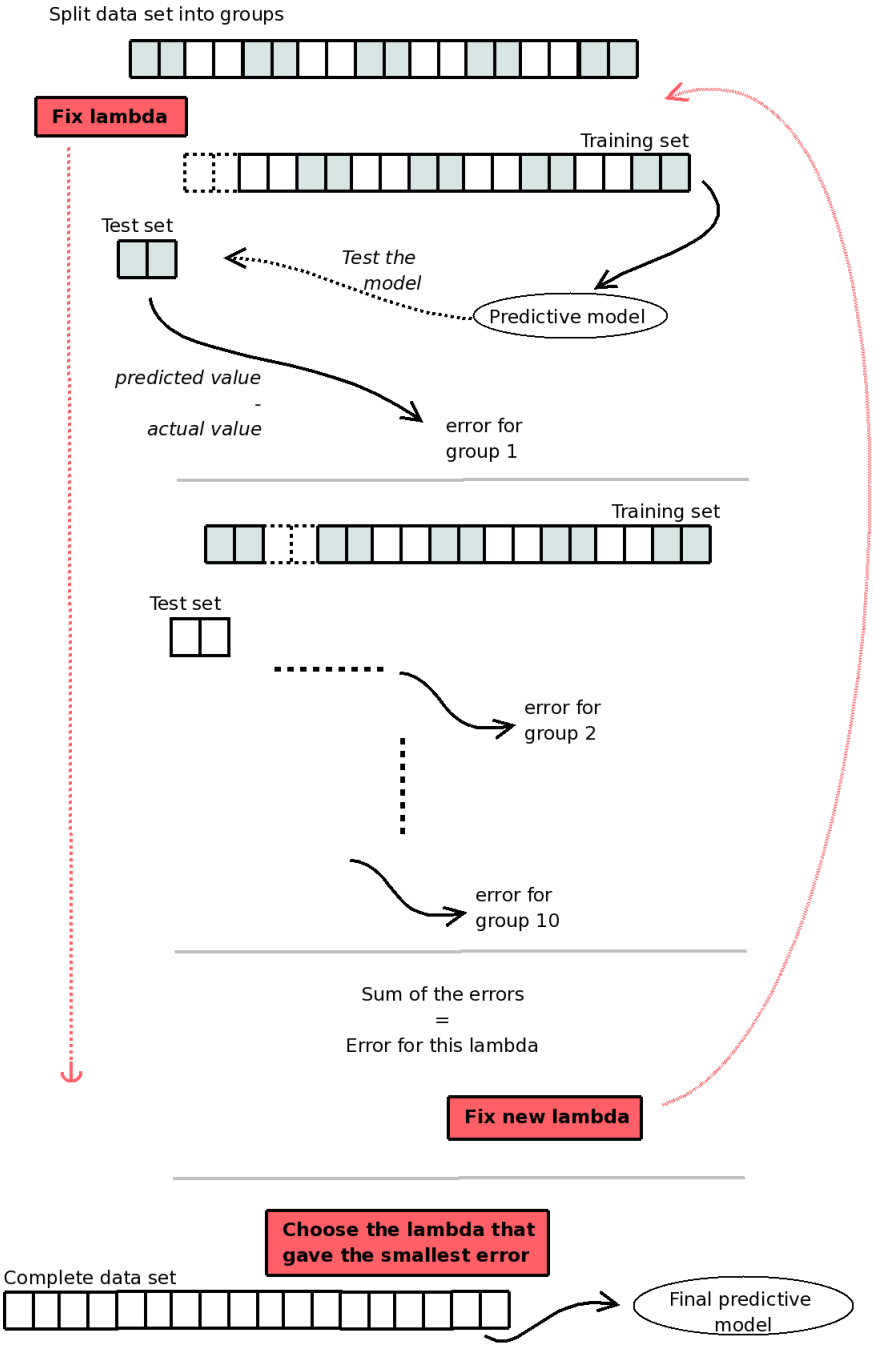

Figure 2.2: Flow of CV. *First, the data set is split into groups, then a λ is chosen. One group is marked as the test set, the other groups as the training set. A model based on the training set is used to predict the values in the test set and an error is recorded. Do this until all groups have been used as the test group and find the average error for this λ. Fix a new λ and repeat. Finally, choose the λ that gives the smallest error.*

every $i = 1, \ldots, n$ and sum up the error term and find the average error. Try a new value for λ and repeat.

By calculating CV for several values of λ, we can choose the λ that gives the lowest value of CV, and find the model $f(\lambda^{(-i)}, X)$ based on data from all observations.

An important property of $CV(\lambda)$ is that

$$E(CV(\lambda)) \approx PSE(\lambda), \tag{2.24}$$

where PSE is the difference between the model and the unobservable real model. In words, the expected shape of the curve of $CV(\lambda)$ is approximately equal to the PSE, see Figure 2.3. $CV(\lambda)$ is easy to calculate, even though $PSE(\lambda)$ is not. By performing CV, we find a good estimate of how the predicted error behaves.

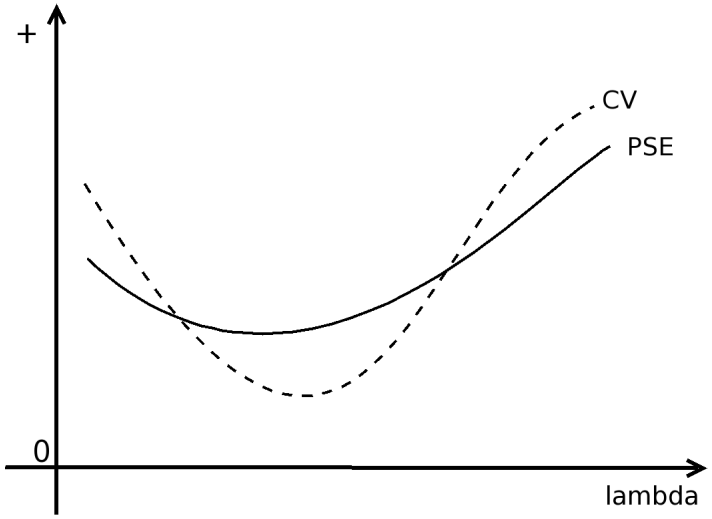

Figure 2.3: Example of how CV and PSE may behave*, according to (2.24). The expected shape of CV is approximately the shape of PSE.*

K-fold cross-validation

Choosing $K = n$, LOOCV, can be time consuming for large n and if there are many λ_k to optimise λ. Instead, it is common to choose e.g. $K = 10$, such that for every iteration, we remove $m \approx n/K$ individuals instead of just one. Only K iterations are needed to calculate $CV(\lambda; X, \mathbf{y})$, instead of n. This will reduce the run time, and lower the variance of the prediction errors. LOOCV will result in higher variance because the fits are trained on n nearly equal data sets[11]. For time measurements, see Section 3.1, especially Figure 3.1.

If the sub data set used as test data is very small, the risk of having a disproportional number of censored events present is greater, i.e. a test data set with nearly none events is a nonrepresentable test set. How the censorings affect the

model depends on what kind of CV is performed, see the list of alternatives in Section 2.2.8. This can cause the wide range of suggested signatures in K-fold CV where K is large.

2.2.8 Cross-validation and the Cox model

So far, CV has been introduced in combination with the linear regression model. Now, we will look at Cox regression. Recall (2.15), the Cox partial log likelihood is defined by

$$l(\boldsymbol{\beta}) = \sum_{i:C_i=1} (\boldsymbol{\beta}^T \mathbf{x}_i - \log \sum_{j:Y_j \geq Y_i} \theta_j) = \sum_{i:C_i=1} \varphi_i(\boldsymbol{\beta}), \tag{2.25}$$

where $\theta_j = \exp(\boldsymbol{\beta}^T \mathbf{x}_j)$. With Lasso penalty, it is

$$l_{pen}(\boldsymbol{\beta}, \lambda) = \sum_{i:C_i=1} \varphi_i(\boldsymbol{\beta}) + \lambda \sum_{j=1}^{p} |\beta_j| \tag{2.26}$$

For more information, see Section 2.2.2. The regression model f is now replaced with the partial likelihood l, with penalised likelihood l_{pen}.

The purpose of using CV to find the best fitting Cox model, is to find the optimal model parameter λ, as for linear regression. Usually, as in the case of (2.22), the likelihood contribution of the i'th individual is independent of the other individuals (when $i \neq j$). So, when cross-validating, it is easy to make the expression $f^{(-i)}$ where observation i is excluded, see (2.23). But, because part two of $\varphi_i(\boldsymbol{\beta})$, i.e. $\log \sum_{j:Y_j \geq Y_i} \theta_j = \log \sum_{j:Y_j \geq Y_i} \exp(\boldsymbol{\beta}^T \mathbf{x}_j)$, depends on information about other observations than itself (those in the still-at-risk group), we cannot use CV directly as described earlier where each observation is removed.

Three variations of CV for the Cox model are described below. Every variation maintains a quality of general CV, at the cost of others. Common for all CV variations are that the λ that maximises the cross-validated likelihood (CVL) is the λ that gives the best model.

1. Kuk[19] presented a type which is based on changing the status of one observation i from non-censored to censored, which is done for each non-censored observation. Changing the event indicator is the corresponding action to removing observation i from the set as in ordinary CV. This means that individual i affects the risk of all individuals still at risk until the time of event i (which is no longer considered an event).

$$\text{CVL} = \sum_{i:C_i=1} \varphi_i \left(\hat{\boldsymbol{\beta}}(\lambda)^{(i)} \right), \tag{2.27}$$

   where $\hat{\boldsymbol{\beta}}(\lambda)^{(i)}$ is estimated by maximising the penalised partial likelihood (2.26) for a given parameter λ based on the data set where censor indicator i

has been changed from non-censored to censored[11]. This method is similar to ordinary CV for a general likelihood model. The fact that individual i was alive up until t_i is used, but not the fact that he/she dies.

2. Another option was presented by Verweij and Van Houwelingen[20]. This takes into account that the components of the partial likelihoods affect each other. The effect of observation i is defined as $l_i(\boldsymbol{\beta}) = l(\boldsymbol{\beta}) - l_{-i}(\boldsymbol{\beta})$ where $l_{-i}(\boldsymbol{\beta})$ is the log-likelihood when observation i is left out. Linked to this, the $\hat{\boldsymbol{\beta}}$ that maximises $l_{-i}(\boldsymbol{\beta})$ is denoted $\hat{\boldsymbol{\beta}}^{(-i)}$. The CVL is defined by $\sum_{i=1}^{n} l_i(\hat{\boldsymbol{\beta}}^{(-i)})$, and considering the penalised likelihood:

$$\text{CVL} = \sum_{i=1}^{n} l_i\left(\hat{\boldsymbol{\beta}}(\lambda)^{(-i)}\right), \tag{2.28}$$

where $\hat{\boldsymbol{\beta}}(\lambda)^{(i)}$ is estimated by maximising the penalised partial likelihood (2.26) for a given parameter λ.

3. One version was presented in Hege L. Størvold's master thesis[11]. This is based on the idea of general CV. We ignore that the contribution of one observation may affect the likelihood of another, so for every calculation of the likelihood in the loops of the CV, observation i is removed entirely, as if it did not exist at all.

$$\text{CVL} = \sum_{i:C_i=1} \varphi_i\left(\hat{\boldsymbol{\beta}}(\lambda)^{(i)}\right) \tag{2.29}$$

This is a hybrid of the previous CV variations; the formula is similar to Kuk's proposal[19], but the $\hat{\boldsymbol{\beta}}(\lambda)^{(i)}$ is calculated as in Verweij and Van Houwelingen's proposal[20] where $\hat{\boldsymbol{\beta}}(\lambda)^{(i)}$ is estimated by maximising the penalised partial likelihood (2.26) for a given parameter λ based on the data set where observation i is left out. Maximisation of the penalised log-likelihood is a common way to find the best λ[21].

In summary, CV for the Cox model may ignore that the observations may affect the likelihood of each other (3), a censored observation may be treated as non-censored up until event (1) or we can consider that the observations may affect the likelihood of each other (2).

2.3 Computations in R

There are several R packages that perform the statistics described in Section 2.2. The package `penalized`[22] is used in the CRC paper[1] and is the package mainly used in this study. In addition, the package `glmnet` has been tested to some degree.

2.3.1 `penalized`

The function `optL1` from `penalized` is the one used by Sveen et al.[1]. It takes input including a Survival object (time and event) and covariates (X matrix, gene expression levels), and optionally an interval for the value of λ in (2.4) and K for K-fold CV. A typical call looks like this:

```
opt <- optL1(Surv(time, event), penalized=genes, fold
   =10)
```

The `genes` matrix consists of elements where each row represents an observation and each column represents a gene, as in (2.1). The `fold` parameter, i.e. K, decides how many groups the data set should be split into in the CV process. Recall the step by step algorithm from Section 1.1. If `fold` is set to n, the number of observations, it will result in LOOCV. In the CV the data set is split into n groups of size 1.

Output

`optL1` suggests a regression model and the coefficients for the covariates can be found by:

```
coeffs <- coefficients(opt$fullfit)
```

These coefficients are the β_is from Sections 2.2.2 and 2.2.5 and describes the weights of the genes. Only the non-zero coeffcients are included in the final model, and obviously then `length(coeffs)` will give the number of genes supposedly relevant to the response.

Challenges with `optL1`

As mentioned in the documentation[22], it is not certain that `optL1` always manages to pick the λ that gives *global* minimum error:

> The `optL1` [...] functions use Brent's algorithm for minimization without derivatives [...]. There is a risk that these functions converge to a local instead of a global optimum. This is especially the case for `optL1`, as the cross-validated likelihood as a function of `lambda1` quite often has local optima. It is recommended to use `optL1` in combination with `profL1` to check whether `optL1` has converged to the right optimum.

We can take a look at the CVL path by applying the function `profL1`:

```
prof <- profL1(Surv(time, event), penalized=genes, step
   =20, fold=opt$fold)
```

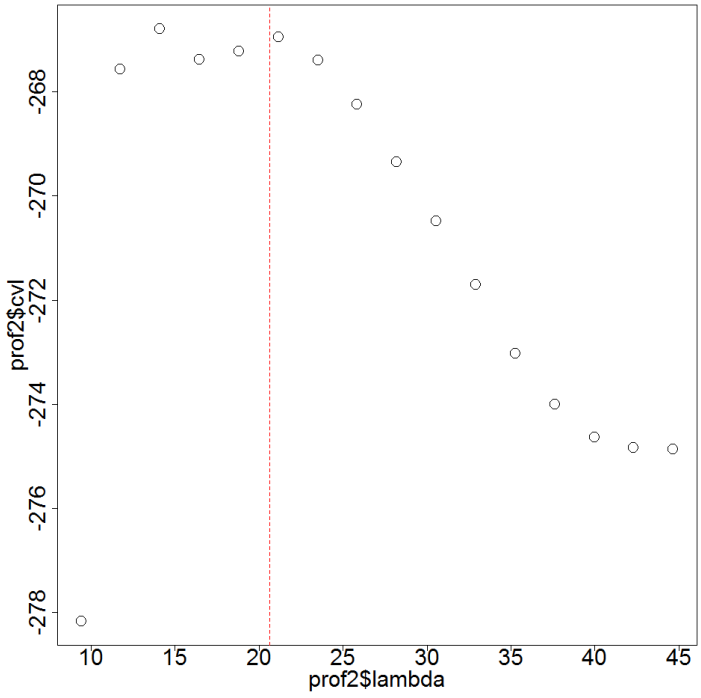

Figure 2.4: CVL plot from profL1 function. *The plot shows the CVL value for increasing values of λ. This should be as close to zero as possible, i.e. the maximum which can be found at $\lambda \approx 15$. The red line marks* `optL1`*'s choice of λ.*

The function takes nearly the same parameters as `optL1`, but to be able to see the previous object `opt`'s path, the exact same folding must be used. Even if we call the function with the same number of folds, the grouping varies from time to time, so to avoid this we need to specify `fold=opt$fold`.

`profL1` shows the steps of the optimisation. A greater value of the parameter `step` will show more detail. It is interesting to look at the plots of `profL1` called by:

```
plot(prof$lambda, prof$cvl)
plotpath(prof$fullfit)
```

These calls give plots shown in Figures 2.4 and 2.5, respectively.

The maximum of the CVL plot in Figure 2.4 gives the optimal λ, which is about 15 in this particular case. By tracing the λ over to the plot to the right, taking notice of how many lines present at `lambda1=15`, we can see that many, more than ten, coefficients are non-zero.

The plot in Figure 2.5 is a nice way to illustrate the effect of the model parameter λ. As λ decreases the number of non-zero coefficients increases. In the extreme case of $\lambda = 0$, there would be no constrain on the penalised ex-

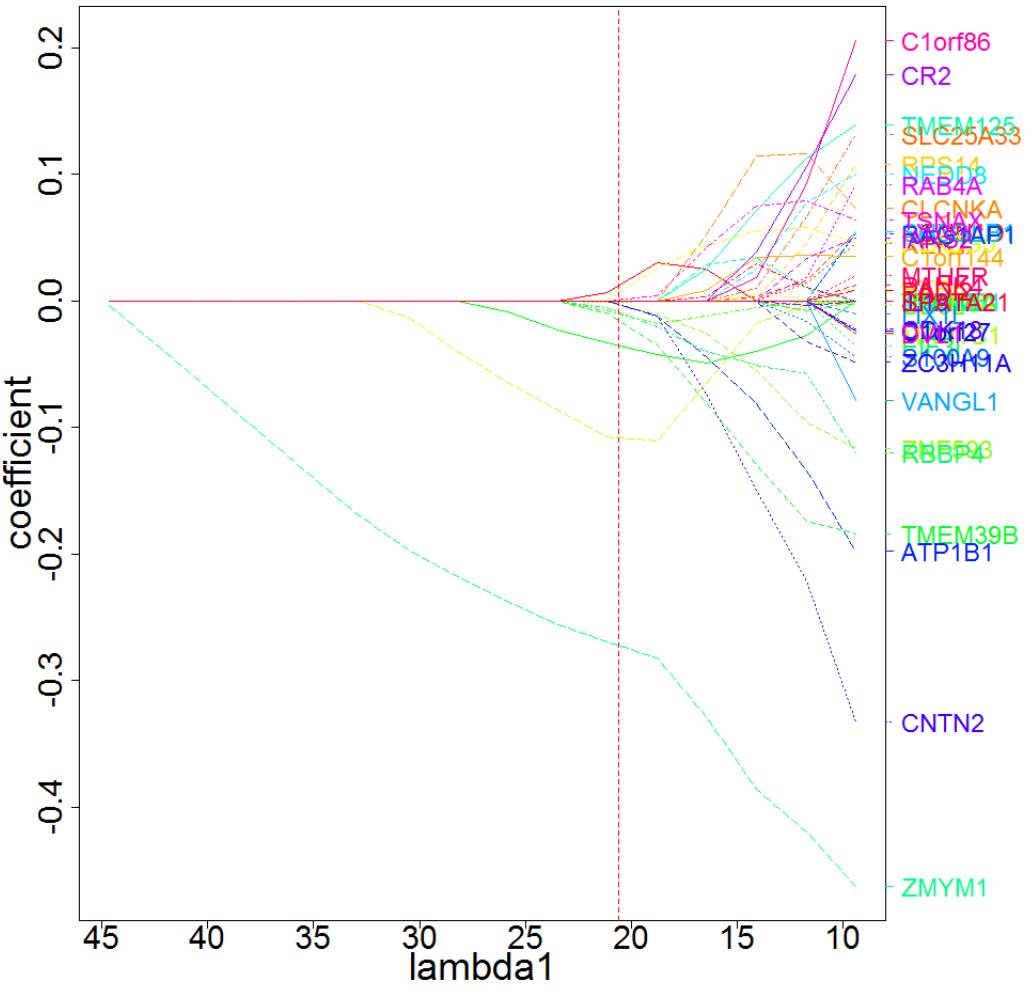

Figure 2.5: Path plot from `profL1` function. *The λ that gave the maximum value from the plot in Figure 2.4, about $\lambda = 15$, represents a large gene signature. The red line marks `optL1`'s choice of λ which results in a gene signature of size 9.*

pression $\hat{\boldsymbol{\beta}} = \text{argmin}_\beta \frac{1}{2} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$ (see Section 2.2.2) making it unpenalised. By making λ greater, it affects the model more and more and eventually the model would have no non-zero coefficients.

Investigation of the optimisation path can detect errors where a local minimum error (the value marked with the red line in Figure 2.4) has given the suggested λ instead of the λ with the global minimum error. We may correct the call to `optL1` by

```
# Find the lambda that gives global minimum error
prlambda <- prof$lambda[match(max(prof$cvl), prof$cvl)]
opt <- optL1(Surv(time, event), penalized=genes, fold
   =10, minlambda1=(prlambda-1), maxlambda1=(prlambda
   +1))
```

This will force the λ to stay in a close interval around the optimal λ found by `profL1`.

2.3.2 glmnet

`glmnet` uses an algorithm called coordinate descent to find the λ that minimises the error. This algorithm is faster than `penalized`, which uses the full gradient method in combination with Newton Raphson.

The application of `glmnet` is similar to `penalized`[23].

```
cv <- cv.glmnet(genes, Surv(time, event), nfolds=10,
   alpha=1, family="cox") # Cross-validation
fit <- glmnet(genes, Surv(time, event), alpha=1, family
   ="cox") # Model
```

The CV finds an optimal λ:

```
> print(cv$lambda.min)
[1] 0.2221717
```

This value of λ can be traced to the corresponding value on the x-axis in Figure 2.6, marked by the vertical left line. As can be seen in the plot (at the top), the number of non-zero coefficients in the model is two. This is confirmed by:

```
> coeffs_vec <- as.matrix(coef(fit, s = cv$lambda.min))
> index <- which(coeffs_vec != 0)
> coeffs <- coeffs_vec[index]
> print(index)
[1] 112 199
> print(coeffs)
[1] -0.1114382 -0.2391665
```

The genes are represented in the 112th and 199th columns in the gene expression matrix `genes` and the coefficient estimates are $\beta_{112} \approx -0.11$ and $\beta_{199} \approx -0.24$.

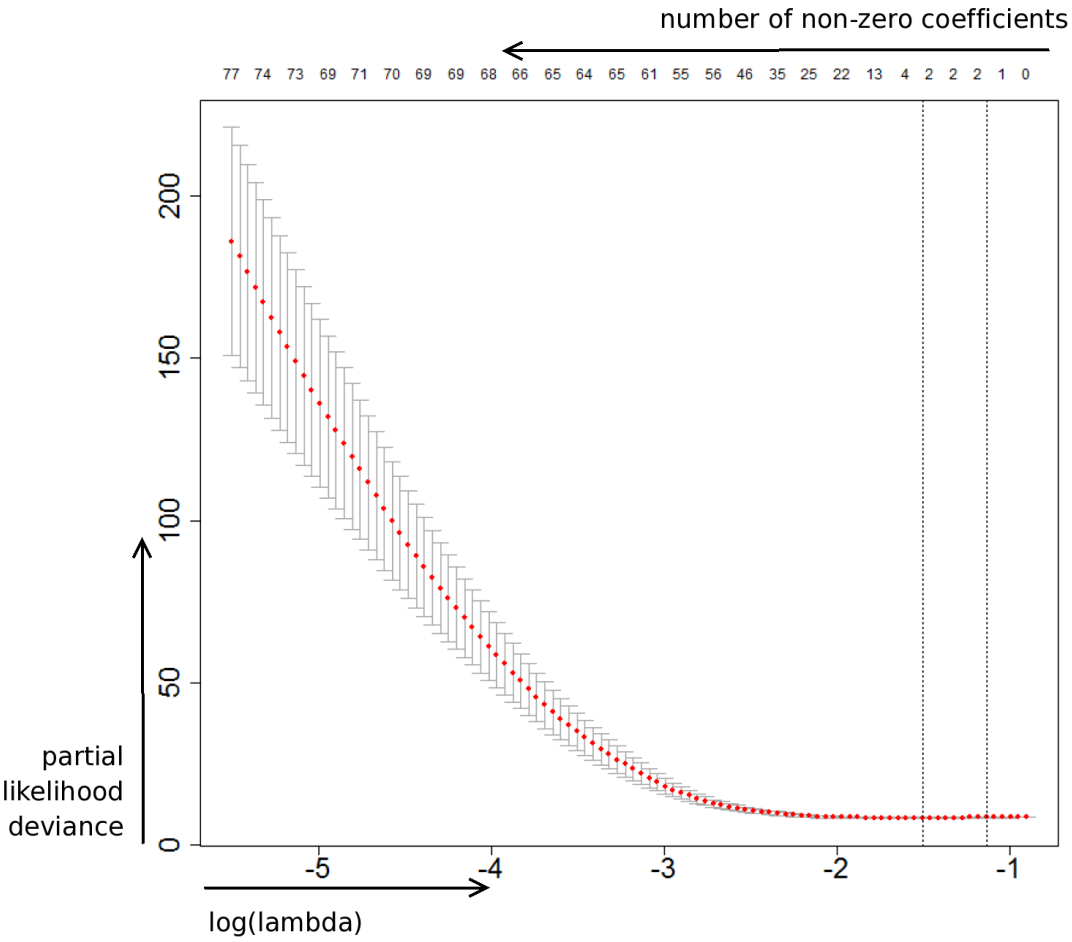

Figure 2.6: CV plot of the `glmnet` model. *The left vertical line marks the minimum point with x-value corresponding to the optimal λ. The right line marks the model which is one standard deviation away from model that gives the minimum error. The number of non-zero coefficients in the model is shown at the top of the plot.*

2.3.3 Differences

Both `glmnet` and `penalized` base their calculations on generalised linear models (GLM) and they both maximise the penalised log-likelihood described in Section 2.2.8 where λ is the penalty parameter corresponding to `lambda` in R.

The main difference described in the documentation[22][24] between the two methods is how they estimate the fits, i.e. $\hat{\boldsymbol{\beta}}$, given λ. `penalized` applies the full gradient algorithm and Newton Raphson when the optimal value is approached. `glmnet` applies coordinate descent which is much faster. Also, the CV is different, but it was difficult to find documentation of this without thoroughly reading the source code.

The computations in this study show that the two R packages do not always give the same results, see Section 4.5.3, but it is difficult to understand why from the documentation. Even though λ is defined as the penalty parameter in the penalised log-likelihood in both packages, the values are very different from each other as the examples above have shown. This is also discussed in Section 4.5.3.

2.4 Recent work

Xu et al.[25] write that

> A sparse algorithm cannot be stable and vice versa. [...] In particular, our general result implies that L1-regularised regression (Lasso) cannot be stable, while L2-regularised regression [Ridge] is known to have strong stability properties and is therefore not sparse.[25]

Here, stability describes how the result of an algorithm varies from one data set to another data set which is nearly identical to the first. The instability discussed in this thesis instead describes how the fold in CV affects the result of Lasso, so called model-selection variability. This topic has been discussed[26][27]. Roberts[26] proposes to apply a method called percentile-Lasso to overcome this instability, which he reasons with an illustration similar to Figure 4.17, which is more thouroughly discussed in Section 4.5.1.

Percentile-Lasso

This section is based entirely on [26]. CV's choice of λ seems to be somewhat small compared to the number of real relevant covariates included in the prediction model. A larger λ would give fewer non-zero coefficients, and mostly it would be the coefficients of the irrelevant genes that would be reduced to zero first. Percentile-Lasso can be a useful guide in the choice of λ. The algorithm is as follows:

1. Perform K-fold CV and Lasso. Take note of the choice of λ, denoted $\hat{\lambda}_1$.

2. Repeat step 1 N times s.t. a vector $\Lambda(N) = (\hat{\lambda}_1, \hat{\lambda}_2, \ldots, \hat{\lambda}_N)$ is obtained.
3. Find the θth largest value of $\Lambda(N)$, denoted $\hat{\lambda}(\theta)$.
4. Set $\lambda = \hat{\lambda}(\theta)$.

An algorithm of how to choose θ is discussed in Roberts' paper, but a typical choice would be $\theta \geq 0.75 \cdot N$, i.e. the 75-percentile or greater, reflecting how the normal Lasso tends to choose too small λs. A value as great as $\theta = 0.95 \cdot N$ is proposed.

Roberts discusses how the percentile-Lasso improves the normal Lasso by how it decreases the model-selection error and the variance of the size of the predictive models. It is also easy to apply to normal Lasso. A downside is that the percentile-Lasso is more time-consuming than the normal Lasso in combination with CV. However, that should not be an issue as modern computers have high speed and compared to the importance of a good prediction result, time is nearly irrelevant.

Chapter 3

Methods

There may be several reasons for the instability of the results from Lasso in the paper about CRC[1], mentioned in the introduction. Some initial thoughts were discussed in Section 1.1. In this chapter there is first a description of some challenges when using K-fold CV and R, then follows how simulation have been used to study properties of Lasso and K-fold CV, how properties of the data set affects the prediction result, and finally some graphical displays used later are presented.

3.1 Causes

3.1.1 K-fold cross-validation

The most worrying property of the results in [1], is that they are not unique. The same data set is sent into the R function `optL1` every time, but the coefficients in the result vary. Some variation is expected because the grouping in the CV process is random. However, the level of variation seen in [1] is worrying, motivating a closer look at the CV process, including simulations.

In [1], K-fold CV is repeated, with $K = 10$, 1000 times. A boundary of fifty is chosen in Sveen's paper[1] and the eight signatures that appear more times than that are kept. Because there are validation series available, they are able to test the signatures on these. First, one of the validation series is used to pick out the signature that gives the smallest error in that data set. Then, the other validation series is used to verify that this signature is sensible.

Now, such validation series may be absent and we would like to draw a conclusion based on our original data set. The purpose of the validation series would then be, exactly that, to validate that the signatures behave satisfactory on independent data.

Therefore, it is interesting to investigate how simulated data sets with predecided relevant covariates behave when run through `optL1` with K-fold CV. Will

	LOO	10-fold	4-fold
$n = 100$	38.7 sec	4.8 sec	2.4 sec
$n = 300$	4.0 min	8.8 sec	3.9 sec
$n = 1000$	3 hrs 23 mins 29.2 secs	2 min 25.2 sec	60.5 sec

Figure 3.1: Average time spent *by running LOOCV, 10-fold CV and 4-fold CV once on a simulated data set with information about 1000 genes.*

there be instability in these cases as well? Should the unstable behaviour be expected? These questions are answered in Section 4.1 where knowledge of especially K-fold CV, as described in Section 2.2.7, and simulations as described in this chapter are applied.

One of the reasons for choosing something other than LOOCV is to save time. But how much time can actually be saved? Is the advantage of choosing $K \leq n/2$ so great that it is worth downgrading predictive accuracy? Running a test in R, using `optL1`, with the same input every time (simulated survival data with 1000 genes from n individuals and their survival times), the elapsed time was measured, see Figure 3.1.[1]

It is not until there are more than 300 observations that LOOCV takes so much time that it may be sensible to choose K-fold CV ($K \leq n/2$) and run it several times instead (to see which result singles out). The time difference between 10-fold and 4-fold CV is considerable however, so as long as the properties of 4-fold CV are as good as those of 10-fold CV, 4-fold CV is preferable when considering time. However, the predictive accuracy is more important than the time gain of choosing K-fold CV in favour of LOOCV.

3.1.2 Programming parameters

As discussed in Section 2.3, it is possible to put λ to be in a specific interval when calling `optL1`, forcing it to stay in it. This may affect the prediction result. Also, does `optL1` actually pick out the correct genes?

The resulting models from the `optL1` function should preferentially not vary as much as they do. Using a different R package may be the solution. One of the alternatives is `glmnet`. Still, the CV process comes first and then regularisation, so the framework is the same. How does `glmnet` behave compared to `optL1`? Does one function predict better than the other?

These questions are answered in Section 4.5. The computations are based on simulations described in this chapter.

[1]The test was run on a computer with an Intel(R) Core(TM) i7 CPU (870 @ 2.93GHz, 3066 Mhz, 4 Core(s), 8 Logical Processor(s)) processor and 8 GB RAM.

3.2 Simulations

In order to investigate the possible causes of instability listed in Section 3.1, survival data sets were simulated. Mainly, two issues are interesting to investigate:

1. The variability of the results using K-fold CV on given data sets.
2. The variation of the Lasso results for different data sets generated by the same probability model.

When simulating data, the recipe by Nygård et al.[28] was followed. They were inspired by Datta et al.[29]. Simulated data sets enabled controlling data set properties. Knowledge of which genes that actually are relevant beforehand made it easier to say how successful the Lasso and CV were.

3.2.1 Data structure

Covariate matrix X

$\mathbf{X}$ is the matrix with observed gene expression levels. Every row represents one observation (individual, patient) and every column represents one gene. The numbers may for instance be a measure of how much a gene is expressed in each observation, as introduced in Section 2.1.1. A typical number of observations is n = 100, and the number of genes was set to p = 1000, making $\mathbf{X}$ a 100×1000 matrix, in accordance with Nygård et al.[28].

The elements of $\mathbf{X}$ is drawn from a multivariate normal distribution with zero mean vector. By varying some parameters, the relations between genes and survival can be altered. The covariance matrix is made block by block, 100 genes every time. The elements in the covariance matrices are defined as σ_i^2 on the diagonal, and $\rho\sigma_i^2$ off the diagonal for $i = 1, 2, \ldots, 10$. i is the block index. ρ may vary between 0, 0.3, 0.6 and 0.9, ranging from weaker to stronger gene covariance, respectively. $\rho = 0$ would be no correlation at all (the columns in the covariance matrix are independent of each other), while $\rho = 1$ would be complete correlation (the columns are identical). Alternatives for σ are as follows:

1. All the gene expressions have the same variance. $\sigma_i^2 = 1$ for $i = 1, 2, \ldots, 10$.
2. The first 200 gene expressions have *greater* variance. $\sigma_1^2 = \sigma_2^2 = 2$, $\sigma_i^2 = 1$ for $i = 3, 4, \ldots, 10$.
3. The first 200 gene expressions have *less* variance. $\sigma_1^2 = \sigma_2^2 = 0.5$, $\sigma_i^2 = 1$ for $i = 3, 4, \ldots, 10$.

These three alternatives cover a range of variations. The control of the gene effect strength within each block of 100 genes is described in the next section, whereas the overall effect of the blocks can be controlled by σ_i^2: equal for all blocks, stronger or weaker in the two first blocks.

Gene effects

When simulating survival times, the relation between gene expressions and survival is affected by

$$\eta = q \sum_{j=1}^{100} X_j \beta_j - q \sum_{j=101}^{200} X_j \beta_j = q \sum_{j=1}^{100} \beta_j (X_j - X_{j+100}), \tag{3.1}$$

where X_j is the jth column (gene) of $\mathbf{X}$ and $\beta_j = \beta_{j+100} = \exp(-a(j-1))$ for $j = 1, \ldots, 100$. Only the 200 first genes are thus related to survival. These genes are made up of two blocks and they will cover two peaks of strong gene effects. The constant a is given by $\exp(-a(k-1)) = 0.5$, and so

$$a = \frac{\ln 0.5}{1-k} = \frac{1}{(1-k)\ln 2} \tag{3.2}$$

k is set to 10, 50 or 100, describing that the k first genes in block 1 and 2 have an effect of more than 0.5.[2] The slope of $\boldsymbol{\beta}$ is illustrated in Figure 3.2.

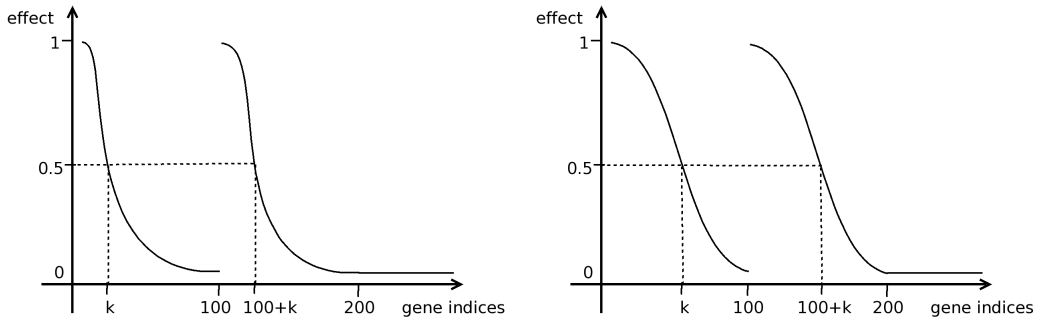

Figure 3.2: Gene effect slope. *The shape of the $\boldsymbol{\beta}$ slope. The **left** plot shows the situation where k is small, whereas to the **right**, the k has been increased. β_k is always 0.5, and β_1 is 1.*

E.g. for $k = 10$, the first 100 elements of $\boldsymbol{\beta}$ will look like this $\beta_{1:100} = [1, 0.92, 0.85, \ldots, 0.5, \ldots, 0.00]^T$ where 0.5 is the 10th element and the last elements are so small that correcting to two decimal places, they are zero. Now take a closer look at the right hand side of (3.1). Because β_j is larger for j small, the differences between the first genes in the two blocks affect the prediction outcome more than the next genes. This way, η describes the effect of the gene expression levels in the covariate matrix $\mathbf{X}$, and the first genes in the two first blocks are

[2]Note that this k is different from the K in K-fold CV. It may be a bit confusing that these two are similarly named, but care will be taken to always place the term in the correct context to ease the understanding. This way, the notation can stay true to the common or original form.

expected to appear more frequently when searching for the relevant genes, if the method works properly.

When a is found and used to find $\boldsymbol{\beta}$, we may find q, which is needed to find η. The q we decide to use, is the one that makes the variance of η to be 1. Thus

$$\text{Var}(\eta) = \frac{1}{n}\sum_{i=1}^{n}(\eta_i - \mu)^2 = 1, \quad \mu = \frac{1}{n}\sum_{i=1}^{n}\eta_i, \tag{3.3}$$

for some large n, say 100. This is used to find the q that solves (3.3).

$$q = \sqrt{\frac{n}{\sum_{i=1}^{n}(\phi_i - \frac{1}{n}\phi)^2}} \tag{3.4}$$

The values of ϕ_i are easily implemented in R, and so it is no problem to find q. With $\mathbf{X}$ from the previous section and $\boldsymbol{\beta}$ and q above, the value of η may be found, leading us onwards.

Survival times

Following Nygård et al.[28], survival times from a Weibull distribution with hazard rate $h(x) = 5x^4\exp(\eta)$ are drawn. η comes from the estimation of gene effects.

To draw Weibull distributed numbers, a scale λ and a shape k to fit to the standard $h(x;k,\lambda) = \frac{k}{\lambda}(\frac{x}{\lambda})^{k-1}$ are required. In this case $k = 5$ and λ is found by

$$\begin{aligned}\frac{5}{\lambda}\left(\frac{x}{\lambda}\right)^4 &= 5x^4\exp(\eta)\\ \lambda &= \exp(-\eta/5)\end{aligned}$$

Censoring times

If observation i ended in death by the illness in interest, the indicator is set to 1: $\delta_i = 1$, otherwise $\delta_i = 0$. To distribute censoring indicators, censoring times were simulated.

The simulation of censoring times differs from [28]. Uniformly distributed numbers between 0 and 2 are generated, as this is approximately the interval of the survival times. The numbers are compared to the survival times. Say t_{cens} are the uniformly distributed numbers and t_{surv} are the survival times drawn previously. Then for every pair of $(t_{surv}, t_{cens})_i$ for $i = 1, \ldots, 100$:

- If $t_{surv} < t_{cens}$: Keep t_{surv} as the survival time and put $\delta = 0$ (noncensored),
- else, that is $t_{surv} \geq t_{cens}$: Replace the survival time with t_{cens} and put $\delta = 1$ (censored).

With this choice, approximately 50% of the observations are censored.

3.2.2 Simulation parameters

There are several parameters in the algorithm described above that can alter the behaviour of the data set. In summary:

- k = **10, 50, 100. The first genes in block 1 and 2 have the greatest effect.** Increasing k, i.e. slowing down the speed of gene effect decrease, is expected to lead to greater signature sizes because the gene effects will be more spread out. With a small k, the non-zero coefficients should be grouped in the start of the two first blocks.

- ρ = **0, 0.3, 0.6, 0.9. Correlation between the genes.** There are several interesting aspects to look upon, e.g. if strong correlation between genes result in small gene signatures, as expected. (For strongly correlated groups that show relevance, Lasso includes only one variable and leaves the rest[17].) That is, with ρ decreasing, greater signature sizes are expected.

- **Three alternatives for variance:** The genes in all ten gene blocks have the same variance; the genes of the first two blocks differ from the rest (greater or lower variance). For now, all genes are set to have the same variance, 1.

- **Number of observations** n**.** A greater number of observations is expected to lead to more accurate results: more of the (predecided) most relevant genes will be picked out as relevant and there will be less false positives. More observations tend to have the effect of better prediction, but it is not guaranteed. If a new observation is an outlier, it may have a negative effect on the prediction result.

The results of investigating the issues raised above are described in Section 4.3.

3.3 Displaying the results

Biostatistics with a large number of covariates have a tendency to bring on huge amounts of information. In order to avoid getting lost in the jungle of information, well-designed visualisation is important. A description of the diagrams that will be used when the results are presented follows.

3.3.1 Signature size bar diagram

Visualising the resulting signature sizes from many runs on one specific data set can easily be done with a bar diagram, as in Figure 1.2. Another example is based on the data below and is shown to the right in Figure 3.3.

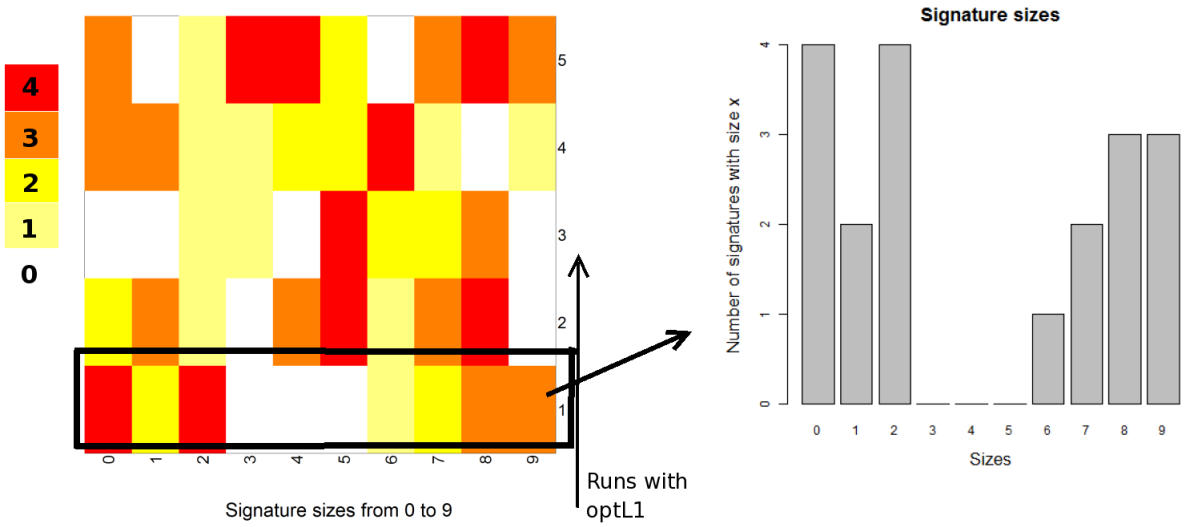

Figure 3.3: Example of signature size heat map and bar diagram. *The bar diagram to the* ***right*** *shows one data set run 19 times, with 10-fold CV. The sizes of the resulting 19 signatures are represented in the bars. E.g. there were four signatures of size zero, no genes with non-zero coefficients. One row in the heat map to the* ***left*** *shows the same information as the bar diagram; the darker the colour, the higher bar. The heat map in total shows five data sets that has been run 19 times each.*

Size	0	1	2	3	4	5	6	7	8	9
Frequency	4	2	4	0	0	0	1	2	3	3

The signature sizes are represented along the x-axis and the height of the column represents how many times the result signature is of that size.

3.3.2 Signature size heat map

Many runs on several different data sets and comparing the results with each other can become messy and difficult to keep track of. Instead of several bar diagrams, a heat map is used, as in Figure 3.3.

Every row in the heat map represents a number of runs with one specific data set, like our previous bar diagram. The colours represent the bar height in the bar diagram in Figure 3.3. Darker nuance symbolises a higher bar. The matrix behind the heat map is as follows:

```
     0 1 2 3 4 5 6 7 8 9
[1,] 4 2 4 0 0 0 1 2 3 3
[2,] 2 3 1 0 3 4 1 3 4 0
[3,] 0 0 1 1 0 4 2 2 3 0
[4,] 3 3 1 1 2 2 4 1 0 1
```

```
[5,] 3 0 1 4 4 2 0 3 4 3
```

3.3.3 Gene index heat map

Sometimes it is interesting to look at which genes are picked out in the resulting gene signature. A heat map may be used in this case as well, but with every column representing a gene instead of signature size. A row will be a gene signature consisting of the indicated genes in that row, see heat map in Figure 3.4. A coloured element in the heat map means that the gene has a non-zero coefficient in the Lasso model.

3.3.4 Gene index density plot

Also, we may wish to look at the coefficient sizes. A density graph can be added to visualise the sum of the coefficients for that gene (column). See density plot in Figure 3.4.

Both the density plot and heat map in Figure 3.4 are based on this matrix:

```
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,] 0.00 0.00 0.00 0.00 0.00 0.00 0.00    0 0.00  0.00
 [2,] 0.12 0.00 0.12 0.12 0.12 0.12 0.12    0 0.12  0.12
 [3,] 0.12 0.00 0.12 0.12 0.12 0.12 0.12    0 0.12  0.12
 [4,] 0.00 0.00 0.00 0.00 0.50 0.50 0.00    0 0.00  0.00
 [5,] 0.00 0.00 0.00 0.00 0.00 0.00 0.00    0 0.00  0.00
 [6,] 0.00 0.00 0.00 0.00 0.00 0.00 0.00    0 0.00  0.00
 [7,] 0.14 0.00 0.14 0.14 0.14 0.14 0.00    0 0.14  0.14
 [8,] 0.00 0.00 0.00 0.00 0.50 0.50 0.00    0 0.00  0.00
 [9,] 0.00 0.00 0.00 0.00 1.00 0.00 0.00    0 0.00  0.00
[10,] 0.00 0.00 0.00 0.00 0.00 0.00 0.00    0 0.00  0.00
[11,] 0.00 0.00 0.00 0.00 0.50 0.50 0.00    0 0.00  0.00
[12,] 0.00 0.00 0.00 0.00 0.50 0.50 0.00    0 0.00  0.00
[13,] 0.17 0.00 0.17 0.17 0.17 0.17 0.00    0 0.00  0.17
[14,] 0.00 0.00 0.00 0.00 1.00 0.00 0.00    0 0.00  0.00
[15,] 0.12 0.00 0.12 0.12 0.12 0.12 0.12    0 0.12  0.12
[16,] 0.14 0.00 0.14 0.14 0.14 0.14 0.00    0 0.14  0.14
[17,] 0.11 0.11 0.11 0.11 0.11 0.11 0.11    0 0.11  0.11
[18,] 0.11 0.11 0.11 0.11 0.11 0.11 0.11    0 0.11  0.11
[19,] 0.11 0.11 0.11 0.11 0.11 0.11 0.11    0 0.11  0.11
```

Note that this matrix also represent the bar diagram and row 1 in the heat map in Figure 3.3. (The number of non-zero elements in every row is equal the signature sizes.)

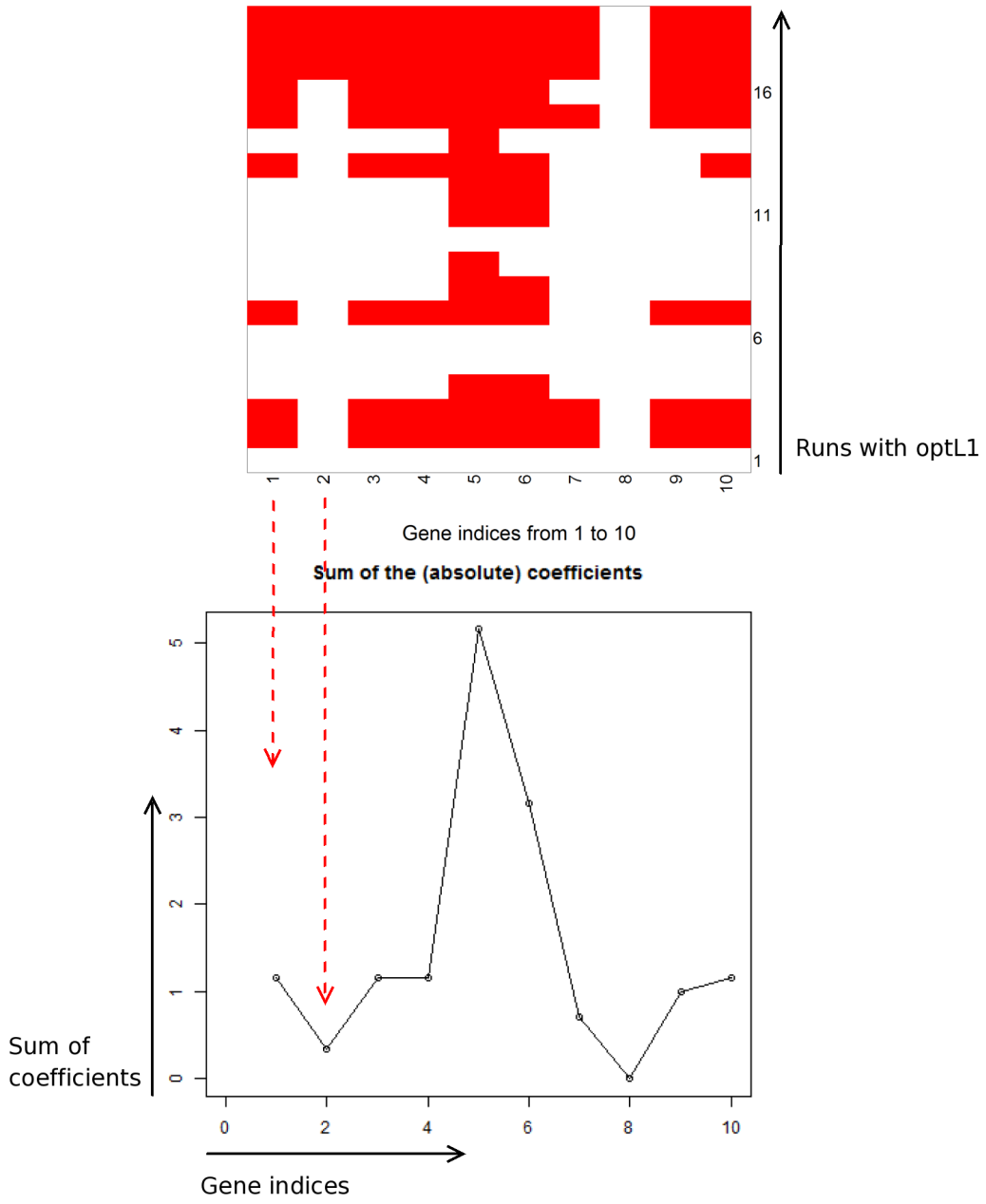

Figure 3.4: Example of gene index heat map and coefficient density plot. Upper: *Every row in the heat map represents a gene signature. E.g. the the signature on the row marked 19 contains nine genes with non-zero coefficients, which is all genes except gene 8. Gene 8 is not present in any of the signatures. The size of the coefficients are ignored in the heat map.* ***Lower:*** *The density plot shows the sum of the coefficients for every gene. The x-axes in the heat map and in the density plot correspond. Note that gene 8 sums to zero, but gene 5, which is present in nearly all signatures, has the highest sum.*

The fact that all non-zero elements in each row are equal and sum up to one is not usually the case. The data set here was generated for illustrative purposes and should not be considered typical in real scenarios.

Chapter 4

Simulation Results

Data was simulated by following the recipe described in Section 3.2. The data was then run through the R function `optL1`, which performs Lasso and CV and was used in the CRC paper[1]. (For motivation and description see Section 1.1.)

Investigation of how Lasso behaves is interesting, as indicated by the CRC example in Section 1.1. Generally it is interesting to see how Lasso behaves on one data set, both with fixed fold parameter in K-fold CV set and also how the result varies between different folds. Another interesting aspect is whether some data sets give less varying results than others, e.g. is the instability of the data set of the CRC paper[1] an atypical result? And, does more observations lead to more consistent results?

Unless specifically stated, the parameters are set to (i) $k = 10$, the first gene with effect 1, the 10th with effect 0.5, then further decreasing; the k first genes in each block have the largest effect, (ii) $\rho = 0$, no correlation, (iii) $n = 100$, number of observations, and (iv) $p = 1000$, total number of genes.

4.1 The importance of the fold parameter

4.1.1 Variation within K-fold CV

Figure 4.1 shows an example with ten repetitions of Lasso and CV using `optL1`. Even though it is the same data set every time, the result is varying. Because the result is so variable, it may be nonsensible to draw a conclusion based on just one or even ten runs. However, note that the smallest signature is of size two, and that those two genes are present in all signatures. CV decides the value of λ, so it only decides how many genes that are included. Thus all smaller signatures will be contained in the larger ones, i.e. the gene of a 1-size signature is also contained in all larger signatures, both genes in a 2-size signature are contained in the larger signatures and so on.

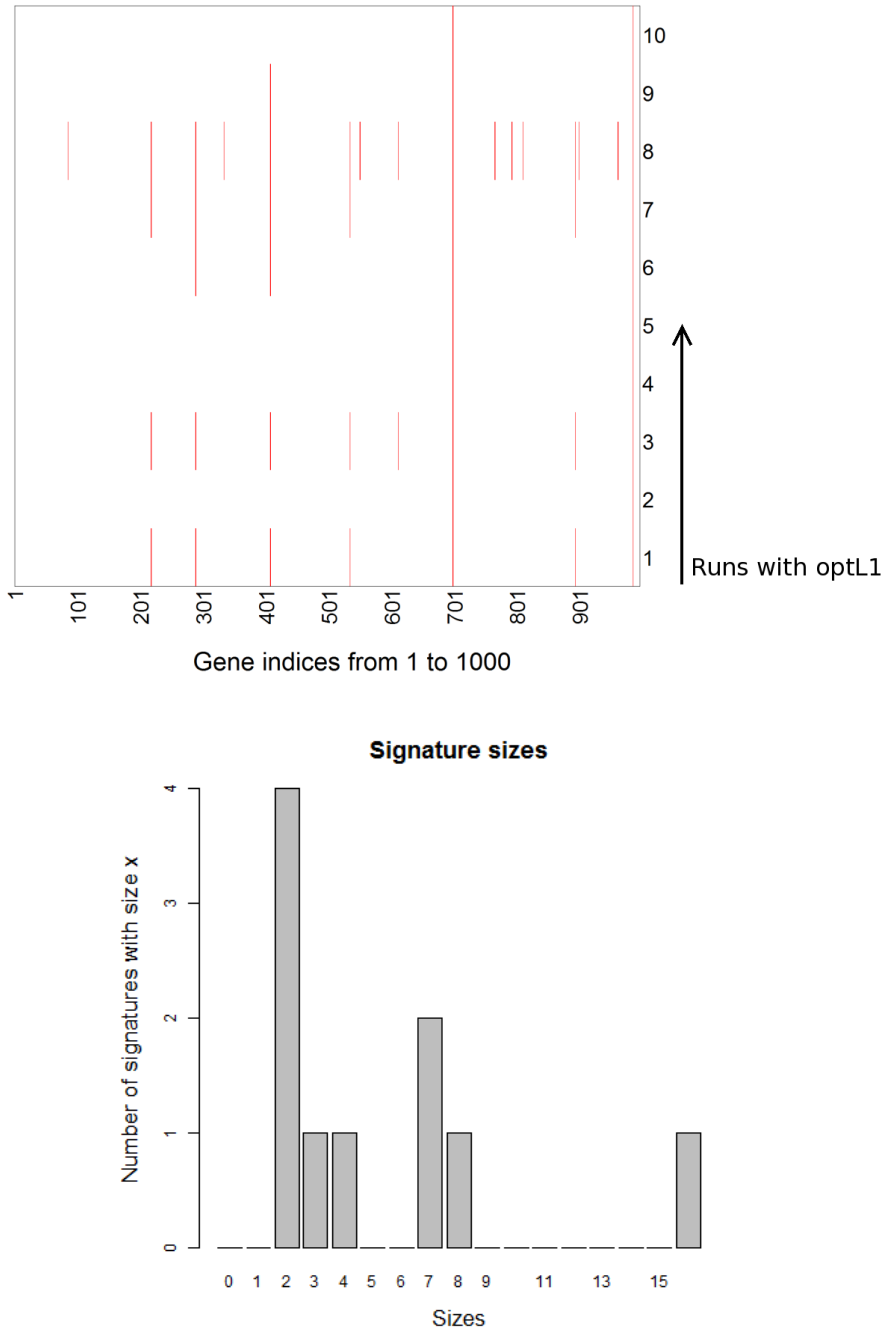

Figure 4.1: One dataset, 10-fold CV. *The heat map (top) shows the result of 10-fold CV on the same data set. Note that smaller signatures are contained in the larger ones. The bar plot (bottom) shows the signatures summed up as sizes. The number of genes in each row in the heat map is the 'size' in the bar plot. Note that the sum of the bar heights is the same as the number of rows in the heat map (10) and shows the same information as Figure 1.2 in the introduction.*

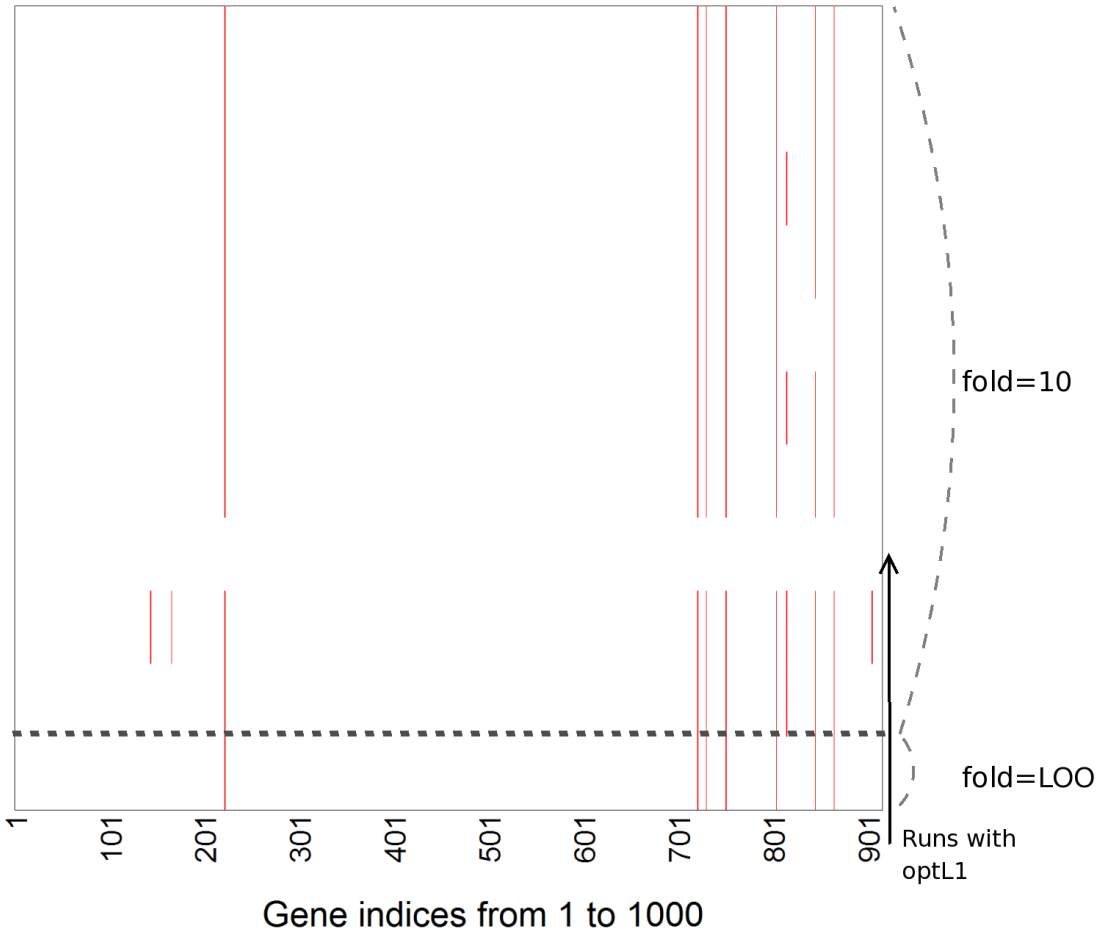

Figure 4.2: One dataset, LOO and 10-fold CV. *The heat map shows the result of LOOCV and 10-fold CV on the same data set. LOOCV was run only once because it results in a unique signature.*

4.1.2 Leave-one-out or K-fold cross-validation

The pros and cons of K-fold CV were indicated earlier. It would appear that LOOCV is the better alternative for uniqueness. It finds the best λ for the given data set, and there is no randomness to it. K-fold CV on the other hand may give a different result every time because the grouping into training and test sets is different every time. A comparison of LOOCV and 10-fold CV can be seen in Figure 4.2.

Even though LOOCV may be the best alternative by providing a unique answer, it may not be the best alternative for the general case. First, K-fold CV may actually be better if the model should be used for predictive measures because it gives a hierarchy of signatures as utilised in [1]. Moreover, a paper by Markatou et al.[30] states that K=4 is the best option, making the model stronger

for the general case. The main argument against choosing LOOCV is that every training set is nearly equal, because only one observation is excluded every time, and therefore strongly depend on each other. This does not train models well for a general scenario, but instead makes it very suitable for the specific data set. With $K \leq n/2$ the differences between the training sets are greater, so the model will be better suited for application on other data.

4.1.3 Variation between K-folds

How do different K-fold CV behave on the same data set? LOO, 25-fold, 20-fold, 10-fold, 5-fold, 4-fold and 3-fold CV were applied to the same data set, see Figure 4.3. It seems that 3-fold CV gives the largest signatures, i.e., highest λ. Is this always the case? For visualisation of more data sets, see Figures A.12 and A.13. A closer look at each pair of K-folds can be found in Figures A.1, A.2 and A.3.

A smaller K in K-fold seems to result in larger signatures, but an exception can be seen in Figure A.13. Here, 10-fold CV seems to pick out fewer genes than both 25-fold and 20-fold. Otherwise, the claim seems sensible.

Investigation of this behaviour on may data sets, e.g. 100, is preferable. In the calculations behind Figure 4.3, the average signature size within each fold was registered. The process was repeated for 100 data sets, even though just one data set is presented here, in Figure 4.3, and two can be found in the appendix. The average of all averages were calculated and the result was:

	LOO	25-fold	20-fold	10-fold	5-fold	4-fold	3-fold
Average sig.size	10.0	10.8	11.1	12.2	14.8	15.9	18.1

The average signature size of all data sets are placed in Section A.1.

The claim that more folds, i.e. larger K, give more sparse signatures holds, and, as previously mentioned, 4-fold CV has been suggested as the ideal fold[30] when predicting a general case. Will smaller Ks give more stable results? This question is not answered in this thesis, but would be interesting to look upon.

4.1.4 Summary

In this section, we have observed that:

- There are variations in the result from repetition to repetition when keeping the same data set and repeatedly running K-fold CV and Lasso. LOOCV gives a unique result, whereas K-fold for $K \leq n/2$ gives varying results.
- More folds, i.e. larger K, give more sparse prognostic signatures.
- All smaller signatures are contained in the larger signatures.

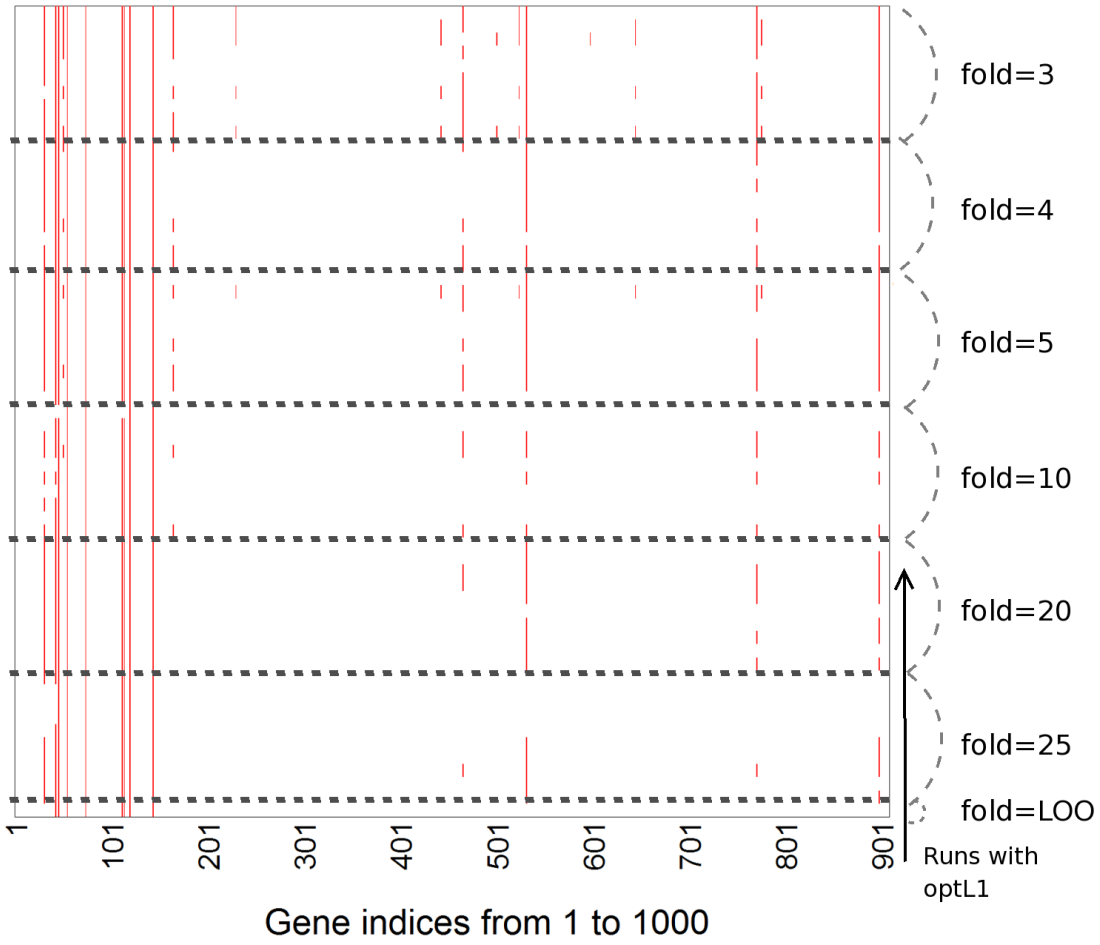

Figure 4.3: One dataset, different folds. *The heat map shows the result of Lasso and CV on the same data set with* $n = 100$ *observations. The columns are the genes ranging from index 1 to the left to 1000 on the right. Each K-fold CV (and Lasso) was performed 10 times, which resulted in 10 rows for each K-fold. LOOCV was run only once as it gives a unique gene signature. Note how the gene signature (genes with non-zero coefficient) goes less sparse as* K *in K-fold decreases from LOO to 3.*

- These results are seen on simulated data with controlled behaviour, and are thus not a result of some specific attributes of Sveen et al.'s data[1].

4.2 The effect of new data sets

The motivation for using regularisation methods are that we have many more genes than observations, $p >> n$. Investigation of a rare condition, lack of volunteers, or lack of appropriate biological material from tumour samples can cause the observation set to be small. A typical size is about 100, as in the CRC data set[1] where $n = 95$. Therefore, $n = 100$ is the default number of observations in the simulated data sets.

4.2.1 *n=100*

Figure 4.4 shows the signatures when $n = 100$, where Lasso was run repeatedly on new data sets. There is generally much noise, i.e. false positives, but peaks can still be seen in the areas with real gene effects, roughly 1-20 and 101-120; Lasso chooses the correct genes more often than irrelevant ones, but not as often as would be preferable.

4.2.2 *n=300*

Tripling of n seems to improve the detection of the real gene effects, see Figure 4.5. The increment of the number of observations has led to a stronger difference between the first genes and the last genes in each of the two first blocks. With more observations, the genes with estimated effects are grouped around the expected areas, roughly 1-20 and 101-120.

In a case where the number of observations can be increased, more than 100 should preferentially be chosen. The pay off of having a greater observation set is marked.

A function of n that describes the relation between the number of observations and false positives would be useful. Whether a more formal approach to such a function would be successful is unknown, but is discussed in Section 6.4.

Figure 4.6 displays the same information as in Figure 4.1, repeated runs with 10-fold CV on the same data set, but the data set consist of 300 observations instead of 100. The two figures mentioned are based on just one data set each, so take care when drawing conclusions. However, increasing the number of observations seems to make the `optL1` results within the same data set markedly more stable.

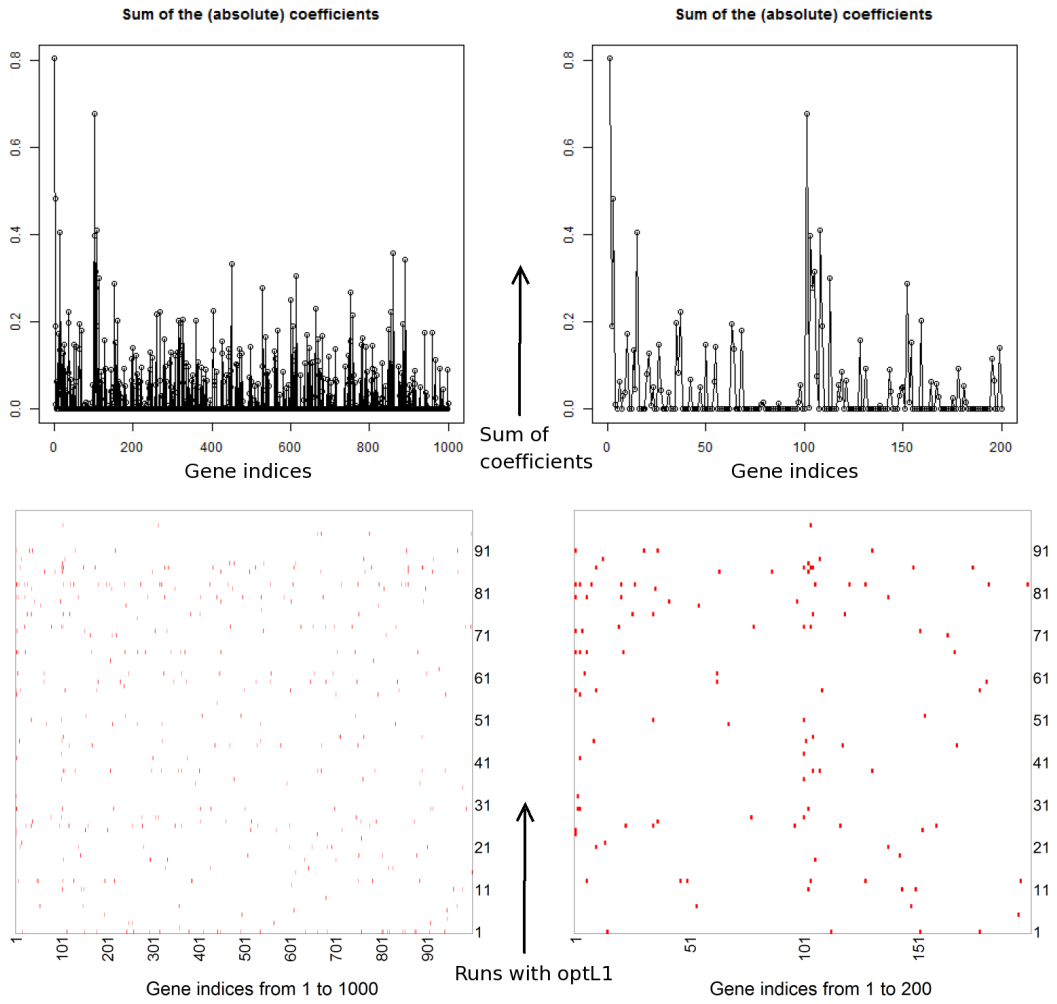

Figure 4.4: 100 observations, LOOCV. *The data sets have simulation parameters* $\rho = 0$ *and* $k = 10$*. The* ***left*** *plots show all 1000 genes whereas the* ***right*** *plots are zoomed in on the first 200. There are many non-zero coefficients outside the expected areas 1-10 and 101-110, but especially at 101-110, there are more occurrences. Larger versions of these images can be found in Figures A.4 and A.5.*

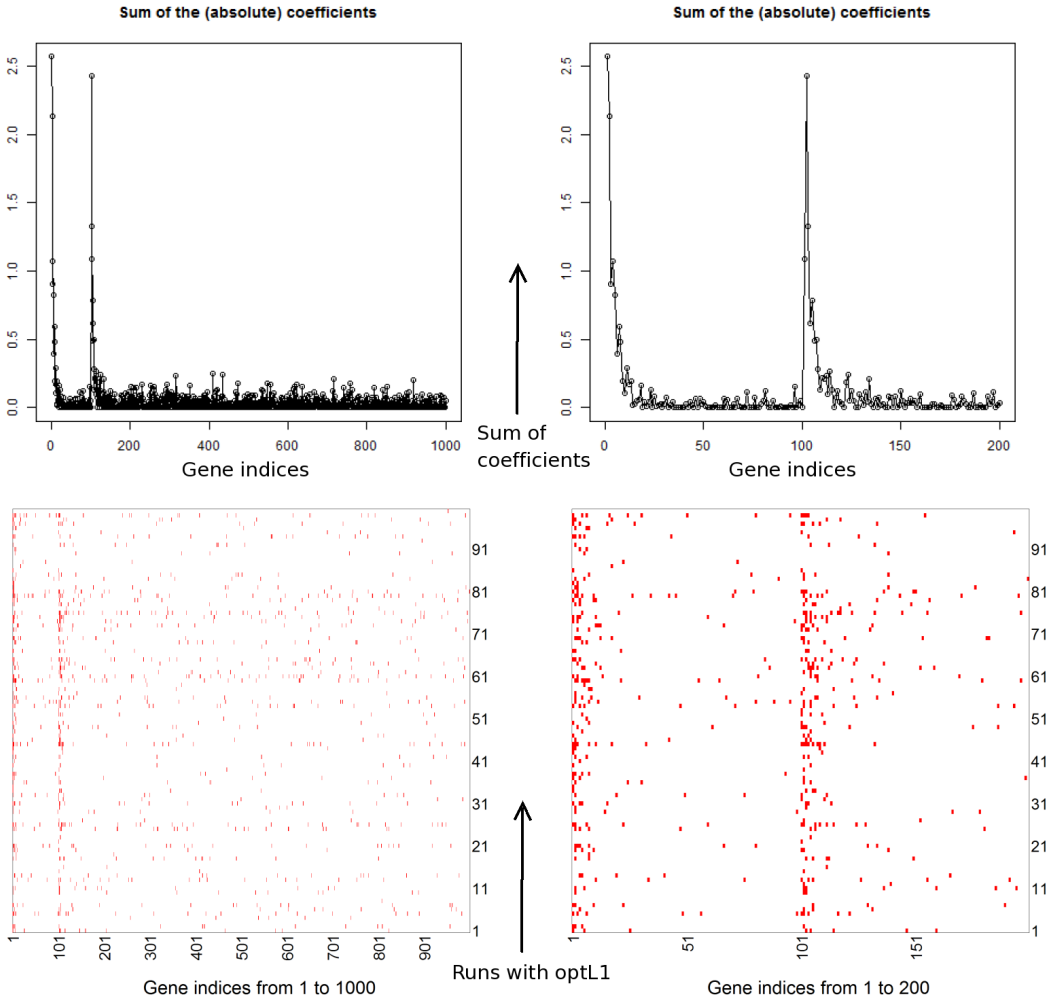

Figure 4.5: 300 observations, LOOCV. Left: *All 1000 genes.* ***Right:*** *The first 200 genes. The effect of making the first genes more relevant to response is markedly more obvious in this result compared to when* $n = 100$*, see Figure 4.4. Larger versions of these images can be found in Figures A.6 and A.7.*

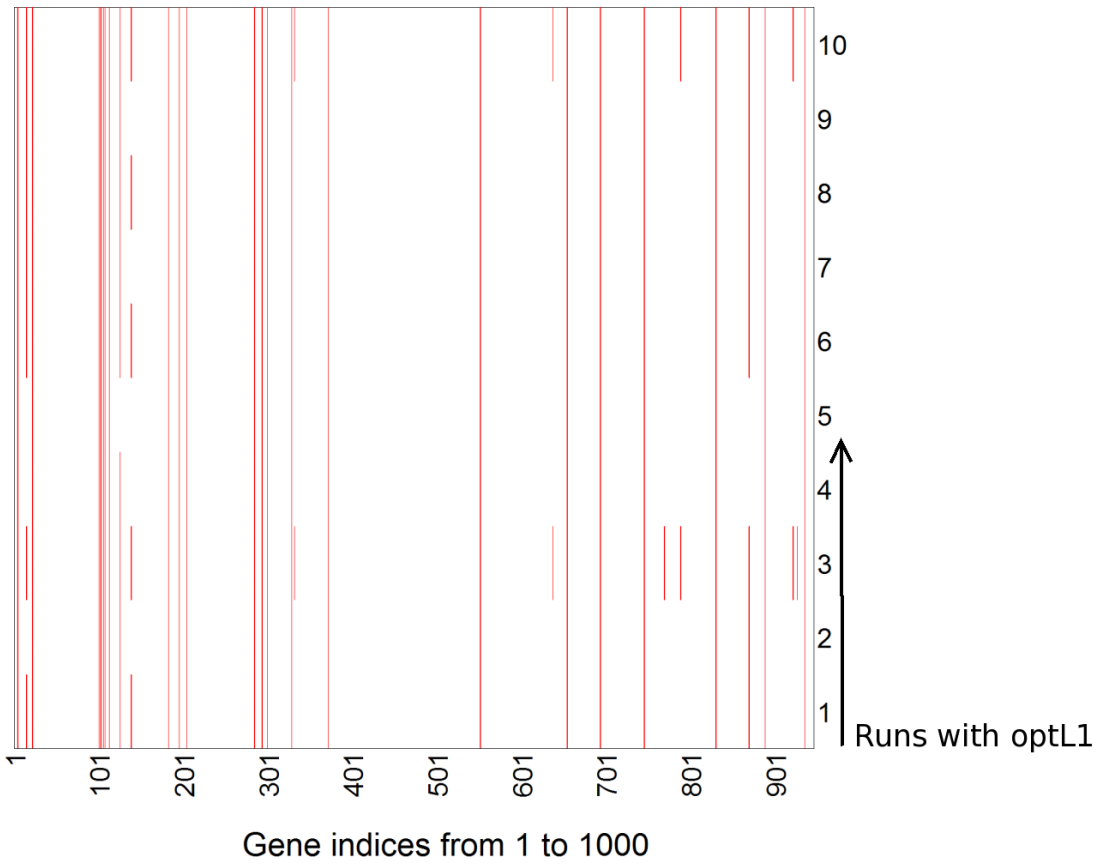

Figure 4.6: One dataset, 10-fold CV, n=300. *The heat map shows the result of 10-fold CV on the same data set.*

4.2.3 *n=500*

To further illustrate that increasing the observation set will lead to more precise results, see Figure 4.7 where $n = 500$. The peaks are sharper, so the real relevant genes have larger coefficients and appear in the predictive models more often. Again, having more observations available is preferable for precision.

Of course, we cannot know for sure whether more observations will always lead to more precise results as the search for an association between response and covariates can lead to the finding of a pattern that is not really there.

4.2.4 Summary

More observations will lead to less varying behaviour of K-fold CV and Lasso in general. The improvement by moving from 100 to 300 is large, and we may ask if data sets of 100 observations are sufficient for drawing conclusions when p is large.

Also, more observations seem to make Lasso choose larger signatures, compare Figures 4.1 and 4.6, which is expected as the true relevant genes are more easily found with the higher test power obtained when n is large.

4.3 The effect of simulation parameters

4.3.1 Variations in correlation

By varying ρ, we vary the correlation between the genes, see Section 3.2. $\rho = 1$ means complete correlation, and $\rho = 0$ means no correlation at all. When a group of variables has strong correlation, the Lasso will tend to pick one of the variables and leave out the rest. So decreasing the correlation ρ was expected to lead to greater signatures, i.e. more genes with non-zero coefficients.

Figure 4.8 shows how the coloured boxes have a tendency to move to the right as ρ decreases, i.e. the signatures are larger when the correlation decreases.

However, signature sizes drop when there is no correlation, $\rho = 0$. A display of the signature sizes when the correlation moves from 0.3 to 0 shows a gradual decrease, see Figure 4.9. A comparison of a run with the number of observations equal 100 and another with $n = 300$, both with zero correlation, showed that the signatures were greater when $n = 300$. The drop in signature size at $\rho = 0$ is most likely due to the genes having too little power by themselves. With correlation "they draw" effect from the correlated genes, which are then left out.

In Figure 4.10, `optL1` was run once on every dataset, and the heat map visualises how the non-zero coefficients are distributed related to gene index. The bottom part consists of data sets generated with strong correlation, and the top part with weak correlation. The correlation seems to affect which genes are

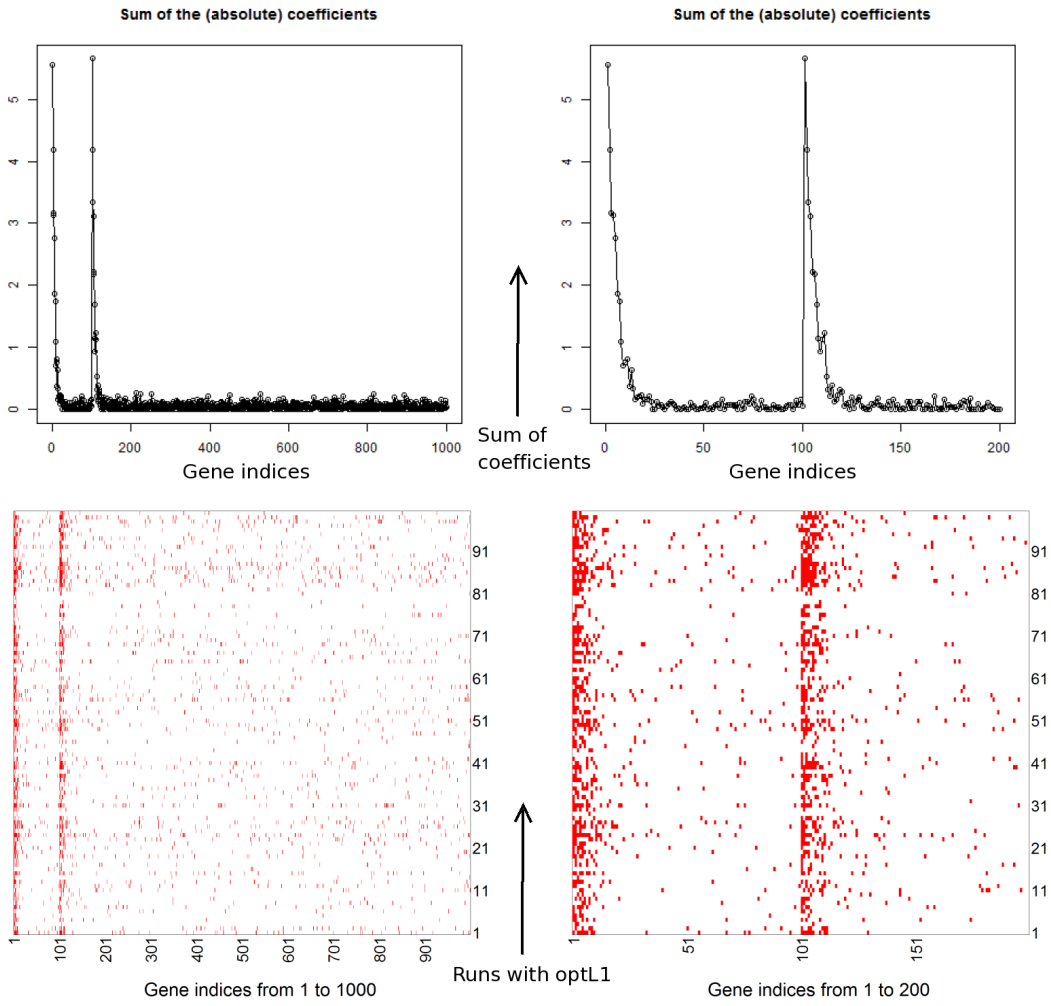

Figure 4.7: 500 observations, LOOCV. Left: *All 1000 genes.* ***Right:*** *The first 200 genes. The effect of making the first genes more relevant to response is more obvious in this result compared to both* $n = 100$ *and* $n = 300$*, see Figures 4.4 and 4.5. Larger versions of these images can be found in Figures A.8 and A.9.*

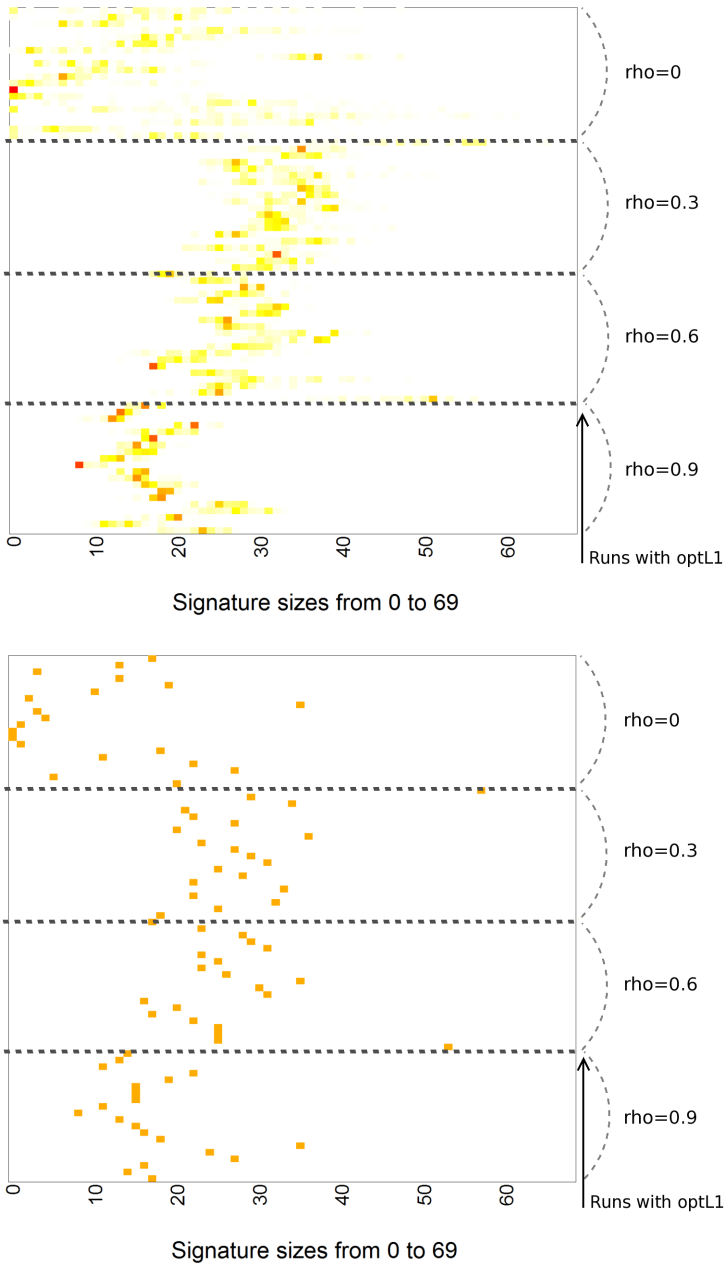

Figure 4.8: Variations in correlation. *These heat maps show the signature sizes. Each row symbolises one dataset with N runs of* `optL1` *(Lasso and K-fold CV);* ***Top****: 10-fold CV;* ***Bottom****: LOOCV. For 10-fold CV, the function was run N=100 times, i.e. the sum of each row is 100. The darker colours symbolises greater values. LOOCV was run only once per dataset as it gives a unique answer. Note: There are 300 observations in these data sets to more clearly see the effect.*

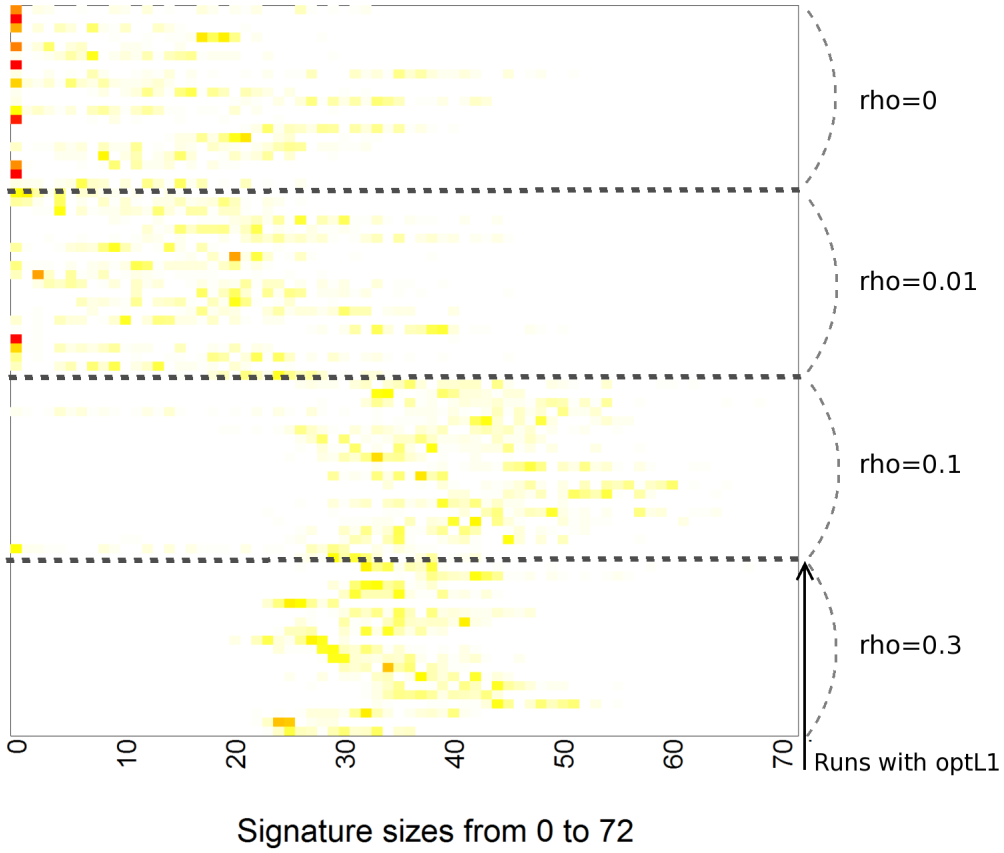

Figure 4.9: Weak correlation, $n = 300$. *The signature sizes increased from* $\rho = 0.9$ *to* $\rho = 0.3$, *but suddenly dropped at* $\rho = 0$. *The behaviour from* $\rho = 0.3$ *to* $\rho = 0$ *is displayed in more detail here. There is a gradual decrease in signature sizes.*

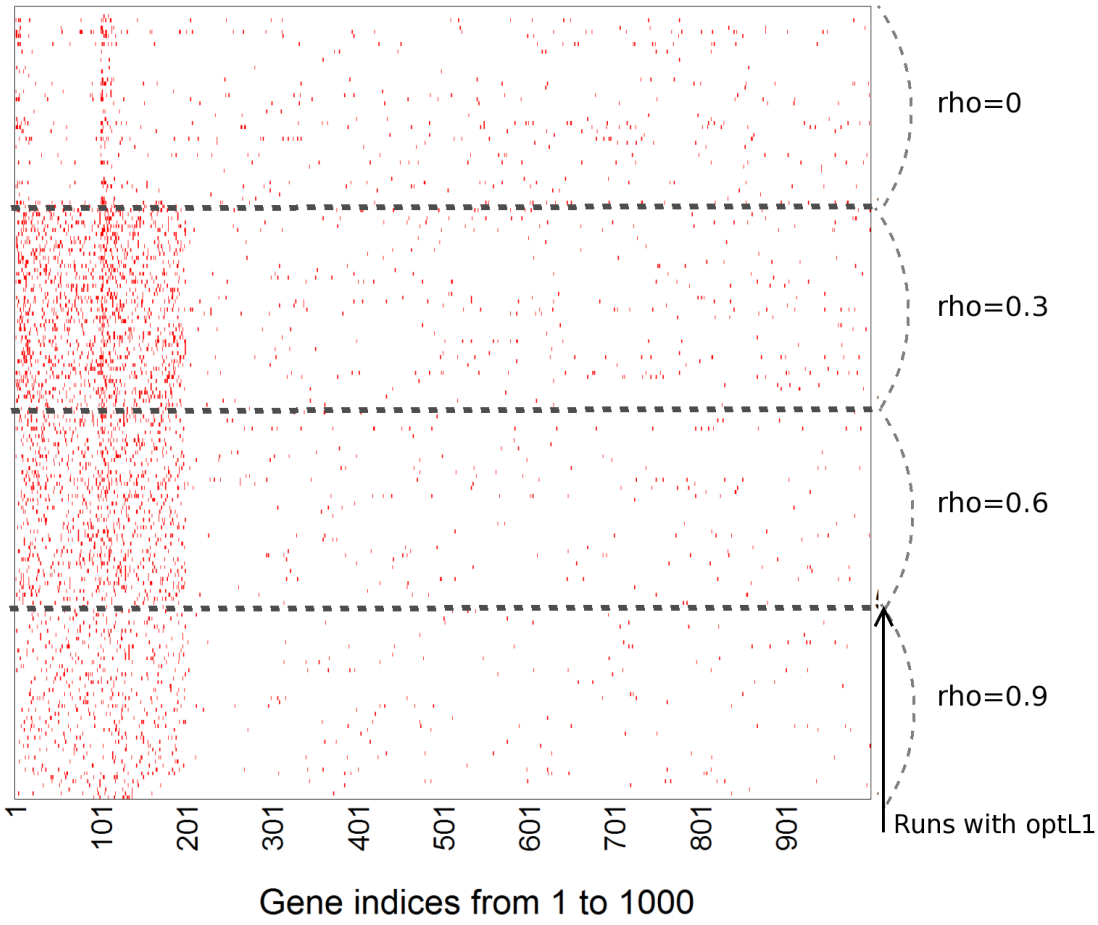

Figure 4.10: Variations in correlation, gene indices. *n=300. This heat map show the genes with non-zero coefficients after running* `optL1` *with LOOCV. Each row is a new dataset and a new run with* `optL1`*.*

picked out in the predictive model. Stronger correlation results in less non-zero coefficients in the irrelevant area (indices > 200).

The increased signature sizes seen in Figure 4.8 are also seen here, as there are more red elements as ρ decreases, with an exception where $\rho = 0$. This is explained in the previous section.

4.3.2 Variations in how fast the gene effect decreases

In Figure 4.11 the correlation is kept constant at 0.3[1] and $n = 300$, but the speed of the gene effect reduction, k, is varied. Obviously, with k increasing, larger signatures are expected because the gene effects would be more spread

[1]The correlation was not set to the neutral zero because of the drop in signature size effect which was shown in Section 4.3.1.

out. However, the normalisation performed when converting $\boldsymbol{\beta}$ to $\boldsymbol{\eta}$, described in Section 3.2.1, makes this claim complicated. The heat maps do not confirm the previous claim; the sizes seem to be similar for all ks. If anything, the results of 10-fold CV seem more stable with a larger k.

Figure 4.12 shows the indices of the genes with non-zero coefficients in each signature. One row represents one dataset run through `optL1` with LOOCV once, resulting in one signature. It was expected that $k = 10$ would lead the genes clustered roughly in the intervals 1-10 and 101-110, or close to them, $k = 50$ in the intervals 1-50 and 101-150, and so on. For $k = 10$, this seems to be the case. For the other values of k it is difficult to say whether they differ from each other.

4.3.3 Summary

It is already known that Lasso normally chooses only one variable with non-zero coefficients out of a (strongly) correlated group, so the effect of correlated genes was expected; strong correlation give smaller signatures.

The drop in signature size when there was no correlation was unexpected. The effect of each gene is weak by itself. It needs a correlated gene to draw strength from. This probably causes the drop in signature sizes when $\rho = 0$. More observations strengthen the gene effects, so the sudden appearance of zero-signatures when $\rho = 0$ is not as dramatic when $n = 300$ as when $n = 100$.

The normalised gene effects complicates the expectancy of how the gene effect distribution affects the result. The signature sizes seemed alike for all values of k.

4.4 Reduction of small coefficients to zero

One of the benefits of Lasso is that it reduces the number of covariates with non-zero coefficients. Other regularisation methods such as ridge only shrinks the coefficients towards zero. However, even though Lasso picks a covariate with non-zero coefficient, the coefficient may be very small. It may be so small that in effect it does not contribute to the prediction. When the focus is on mapping the relevant genes, i.e. not finding the exact coefficients but just finding the genes with non-zero coefficients, these genes with small coefficients may be of no interest. In fact, it would be useful to rule them out and instead find the most relevant genes.

To strengthen the effects of the truly relevant genes, `optL1`'s choice of λ was decreased by a constant `eps=10`. This way, λ is forced to move to the right of Figure 2.5 which leads to more covariates; those that already were in the model will mostly have greater coefficients, and the new covariates will have small coefficients.

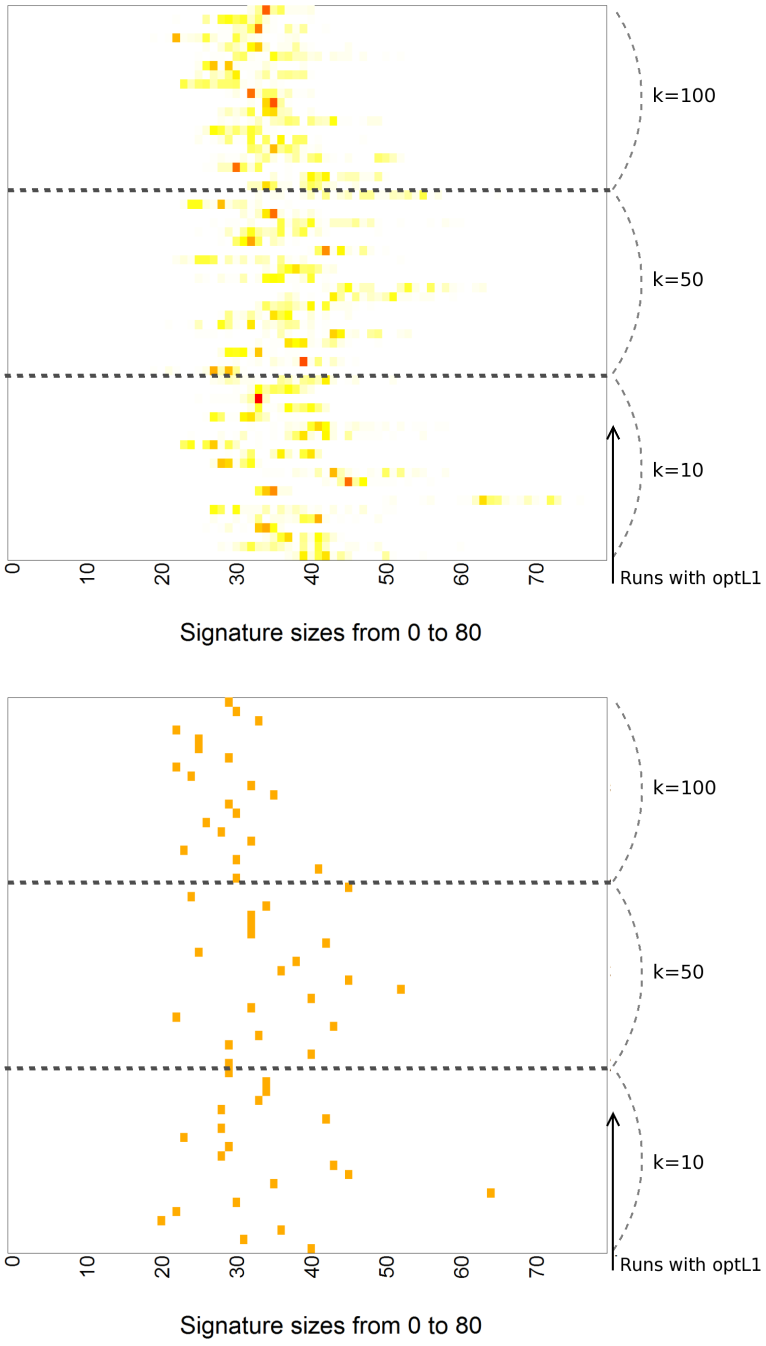

Figure 4.11: Speed of the gene effect reduction, sizes. *The simulation parameters were set to* $n = 300$ *and* $\rho = 0.3$*. The heat maps show that the signature sizes stay approximately the same even though the speed of the gene effect reduction decreases. The top image shows 10-fold CV, and the bottom image shows LOOCV.*

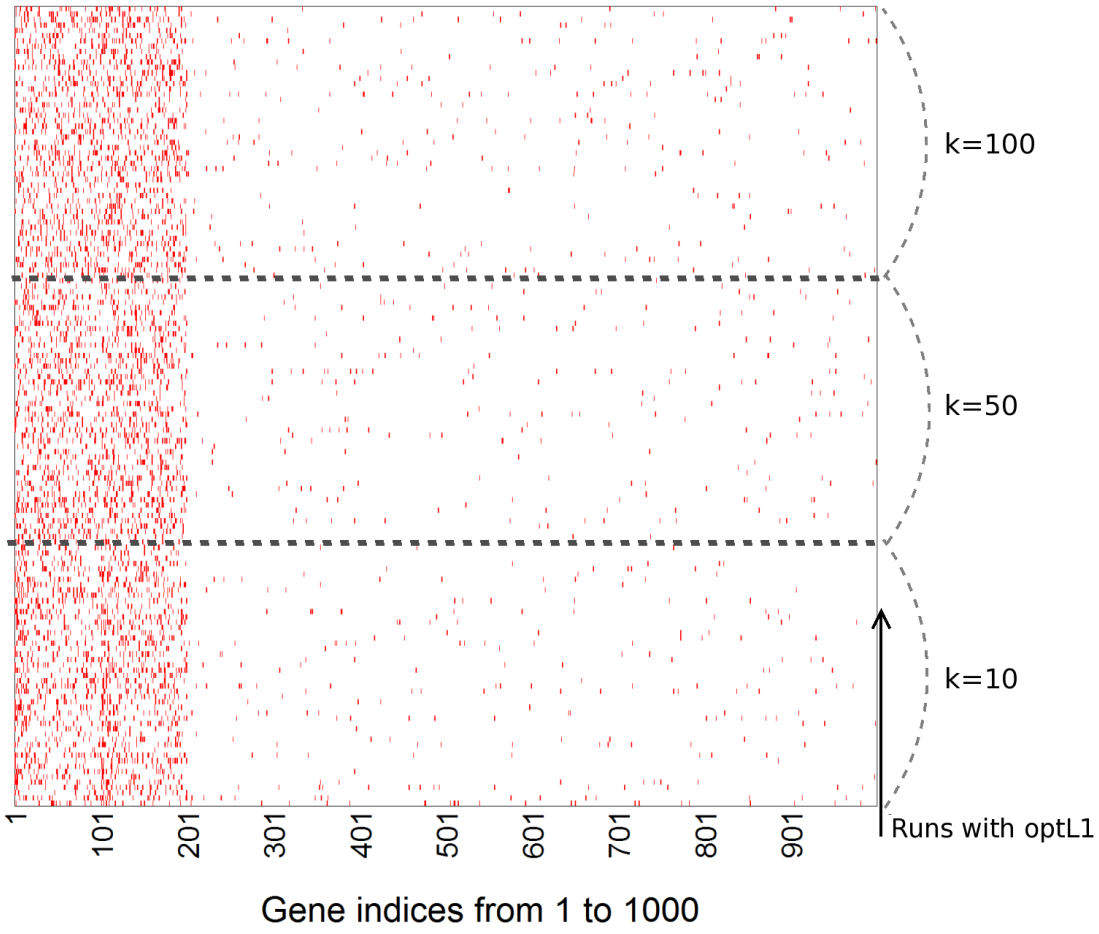

Figure 4.12: Speed of the gene effect reduction, indices. $n = 300$ *and* $\rho = 0.3$. *LOOCV. This heat map shows the genes with non-zero coefficients after running* `optL1`. *Each row is a new dataset and a new run with* `optL1`.

Then the coefficients smaller than a limit `cutoff` were reduced to zero, excluding these genes from the model. The new set of coefficients of *all* genes form the basis of this calculation:

$$\text{sensitivity} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{4.1}$$

$$\text{specificity} = \frac{\text{True Negatives}}{\text{False Positives} + \text{True Negatives}} = \frac{\text{TN}}{\text{FP} + \text{TN}}. \tag{4.2}$$

TP is the number of real relevant genes which have a non-zero coefficient, FN are the real relevant genes with coefficient zero, FP are the irrelevant genes which have a non-zero coefficient and TN are the irrelevant genes with coefficient zero. The terms are summarised in Figure 4.13. Sensitivity is also called

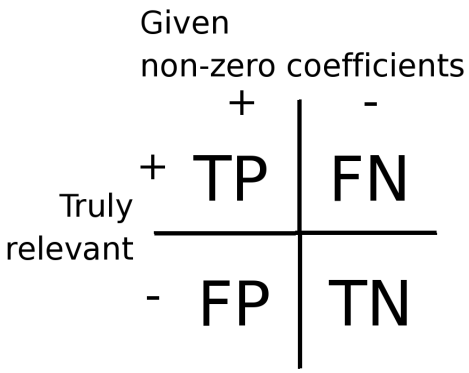

Figure 4.13: ROC grid.

True Positive Rate (TPR), and specificity is 1−False Positive Rate (FPR). The relation between these numbers are visualised in an ROC (Receiver Operating Characteristic)[13] in Figure 4.14.

The ROC curve is based on simulated data with simulation parameters $n = 300$, $\rho = 0.3$, $p = 1000$ (genes) and $k = 10$. The relevant genes were therefore defined to be the genes with indices 1-20 and 101-120. `optL1` with 10-fold CV was run on the data set, λ decreased by 10, which resulted in a prediction model. To more easily see how well `optL1` picks out the real relevant genes among the first 100, the coefficients of the genes with indices greater than 100 were ignored.

The set of 100 coefficients formed the basis of the calculations described in (4.1) and (4.2). The cutoff-limits were set to 0, 0.025, 0.05, 0.075, 0.1, 0.125, 0.15 and 0.175, forming a new predictive model every time. For every new model, sensitivity and specificity were calculated. 10-fold CV and cutoffs were repeated several times, and the average of sensitivities and specificities were calculated. This again was applied on many data sets and the averages were calculated. These values are presented in the ROC.

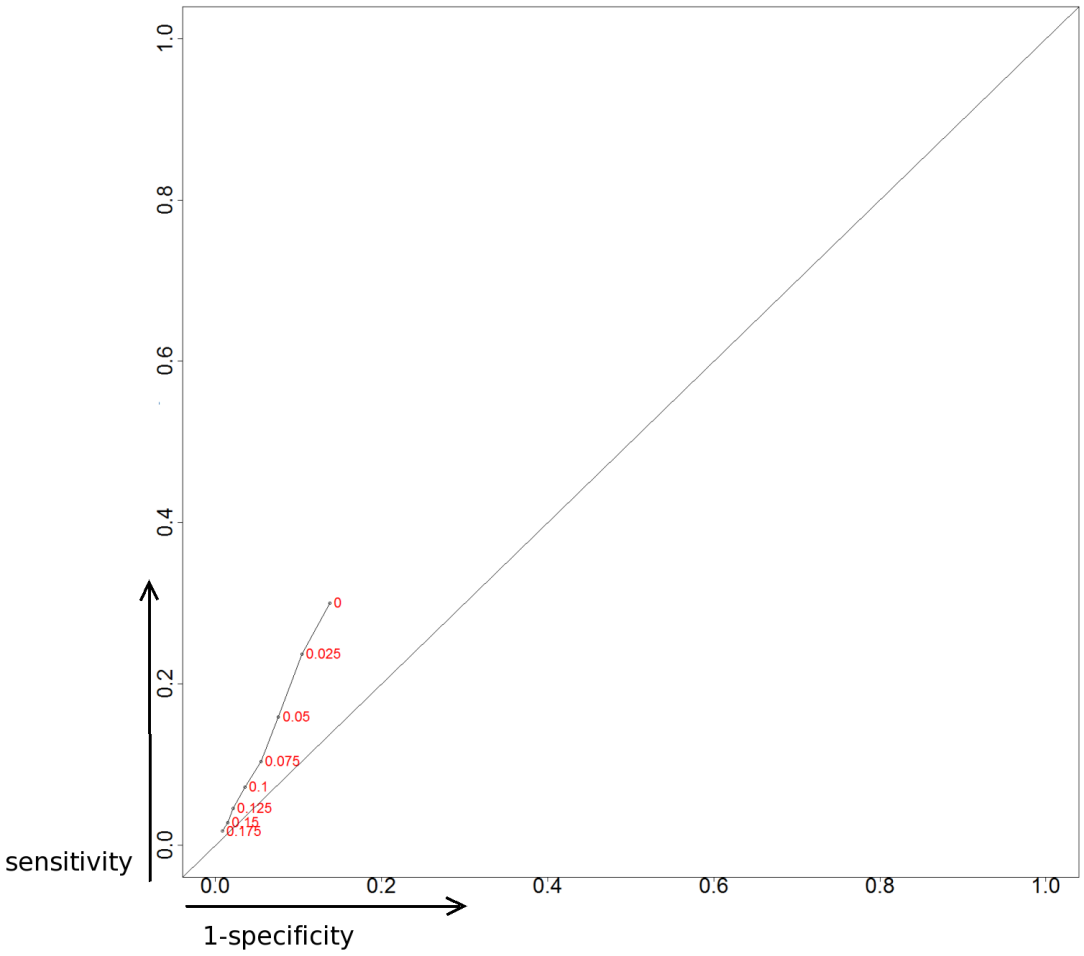

Figure 4.14: ROC of cutoff effect when λ has been decreased by 10. *The x-axis describes (1-specificity) which is equivalent with FPR, and the y-axis describes the sensitivity which is equivalent to TPR. If a model is well fitted, TPR should be large and FPR small. The models are marked by circles in the plot with cutoff-valued label. Markings above the diagonal line are well fit.*

The perfect model would have only the real relevant genes and no irrelevant genes, i.e. perfectly classified, which would be represented by a point in the ROC at (0,1). If genes were picked at random, the corresponding points would be placed on the diagonal line $y = x$. I.e. that points above the diagonal line represent a good prediction, and points below bad predictions, even worse than random. The distance to the diagonal reflects the quality of the predictive model: The point (0,1) is furthest away, positively, and is perfectly classified. A model with all coefficients non-zero gives FN=0 and TN=0 which gives sensitivity 1 and specificity 0. This would give the point (1,1). A model with only zero-coefficients will result in the point (0,0).

The ROC shows that the models lie close to a random guess. Of course, the size of the coefficients are not considered, which means that an irrelevant gene with non-zero coefficient is weighted equally as a relevant gene with non-zero coefficient. Hopefully, removing genes with very small coefficients from the model (reducing the coefficient to zero) would lead to better predictive models, but this does not seem to be the case.

Notice that even after reducing the number of genes in focus to 100, because of the sparsity of Lasso, TN will always be large, which implies specificity ≈ 1, i.e. the points will have x-coordinate close to 0, which is good. Also, it seems that FN will be large compared to TP, so sensitivity and the y-coordinate will be close to zero. Reflecting on this, it is not so strange that the points in Figure 4.14 lie close to (0,0).

4.5 Program systems

4.5.1 `penalized`

Checking for global minimum error

As discussed in Section 2.3, the function `optL1` in the R package `penalized` is imperfect. The search for the best λ can result in a λ that gives a local least error. To avoid this, the function `profL1` can be used to check the behaviour of the likelihood and verify the chosen λ. Given a data set, this comparison is interesting:

1. Run as normal with just `optL1`. Which genes have non-zero coefficients in the proposed gene signature?

2. Check with `profL1` and force the λ in `optL1` to be the λ resulting in global least error. Which genes are in the signature now?

These tests were run on a data set and are presented in heat maps in Figure 4.15.

Notice how some rows (2, 5) are blank after adjusting for global minimum error. This is because `optL1` originally chose a local λ, while `profL1`'s choice is a

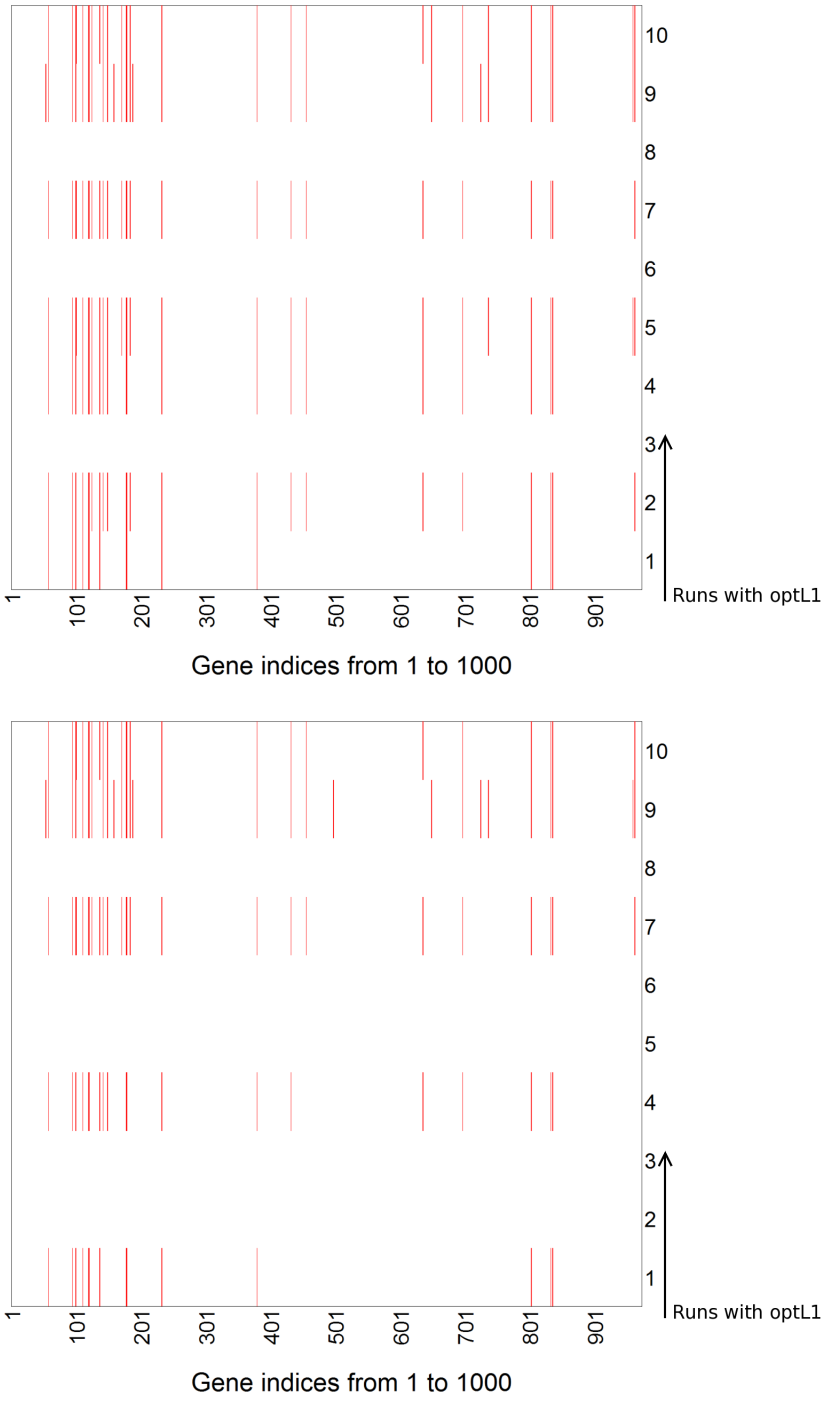

Figure 4.15: Checking for global minimum error. *Heat maps corresponding to the aspects discussed in Section 4.5.1. Both heat maps are based on the same data set, but every row is a new run with* `optL1`*.* ***Top:*** *Run as normal.* ***Bottom:*** *Replace λs that give local minimum error with global λ. For more examples, see Figures A.14 and A.15.*

very large λ that includes no covariates in the predictive model. Some rows show no change, whereas some show small change, e.g. rows 4 and 10.

When `optL1` and `profL1` disagree about the choice of λ, one function does not necessarily choose a larger signature than the other. The difference in number of non-zero coefficients in such cases are sometimes in favour of `optL1` and sometimes in favour of `profL1` even though from Figure 4.15 it may seem that `profL1` chooses smaller signatures because of the blank lines.

Does `optL1` pick out the correct genes, and what is LOOCV's suggestion?

The behaviour of `optL1` for varying λ is shown in Figure 4.16. The gene effects are here set to constant such that genes with indices 1-20 and 101-120 are one, and the rest zero. The genes with effect one are named "correct" whereas the rest are "incorrect". Note that each subfigure is based on just one data set.

From Section 4.2 we know that more observations lead to more correct results. Figure 4.17 shows the corresponding plots to Figure 4.16, but the number of observations has been increased to $n = 300$. Remember that there may be large differences between data sets, because the predictive result depends on data set properties as seen in Section 4.3, so be careful when comparing the plots.

The line that represents the correct genes is nearly constant, especially when n is large, but the number of incorrect genes decreases as λ increases. LOOCV's choice of λ in Figure 4.16 is small if focusing on the relation correct vs. incorrect genes. λ could have been increased just a little bit, and would have had the same number of correct genes, but fewer incorrect genes. This may be unique for these particular data sets, but LOOCV does tend to choose a small λ[26]. A suggestion for solving this problem is discussed in Sections 2.4 and 6.2.

4.5.2 `glmnet`

How well does `glmnet` pick out the genes with strong effects compared to `penalized`?

Data was simulated in the same way as for `penalized`, but was run through the R package `glmnet` instead. k was set to 10, so the genes with indices 1-10 and 101-110 were expected to show up more often in the gene signatures. But, there seemed to be more noise than for the results of `optL1`,shown in Figure 4.4. The `glmnet` results are shown in the left hand side of Figure 4.18. A direct comparison of the two R functions when $n = 100$ is not recommended. The data set is too small, so any differences may be a result of randomness. However, by increasing the observation size to $n = 300$, the expected shape shows up, see the right hand side of Figure 4.18.

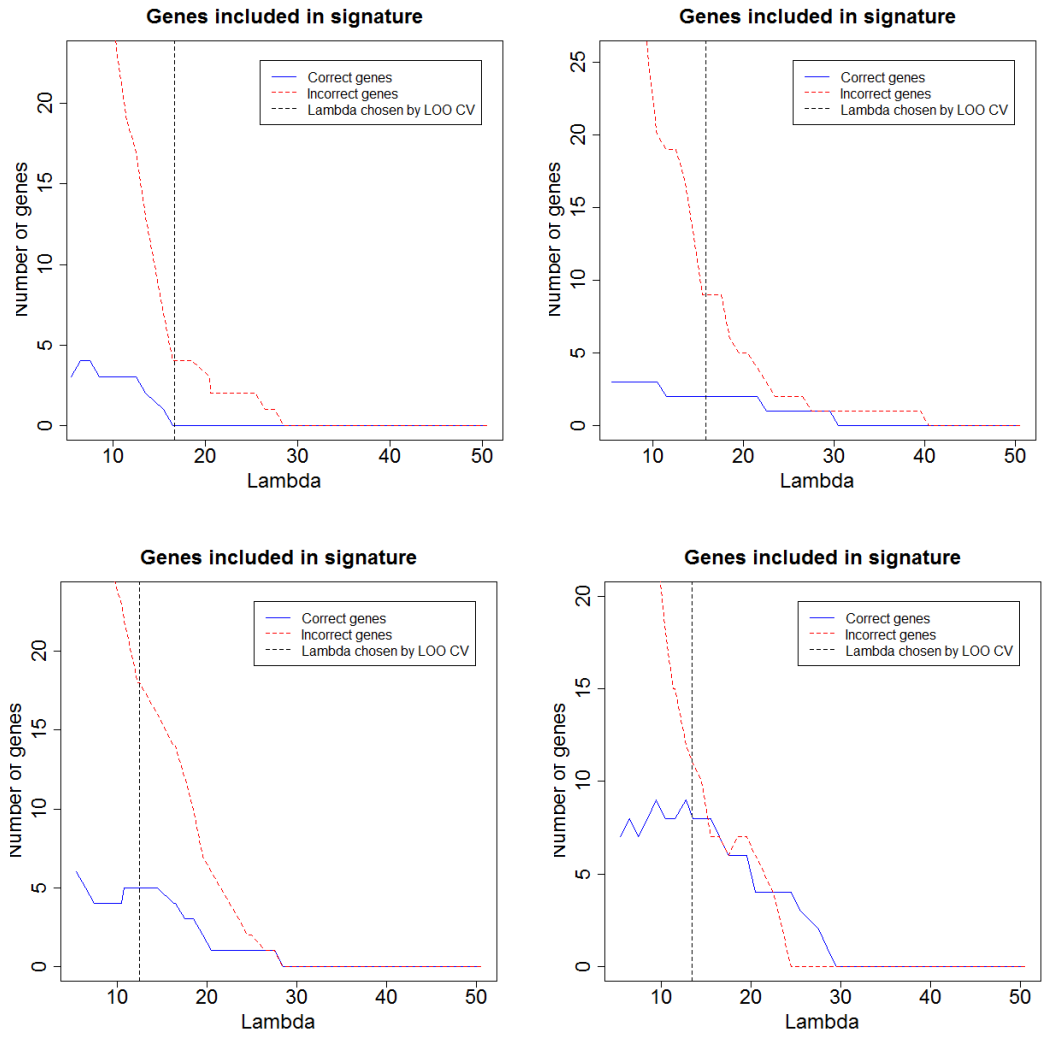

Figure 4.16: Visualisation of how well `optL1` picks out the correct genes for varying λ. *The simulation parameters are set to $n = 100$, $\rho = 0.3$ and k constant s.t. genes 1-20 and 101-120 have effect of size one. Each subfigure is based on one data set.*

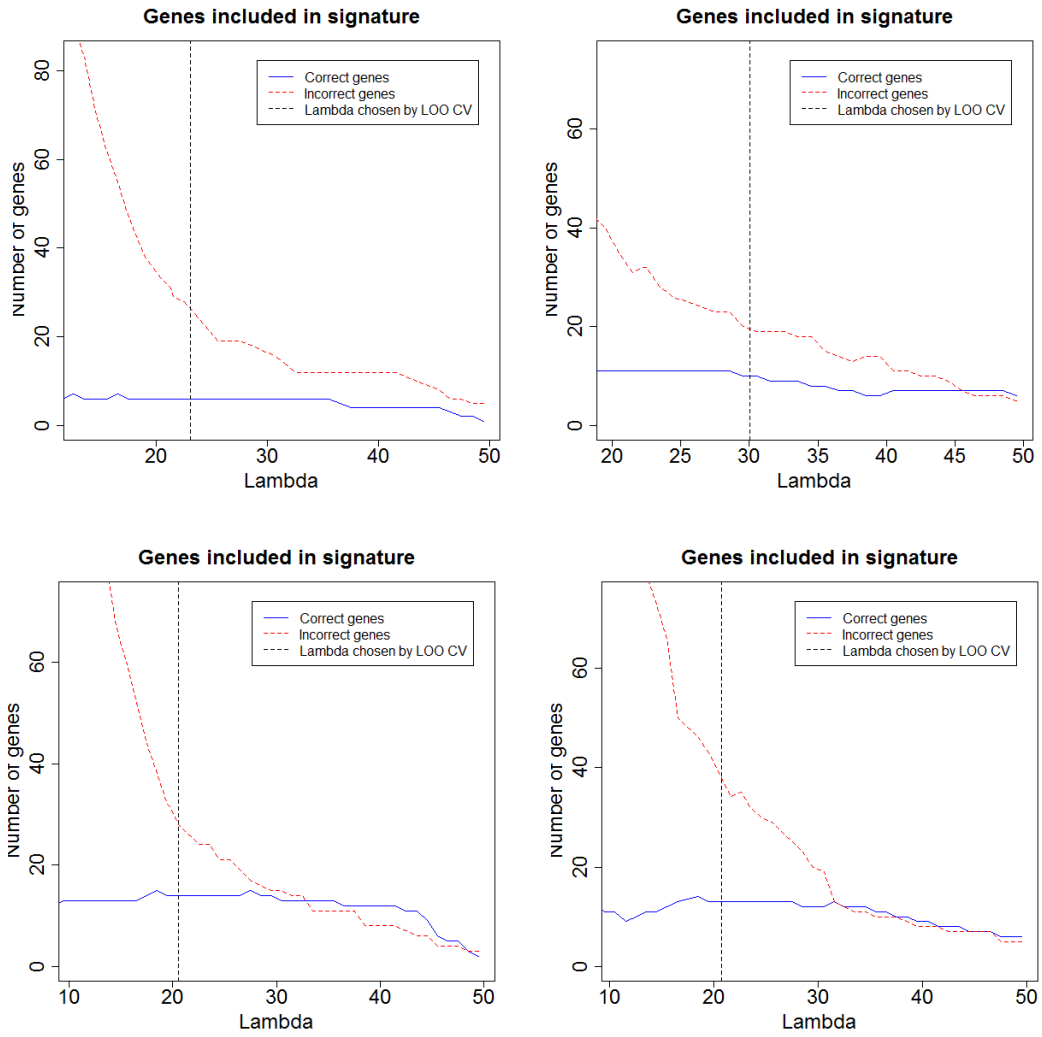

Figure 4.17: Visualisation of how well `optL1` picks out the correct genes for varying λ, $n = 300$. *Figure corresponding to Figure 4.16. The number of observations has been increased to $n = 300$, the other simulation parameters as before.*

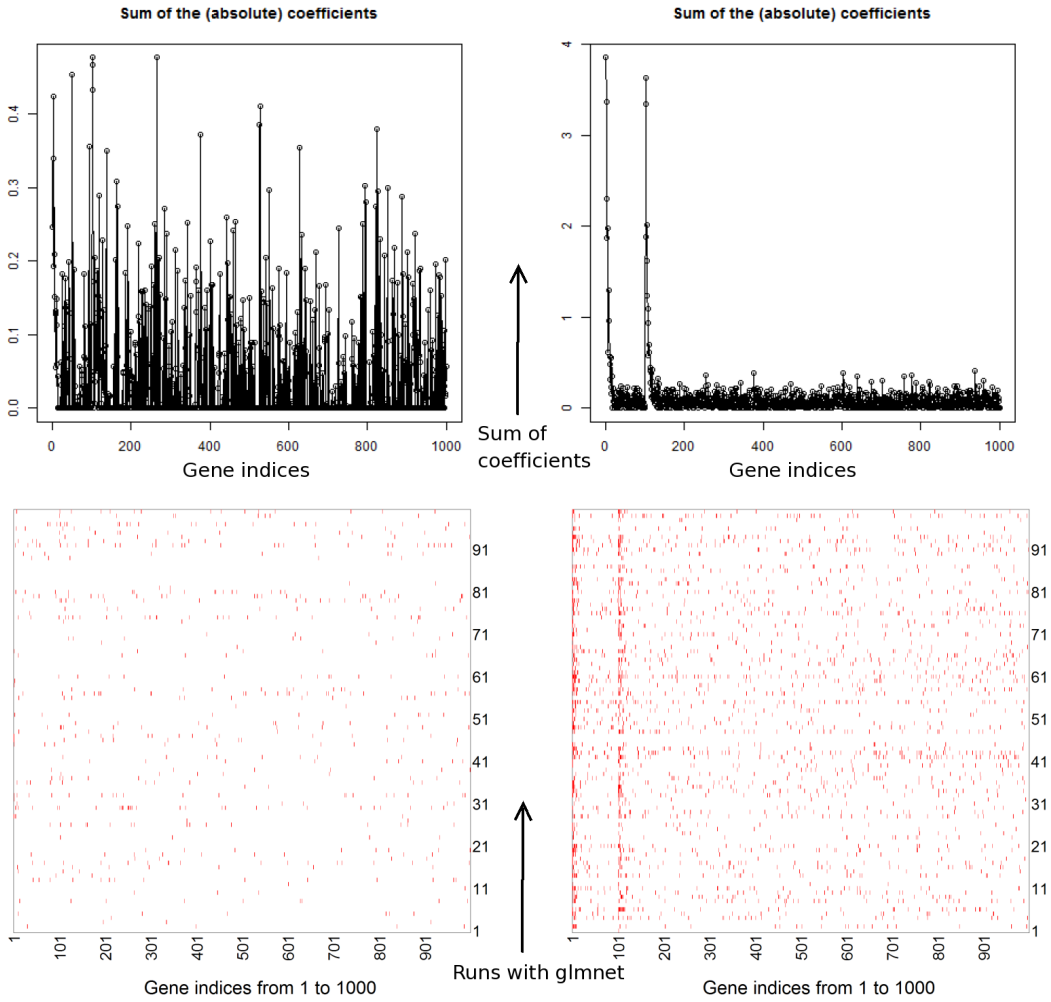

Figure 4.18: Glmnet, LOOCV. *The two plots to the left are based on data with $n = 100$ observations. The plots to the right are based on data with $n = 300$ observations. For larger images, see Figures A.10 and A.11.*

4.5.3 Do `glmnet` and `penalized` agree?

Both `glmnet` and `penalized` use the same approach to find the best prognostic gene signature: Lasso and CV. The results of LOOCV of both methods are shown in Figure 4.19. The simulation parameters of the data sets in this computation were set to: $n = 300$, $k = 10$, $\rho = 0.3$.

The black elements in the heat map represent a gene that both `penalized` and `glmnet` gave a non-zero coefficient. These are frequent, but there are also green and pink elements present. These represent genes that only `glmnet` or only `penalized` gave non-zero coefficients, respectively.

The presence of many black elements are reassuring as it means that the two methods mostly agree. The number of green elements seem to exceed the number of pink elements, which indicate that `penalized` tend to pick more sparse models than `glmnet`.

Also, CV chooses the best penalty parameter λ, but the underlying model should be the same every time, which causes all smaller signatures to be contained in the larger ones, as seen previously. One would think that the difference between the two R packages is the CV process. However, LOOCV eliminates this factor and some rows in Figure 4.19 contain both pink and green elements which mean that `penalized`'s signature is not contained in `glmnet`'s, or the other way around. There must be other differences than CV.

From the documentation[22][24] it seems that λ is identically defined in both packages. Forcing the λ to be equal[2] in either function results in different signatures, and results in a heat map alike Figure 4.19.

4.5.4 Summary

The risk of choosing a λ from a non-global minimum point on the error curve is considerable, as can be seen in Figure 4.15. To achieve a reliable result, this must be taken into account and it should be made sure that the chosen λ gives the global least error. Even though `optL1` converges and settles, verification with `profL1` should be performed, unless we will risk ending up with a too sparse or too large set of genes with non-zero coefficients.

Also, as shown by comparing `penalized` and `glmnet`, the result depends on the R package applied. This may indicate that the final results from an analysis preferentially should be checked.

[2]The values of λ are different in the two packages. `glmnet` uses values between 0 and 1, whereas `penalized` uses values approximately between 15 an 40. Computations show that the relation is $105 \cdot \lambda_{glmnet} \approx \lambda_{penalized}$.

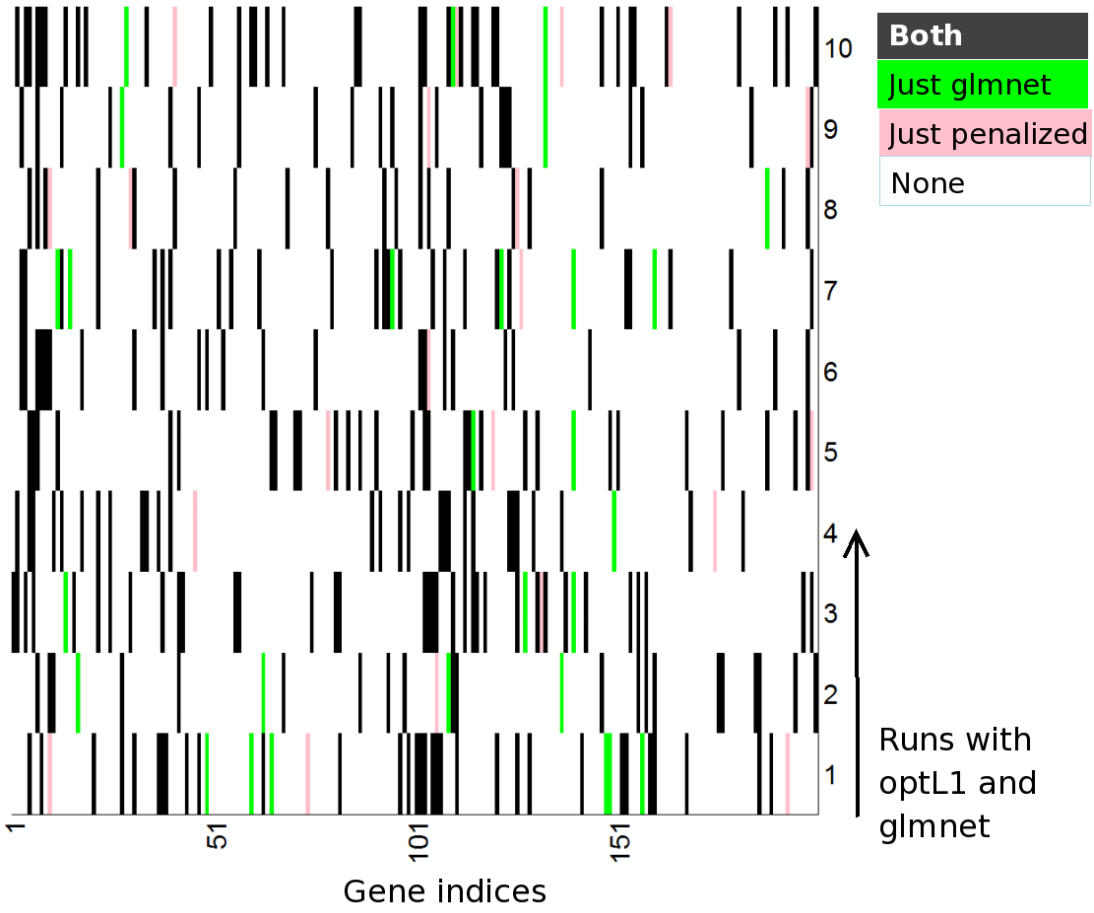

Figure 4.19: Comparison of `glmnet` and `penalized`. *Each row represents the signatures chosen by LOOCV with* `glmnet` *and* `penalized`*, based on the same data set with 310 observations. The columns correspond to the genes, as described in Figure 3.4, but only the genes with indices 1-200 are displayed here. All 1000 genes can be seen in Figure A.16.* ***Black:*** *Both* `penalized` *and* `glmnet` *have given this gene a non-zero coefficient.* ***Pink:*** *Only* `penalized` *has given this gene a non-zero coefficient.* ***Green:*** *Only* `glmnet`*.*

4.5.5 p-value and variance filter

In Sveen et al.'s paper[1], a filter was applied before running the data trough CV and Lasso. To reduce the data set size, they ran univariate Cox proportional hazards analyses on the gene expression data and obtained Wald p-values. Sveen et al. also calculated the variance of all genes. The genes with variance less than 0.2 and p-value greater than 0.5 were left out in the continued investigation as they were considered uninteresting and probably irrelevant.

The p-value filter is probably applicable on simulated data as well. The variance of our simulated data is restricted by the parameter `alt` which is set to one for all simulations. This choice results in a gene expression matrix where every column has variance approximately equal one. The first genes have no different variance from the other genes, so application of a variance filter would be useless.

To see how filtering by eliminating genes with low p-values affects the simulated data, two sets of images were produced, shown in Figure 4.20. There is no great difference. The maximum points are similar, and there seems to be equally much noise, i.e. false positives.

Even though the filter may have a positive effect on the simulated data as well, it was not generally applied in the investigations in this thesis. This way there are fewer factors that may affect the results.

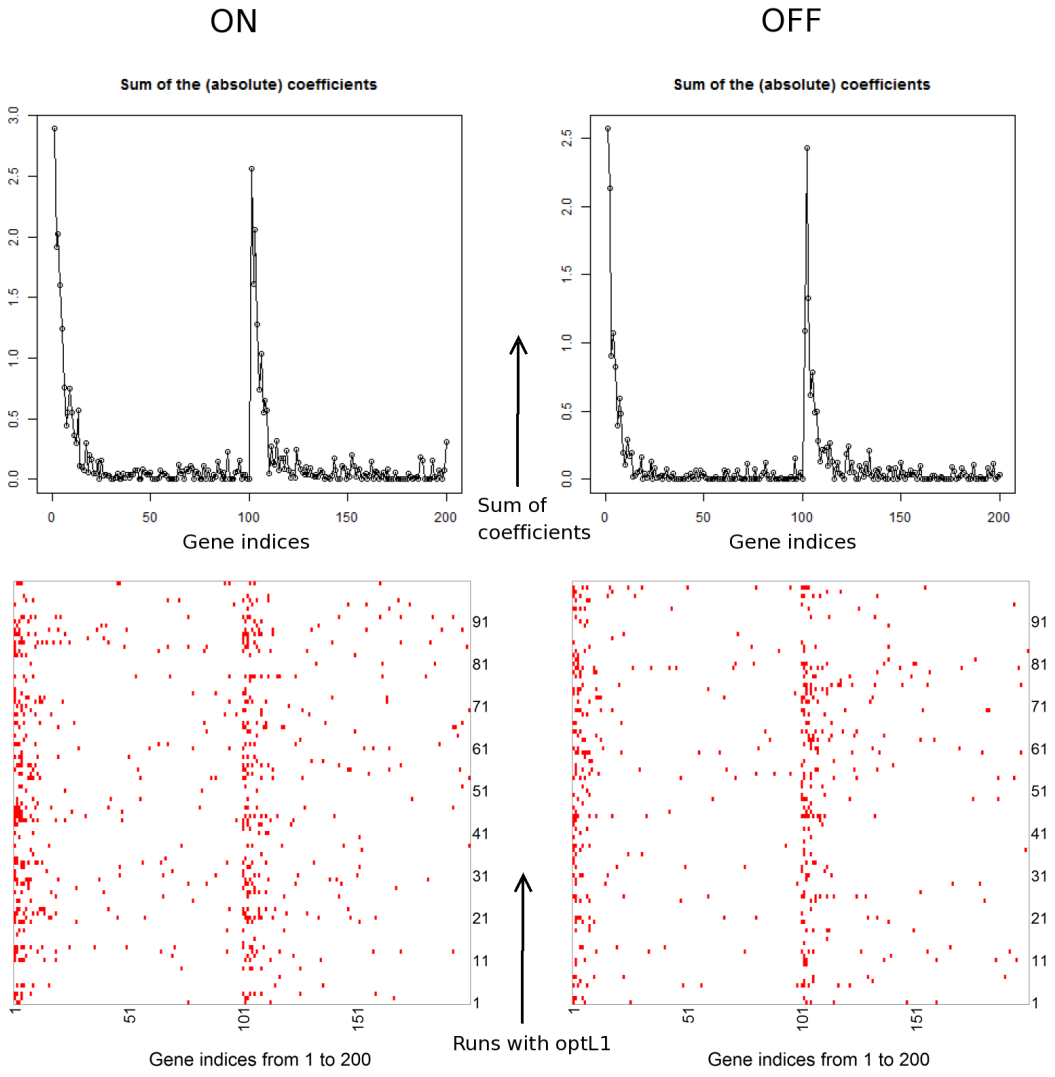

Figure 4.20: Filter on and off. *These plots were generated after simulating data with zero correlation, 300 observations and* $k = 10$. ***Left:*** *The p-value and variance filter is switched on.* ***Right:*** *The filter is switched off. The plots show the distribution of the 200 first genes in the fits resulting from* `optL1` *in* `penalized` *with 10-fold CV. The density plots show the sum of the coefficients and the heat maps show which genes are included in each signature. Every row represents one signature, generated from different data sets.*

Chapter 5

Estimation of Accuracy

The varying results of `optL1`, with the same input data set, is caused by K-fold CV. When there are as many groups as there are observations, i.e. LOOCV, the result is identical from run to run.

The instability is worrying because it is difficult to propose a single prognostic gene signature, which would be the easiest to interpret. But, the hierarchy of signatures, where each smaller signature is contained in the bigger ones, provides an opportunity to test several models. Sveen et al.[1] utilises this hierarchy and the three available data sets to settle for one model. The first data set was the training set which is the foundation of the model. The next set, test set 1, was used to test the models suggested by step one and find the best model. The third data set, test set 2, was the validation set. If LOOCV had been used, they would not have had the range of models to test, and Sveen et al.[1] may have settled on a different model.

5.1 The range of signatures

The varying result of K-fold CV where $K \leq n/2$ can be used in our benefit. A wide range of signatures where no signature appears more often than others indicate that it is difficult to say which is the better prognostic signature. On the other hand, a more narrow range, in the extreme case a unique result, indicate a robust result. Thus, while LOOCV could be used to select the model, the CV-process may indicate the robustness of the result.

Application on real data

Ten-fold and four-fold CV were applied on the CRC data[1] described in the introduction. The resulting signature size bar diagrams are shown in Figure 5.1. The tendency from Section 4.1.3 reappears here: Four-fold CV gives some larger signatures than ten-fold CV, but there are also more cases of signatures with sizes as small as three to five.

Four-fold CV is not more stable, however, which is our main interest. In fact, there are fewer peaks in the results of 4-fold CV than in 10-fold CV. The latter favours the signatures of sizes 0-2, 8 and 10-12, whereas considerably many of the 4-fold CV signatures are of other sizes.

LOOCV on the CRC data set results in a signature of size 2, i.e. if the chosen λ had been smaller, the signature would contain more genes. The chosen value of λ is actually not the parameter value that gives global minimum error: A `profL1` run confirms this. The actual optimal λ gives a signature of size 10. It may be that the K-fold CVs presented in Figure 5.1 are inclined to choose a suboptimal λ which results in the very high bars at sizes 0-2 in the bar diagrams, see Section 6.3.1.

The effect of checking with `profL1` may be great, as with the CRC data, therefore it may be sensible to always verify the result of `optL1` with `profL1` to be sure of the quality of the predicitve model.

5.2 Bootstrapping

There exists several methods for estimation of accuracy which may be applied to survival data. One of these methods is the bootstrap[31].

The bootstrap constructs data sets based on the observed data set, which consists of data for each observation: $\{(y_1, \mathbf{x}_1, \delta_1), \ldots, (y_n, \mathbf{x}_n, \delta_n)\}$, where y_i is the observed survival time for observation i, $\mathbf{x}_i$ are the observed covariates (e.g. one row in a gene expression matrix) and δ_i is the censoring indicator where 1 is an event, and 0 is a censoring. A bootstrap data set consists of n such elements where each element is drawn randomly from the observed data set with replacement.

The process of generating data sets and modelling is repeated numerous times to describe the accuracy of the model trained on the real data. Efron and Tibshirani[31] describe a Cox example based on leukemia remission times in mice. The study is old, it consists of few observations and there is just one covariate included in the regression model, so it is not directly applicable to the CRC data[1]. However, the method to find the best $\hat{\boldsymbol{\beta}}$ is similar: the $\hat{\boldsymbol{\beta}}$ that maximises the partial likelihood, as seen in (2.14), is chosen.

Now, how accurate is the estimation of $\hat{\boldsymbol{\beta}}$? The partial likelihood is maximised based on each new data set, and the choice of $\hat{\boldsymbol{\beta}}$ is taken note of. The resampling and maximisation of the likelihood is repeated e.g. 1000 times, and we can make a histogram for each $\hat{\beta}_i$. The form of the histogram will reflect the accuracy of the value of $\hat{\beta}_i$ and whether its distribution is right or left skewed, or unskewed.

Adjustment to penalised Cox regression should be straightforward: The penalised partial log-likelihood replaces the partial likelihood described in the general bootstrapping case.

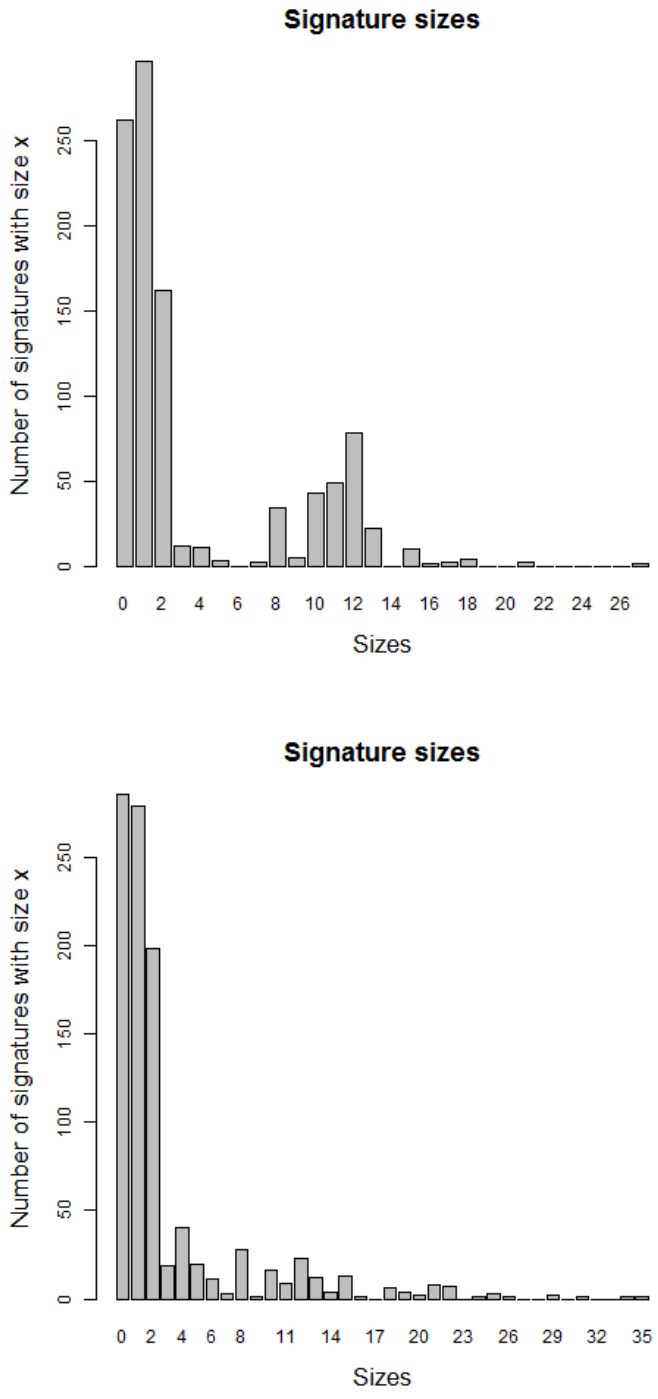

Figure 5.1: 10- (top) and 4-fold (bottom) CV applied on CRC data. *`optL1` was run 1000 times with the CRC data set as input. The sizes of the resulting gene signatures are represented in these bar diagrams.*

Two methods for censor simulations are described in Efron and Tibshirani's paper[31]: One where the underlying distribution of the survival times is used, and another where the censor indicator is ignored. They chose to ignore the censoring indicator of each observation when resampling. For more detailed explanation, see [31].

5.3 ROC curve

An ROC curve, such as described in Section 4.4, can be generated from prediction models. If the points lie close to or below the diagonal, the models do not have a good predictive value; in fact, a randomly chosen model would be a better choice. Note that the complementary models of the models beneath the diagonal give better predictions than the original model. When discussing genes as in the examples of this thesis, instead of giving the proposed genes non-zero coefficients, reduce them to zero and give the genes that were not chosen non-zero coefficients. This model will lie on the opposite side of the diagonal and hence be of better predictive value.

Visualisation of the chosen fit on real data by an ROC curve is a very difficult, even impossible, task. To produce an ROC, the real relevant and irrelevant genes must be known, but they are what is sought when applying Lasso and CV on real data, and therefore unknown.

Chapter 6

Discussion

6.1 Summary and conclusions

The instability of K-fold CV is worrying, although potentially useful. Ideally, a unique result would help by choosing one prognostic signature, but the range of signatures suggested by K-fold CV can indicate the accuracy of the prediction.

It is not obvious that LOOCV should be used to select the final model. It seems that a small K in K-fold CV increases the sizes of the signatures. This is also the case when there are more observations available. It has been suggested that 4-fold CV is the best choice for K-fold CV[30].

Increasing the number of observations is difficult. Simulations and common sense show that having more observations usually generates more stable predictive results and increases the predictive accuracy. In real life we must wait for more cases of the condition in interest to appear or settle with the data set available. As few as a hundred observations is common in survival studies. However, our simulations underline that this is a low number for studies with large numbers of covariates.

Properties of the survival data such as correlation and strength of the gene effects are impossible to control in real data. However, it is important to be aware that these properties affect the prediction result. E.g. if there is strong correlation in some groups of genes, the Lasso will probably choose one or none genes in the group to include in the predictive model. Depending on the application, it may thus be useful to apply a different regularisation method.

There are several ways to investigate the accuracy and the predictive power. Established methods include both ROC curves and bootstrapping. Bootstrapping can describe how well we have estimated the values of the coefficients. The ROC can describe the how well we have sorted real relevant genes and irrelevant genes . Additionally, the method described in this thesis is to evaluate the range of signatures suggested by Lasso and CV.

6.2 Relation to other recent work

The reasoning for percentile-Lasso, described in Section 2.4, matches the results in this thesis. An illustration in Robert's paper[26] is similar to Figures 4.16 and 4.17. When $n = 300$, the line describing the number of correct genes given non-zero coefficients is approximately constant. The line of incorrectly given non-zero coefficients is decreasing. Thus, by choosing a larger λ than LOOCV's choice, we would mostly be left with fewer "incorrect" covariates and the same number of "correct" covariates.

6.3 Discussion of results

6.3.1 My contributions

K-fold CV instability

Some properties, e.g. how Lasso picks one or none in a group of correlated covariates[17], was known from earlier studies. However, the display of how the K in K-fold affects the prediction result is new. There have been arguments for choosing K=4[30], and other Ks for different types of data[32]; The main argument against choosing LOOCV is that every training set is nearly equal, because only one observation is excluded every time, and therefore strongly depend on each other. This may not train models well for a general scenario, but instead makes it good for the specific data set. With $K \leq n/2$ the differences between the training sets are greater, so the model will be better suited for application on other data[30].

In our case, where the CRC data[1] has been the motivation, the multiple results with repeated 10-fold CVs have proven useful. Even though LOOCV gives a unique result, K-fold CV when $K \leq n/2$ provides useful information about the certainty of the predictive model.

LOOCV sparsity

A large K of K-fold CV tends to give smaller signatures than small Ks, i.e. LOOCV gives the fewest number of covariates with non-zero coefficients. It has been discussed that LOOCV's choice of λ usually is too small[26]. A larger λ would give less covariates in the model, but mostly the "incorrect" genes would be reduced. The number of "correct" genes would remain approximately the same. Therefore, the increasing signature size when K decreases, i.e. it chooses a smaller λ, encourages choosing K large.

More observations

More observations usually give more accurate predictive results, which is well known. My computations confirm this and show that increasing the number of observations from 100 to 300 gives a much less varying result. In fact, the instability when there are only 100 observations available is so apparent that it may lead to a conclusion indicating that $n = 100$ observations are too few to state an association between gene expression levels and survival.

Another effect of having more observations is that the suggested prognostic signatures tend to be larger when n increases, and more observations are thought to give better precision. Contrary to this observation, the previous section "LOOCV sparsity" encourages us to choose a smaller signature. Both are useful observations and underline the importance of being aware of the influence of these factors.

Decreasing λ and reducing small coefficients to zero

The increasing signature sizes when n is large encourages the attempt to decrease λ to include more covariates in the model, but then reduce the very small coefficients to zero. This was attempted in this thesis and the result visualised by an ROC curve. The curve was discouraging, but there may be a better way to illustrate the effect of reducing coefficients to zero. It is known that Lasso and CV do not always suggest the best λ[26], so the idea of an alternative to the ordinary method is appealing.

The risk of choosing a suboptimal λ

Sveen et al.[1] gave a recipe of how to apply the R package `penalized` on CRC data. They did not mention the risk that `optL1` may choose a λ that gives a local minimum error.

This risk is considerable as shown in this thesis and must be taken into consideration when using the function. We can save time by not checking with a function such as `profL1`, but time is usually not an issue in these cases.

Specific data set properties?

The motivation of this thesis was based on the CRC paper[1] and the question "Is the method dependent on details of the data set?" was raised. Control of simulation parameters such as correlation, number of observations and strength of gene effects has shown that the instability is present for all types of data. Therefore, the instability in the mentioned CRC paper is not unique, and cannot be caused solely by properties of the CRC data.

6.3.2 Weaknesses

Some computations, such as the information illustrated in the heat maps in Figures 4.1 and 4.6, where the variation of K-fold within one data set is shown, are based on only one data set. This varying behaviour depends on different factors, such as simulation parameters and the number of observations. Thus, the claim of varying behaviour within one data set does not necessarily hold for all other data sets.

As mentioned in the previous section, the risk of choosing a λ that gives a local instead of the global minimum error is considerable, as shown in Section 4.5.1. This thesis has directed attention to this risk, but has ignored this scenario in other calculations. Therefore, the other results may have looked somewhat different if the λs had been corrected by e.g. `profL1`.

6.4 Further work

Several of the weaknesses listed in the previous section deserves attention, and some questions have been left unanswered, such as: Will correcting for global minimum error change the result of the CRC data, as discussed in Section 5.1? and: In Section 4.1.3, the claim that more folds, i.e. larger K of K-fold, give more sparse prognostic signatures was confirmed. The question "Will smaller ks give more stable results?" was raised. These issues among others would have been interesting to look further into.

The idea of reducing very small coefficients to zero has not been thoroughly tested. A better way of illustrating the predictive results than an ROC curve should be found and an alternative Lasso method may be developed in some way.

A better and more detailed mapping of the differences between `penalized` and `glmnet` is missing. This thesis touches upon some of the differences, but no proper documentation of e.g. what type of CV the two packages apply could be found. Exactly what causes the different prediction result is therefore left out, but should be looked further into, e.g. by comparison of source code.

Percentile-Lasso was presented based on information gathered from[26], published online in September 2013, so it is fairly new. The proposed method of increasing the penalty parameter λ is interesting and should be tested on data such as the CRC data[1] discussed in this thesis.

An idea of a function of n to describe false positives when searching for the real relevant genes was proposed in Section 4.2.2, i.e. how many irrelevant genes are classified as relevant with non-zero coefficients. Finding such a function would be useful when deciding whether there are enough observations available to draw a conclusion.

All the factors that affect Lasso and CV stability presented in this thesis have been tested more or less separately. A combination of factors, e.g. how varying

correlation affects the differences between K-fold CV with different Ks, is also of interest. Especially the effect of always using the correct (global) penalty parameter λ is an interesting topic for further studies.

Bibliography

[1] A. Sveen, T.H. Ågesen, A. Nesbakken, G.I. Meling, T.O. Rognum, K. Liestøl, R.I. Skotheim, and R.A. Lothe. Cologuidepro: A prognostic 7-gene expression signature for stage iii colorectal cancer patients. *Clinical Cancer Research*, 2012. *Note: The thesis is based on a draft from 2012, and the temporary title of this paper was "Prognostic stratification of stage II and III colorectal cancer patients by a 7-gene expression signature".*

[2] J.L. Devore and K.N. Berk. *Modern Mathematical Statistics With Applications*. Duxbury Pr, 2006.

[3] World Health Organization International Agency for Research on Cancer. Globocan 2008 estimated cancer incidence, mortality, prevalence and disability-adjusted life years (dalys) worldwide in 2008. `http://globocan.iarc.fr/`. Accessed: 11-10-2013.

[4] S.C. Larsson and A. Wolk. Meat consumption and risk of colorectal cancer: A meta-analysis of prospective studies. *International Journal of Cancer*, 119(11):2657–2664, 2006.

[5] Cancer Reasearch UK. Bowel cancer: Treating bowel cancer: Treatment types: Types of treatment. `http://www.cancerresearchuk.org/cancer-help/type/bowel-cancer/treatment/types/which-treatment-for-bowel-cancer`. Accessed: 11-10-2013.

[6] E. Van Cutsem, A. D'Hoore, C. De Vleeschouwer, J. Decaestecker, and F. Penninckx. Colon cancer: Management of locoregional disease. *Principles and Practice of Gastrointestinal Oncology*, pages 581–590.

[7] L.C. Bergersen, I.K. Glad, and H. Lyng. Weighted lasso with data integration. *Statistical Applications in Genetics and Molecular Biology*.

[8] J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *J Stat Softw*, pages 1–22.

[9] R.J. Brooker, E.P. Widmaier, L.E. Graham, and P.D. Stiling. *Biology*. McGraw-Hill, 2008.

[10] `http://commons.wikimedia.org/`. Accessed: 28-11-2013.

[11] H.L. Størvold. Estimering av en straffet cox-modell på bakgrunn av mikroarray genekspresjonsdata. Master's thesis, University of Oslo, 2004.

[12] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58:267–288, 1996.

[13] T.J. Hastie, R.J. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, New York, second edition, 2009.

[14] H.H. Zhang and W. Lu. Adaptive lasso for cox's proportional hazards model. *Biometrika*, pages 1–13, 2007.

[15] O.O. Aalen, Ø. Borgan, and H.K. Gjessing. *Survival and Event History Analysis.* Springer, 2008.

[16] R. Tibshirani. The lasso method for variable selection in the cox model. *Statistics in Medicine*, 16:385–395, 1997.

[17] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67:301–320, 2005.

[18] A.E. Hoerl and R.W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):pp. 55–67, 1970.

[19] A.Y.C. Kuk. All subsets regression in a proportional hazards model. *Biometrika*, 71(3):587–592, 1984.

[20] P.J.M. Verweij and H.C. Van Houwelingen. Cross-validation in survival analysis. *Statistics in Medicine*, 12(24):2305–2314, 1993.

[21] J.J. Goeman. L1 penalized estimation in the cox proportional hazards model. *Biometrical Journal*, 52(1):70–84, 2010.

[22] J. Goeman, R. Meijer, and N. Chaturvedi. *penalized: L1 (lasso and fused lasso) and L2 (ridge) penalized estimation in GLMs and in the Cox model*, 2012.

[23] N. Simon, J. Friedman, T. Hastie, and R. Tibshirani. Coxnet: Regularized cox regression. 2011.

[24] J. Friedman, T. Hastie, and R. Tibshirani. *glmnet: Lasso and elastic-net regularized generalized linear models*, 2008.

[25] H. Xu, C. Caramanis, and S. Mannor. Sparse algorithms are not stable: A no-free-lunch theorem. *Pattern Analysis and Machine Intelligence, IEEE Transactions*, 34(1):187–193, 2011.

[26] S. Roberts and G. Nowak. Stabilizing the lasso against cross-validation variability. *Computational Statistics and Data Analysis*, 70:198–211, 2014.

[27] H.M. Bøvelstad, S. Nygård, H.L. Størvold, M. Aldrin, Ø. Borgan, A. Frigessi, and O.C. Lingjærde. Predicting survival from microarray data - a comparative study. *Bioinformatics*, 23(16):2080–2087, 2007.

[28] S. Nygård, Ø. Borgan, O.C. Lingjærde, and H.L. Størvold. Partial least squares cox regression for genome-wide data. *Lifetime Data Analysis*, 2007.

[29] S. Datta, J. Le-Rademacher, and S. Datta. Predicting patient survival from microarray data by accelerated failure time modeling using partial least squares and lasso. *Biometrics*, 2007.

[30] M. Markatou, H. Tian, S. Biswas, and G. Hripcsak. Analysis of variance of cross-validation estimators of the generalization error. *Journal of Machine Learning Research*, 2005.

[31] B. Efron and R. Tibshirani. Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. *Statistical Science*, 1(1):54–77, 1986.

[32] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. *International Joint Conference on Artifical Intelligence*, 1995.

Appendix A

A.1 Average signature sizes

Here follows the average signature sizes for all 100 data sets from the calculations in Section 4.1.3. The columns represent the folds, named in the first row. The rows represent every data set.

"100"	"25"	"20"	"10"	"5"	"4"	"3"
0	0	0	1.2	9.1	7.3	14.2
11	10.8	12.9	14.4	17	19.6	20.4
6	6.4	6.3	9.6	21.6	24.7	22.7
6	11.8	14.5	15.9	20.8	19.2	24.2
18	17.9	19.5	17.4	22.5	19.6	21.6
11	10	10.5	10.9	13.8	12.1	16.6
17	18.1	18.2	19.7	21.9	24.2	25.8
6	5.4	9.6	9.8	10.6	16.2	18.5
9	9.9	11.8	15.2	15.8	18.9	20.2
2	3.1	3	3.1	5.4	6.5	6.8
10	12.9	14.1	14.1	18.8	22	22.8
0	7.7	8.2	8.7	16.6	18.5	27.2
6	6.9	8.3	11.5	15.1	20.1	25.8
8	8.9	9.1	9.8	12	18.2	18.2
16	20.1	20.1	20.3	22.7	26.1	28.7
0	0.2	0.2	0	3.1	3	3.7
7	8	8.9	10.7	14.1	10.3	15.1
11	10.8	11.3	11.2	13.6	18.4	18.8
10	10.4	10.8	12	20	21.2	24.6
22	21.1	21.5	22.6	20.5	23	24.6
7	8.5	9.2	11.8	14.6	11.9	17.5
9	9.7	10	12.7	14.6	17.3	21.5
16	13.3	14.7	13.2	10.4	9	7.9
7	7.5	8.1	13	21.6	16.7	18.5
13	13.2	13.6	13.6	11.8	12.9	15.1

14	11.2	14.2	11.9	16.1	12.2	24.5
13	9.1	10.5	10.1	12.9	15.2	12
0	0	0.1	3.8	3.5	4.7	11.3
12	17.4	19.7	20.1	21.3	20.2	20.1
11	14.5	15.4	15.9	16.9	20.9	19.7
21	9.5	9.3	13.5	13.5	4.6	10.4
12	11.7	11.5	12	19.2	21.5	30
12	12	13.5	13.4	12	9.3	19.7
5	7.3	8.2	13.5	15.6	17.1	21.8
11	7.8	8.6	10	8	10.9	14.5
10	10.4	11.1	10.9	13.7	17.9	25.5
8	11.2	13.6	24.1	29.9	29.6	26.1
7	8	8.6	12.7	10.9	11.7	17
18	19.8	17.5	19.3	17.1	20.4	22.8
3	6.2	6.4	6.1	12.1	12.8	10.3
18	19.5	20.1	20.4	21.5	23.5	22.7
7	5.8	6	8.3	15.8	19	14.9
22	21.7	20.8	20.9	19	22.3	22.7
25	21.9	19.2	17.6	15.3	14.9	12
14	14.4	14.3	15.7	20.5	24.9	26.7
18	18.6	18.9	16.7	13.9	17.2	13.6
8	10.1	11.2	12.4	16.1	19.6	19.6
16	16.5	15.9	18.9	18.1	21.4	21.8
15	16.6	16.6	16.4	16.8	18.4	22.6
0	6.9	6.8	14.5	16.3	18	21.9
6	10.7	10.4	14	16.6	15.4	20.1
9	9.3	11.1	10.5	15.6	21.4	21
13	13.7	13.9	13.8	20.1	20.6	20
10	11.3	12	10.8	15.6	12	13.7
17	16.2	15.7	15.9	18.2	18.2	20.8
1	1.3	1.6	5.3	11.6	14.8	14.7
3	5.1	3.2	6.1	8.6	8.7	9.6
21	20.2	20.3	21.2	21	20.8	20.6
4	5.3	5.2	7.7	15.3	18.2	17.8
7	7.5	8.7	11.5	14.9	16.8	16.2
1	1.9	2.2	2	3.6	4.4	6.5
1	6.5	4	6	9.8	16.7	19.4
11	9.9	7.8	8.4	9.2	6.7	12.7
0	0.4	0.6	4	8.4	6.8	12.9
3	9	8.5	10.9	15.5	17.2	14
17	18.3	19.4	19.6	20.4	22.2	23.1
12	12.8	14.1	14.3	17	19.5	20.6
19	20.6	21.4	23.3	23.6	25.4	27.2

14	14	12.9	11.7	13.1	14.7	14.9
0	3.8	3.1	5.4	12.8	10.2	11.7
4	5.9	4.6	8.3	10.7	11.9	8.5
12	11.9	11.4	10.2	14	9.6	14.6
15	14.6	12.9	15.8	19.1	19.6	21.1
11	11	10.1	11.1	12.9	10.7	15.9
12	10.8	10.8	11.4	15.1	14.2	18.6
11	9.4	10.8	10	11.5	9.8	14.9
10	12.7	12.4	14.2	18	17.7	19
3	1.9	4.3	2.5	7	8.1	8.1
1	1.9	1.8	3.2	9.9	9.8	11.3
22	23	23.7	21.5	24.6	23.1	22
9	12	11.9	14.3	19.7	22.4	25.4
13	14.1	14	13.9	12.6	14.7	18.7
10	17.2	18.3	22.5	26.4	26.8	28.4
2	2	2	2.6	5.2	8.8	15.3
12	12.3	14.2	13.5	17.6	17	18.6
16	20.1	20.2	22	24.3	25.3	30.2
7	8.3	9	9.1	13.1	13.5	16.1
10	18.9	20.4	23	22.9	26.9	29.9
6	6.9	7.5	11.7	15.1	18	25.5
17	14.8	12.9	11.3	9.2	9.2	13.9
13	13.1	13.2	13.5	15.3	16.1	15.6
4	4.7	6.4	3.8	6.8	8.5	8.1
7	8.9	7.8	11.2	11.8	14.1	19.4
15	14.1	14.3	14.9	15	19	16
11	9.6	11.3	8.1	9.3	17.8	17
0	0	0	0	3.7	2	5.4
5	5.7	6.5	9.6	8.3	11.9	11.9
24	23.1	17.7	10.3	10.2	5.9	10.2
19	9.5	9.3	6.2	5.7	12.5	12.2
1	1.1	1.5	3	5.2	6	13.1

A.2 Larger illustrations

Larger illustrations of some of the figures in the thesis are shown here, in Figures A.1-A.11.

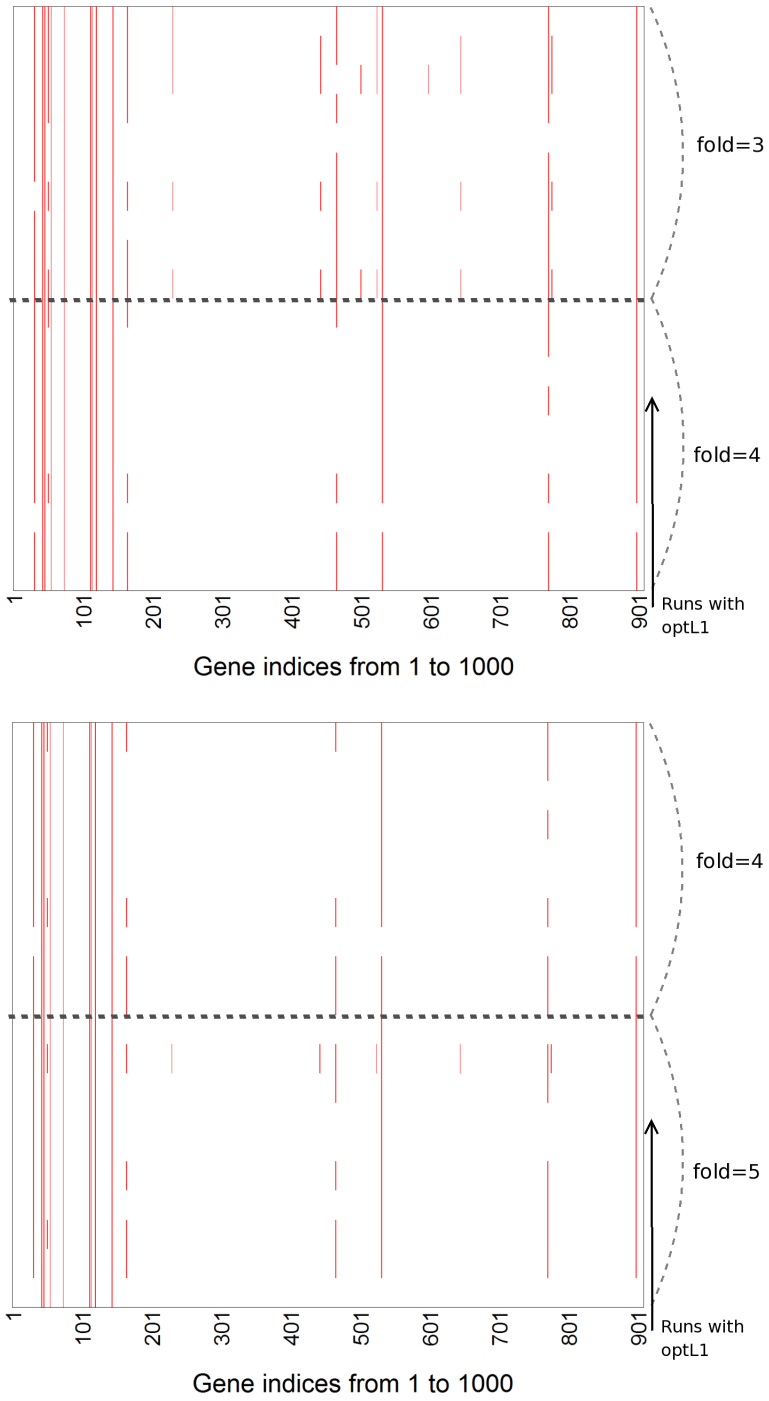

Figure A.1: One dataset, 5-, 4- and 3-fold CV. *A closer look at part of Figure 4.3.*

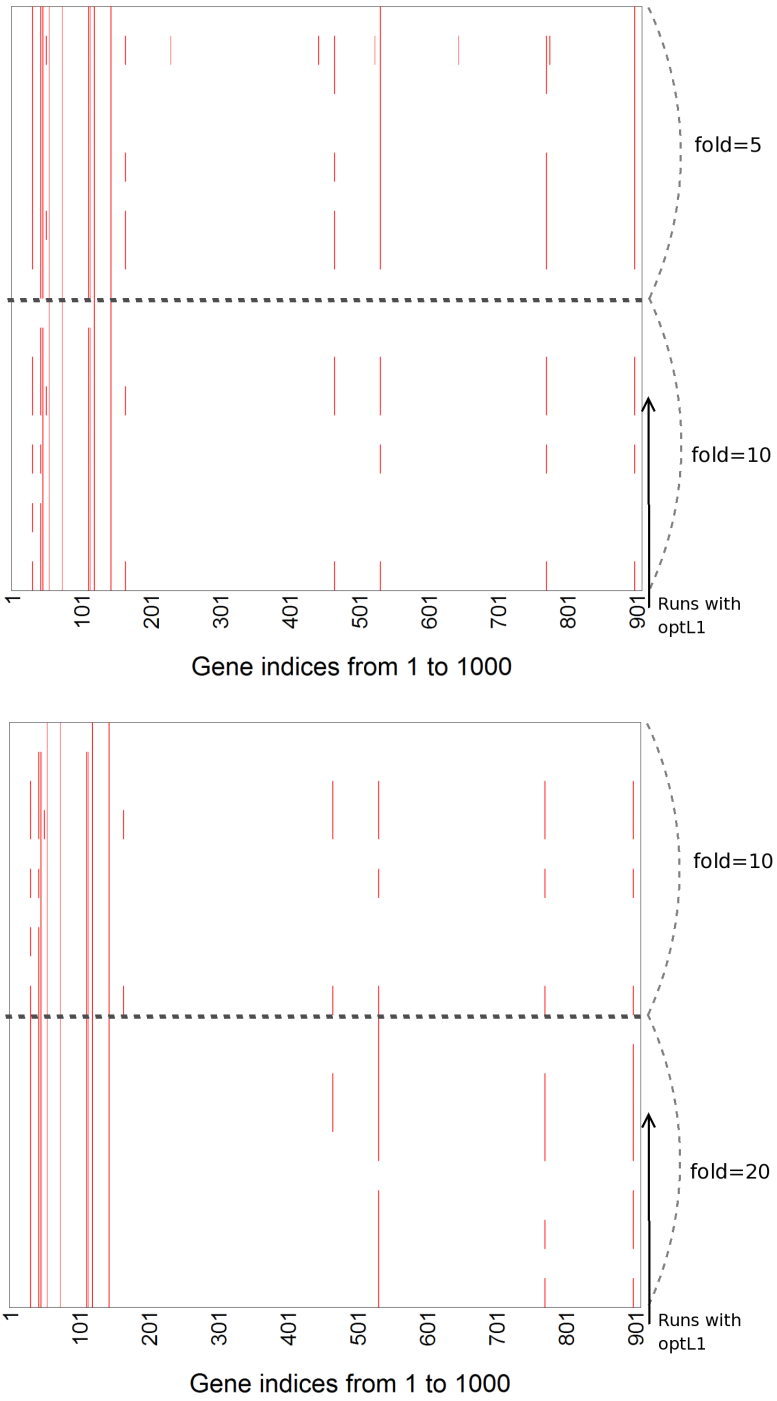

Figure A.2: One dataset, 20-, 10- and 5-fold CV. *A closer look at part of Figure 4.3.*

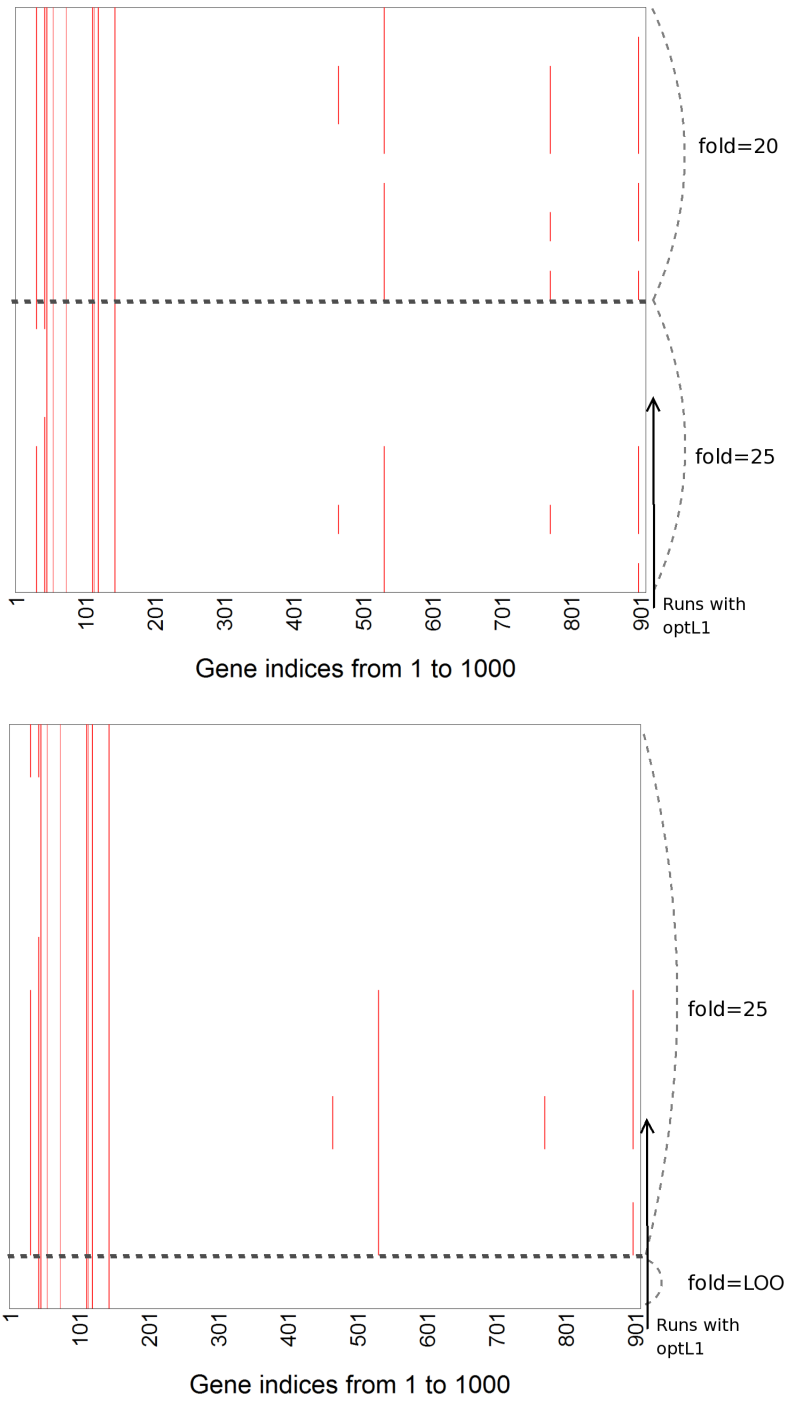

Figure A.3: One dataset, LOO, 25- and 20-fold CV. *A closer look at part of Figure 4.3.*

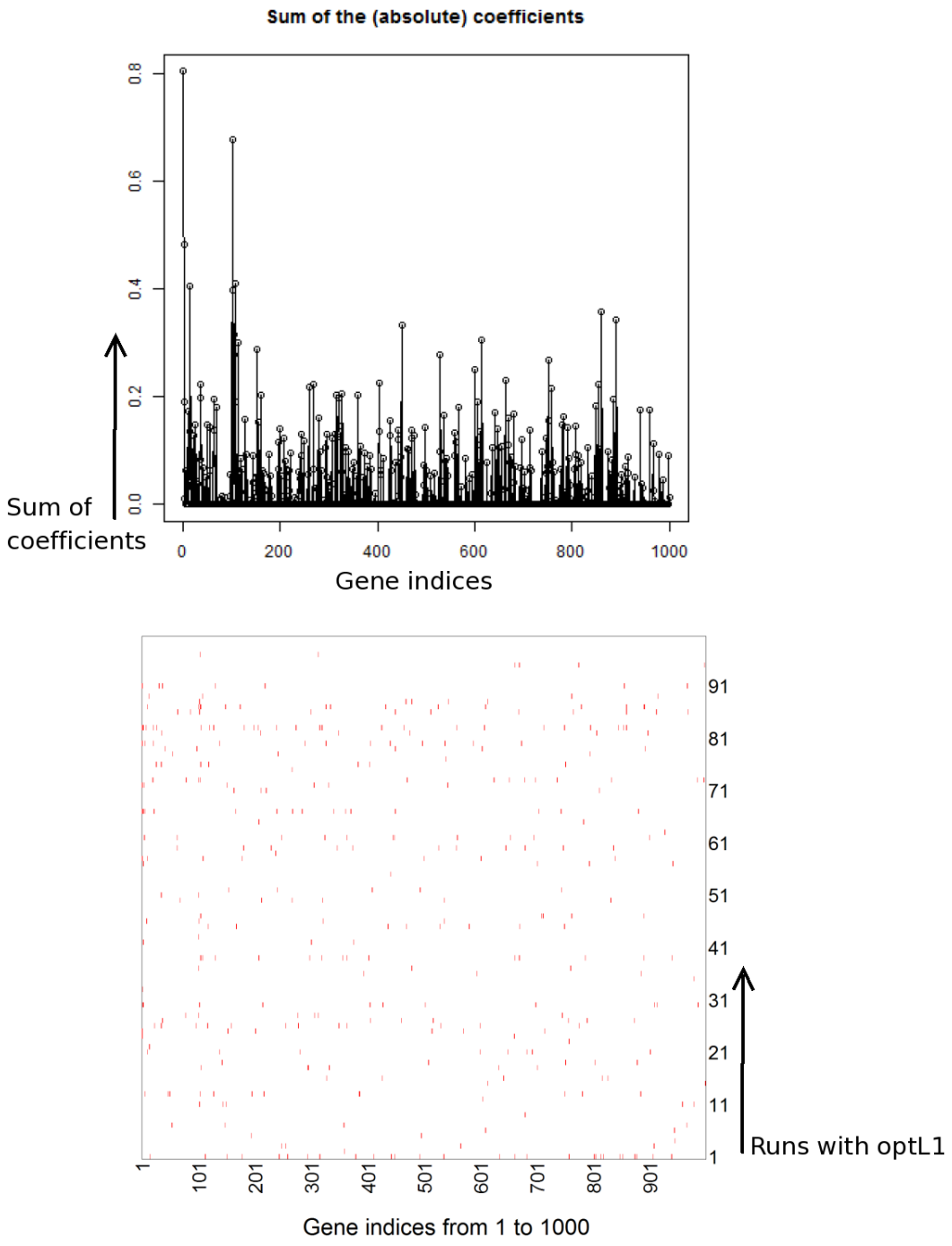

Figure A.4: 100 observations, LOOCV. *Larger version of left hand side of Figure 4.4, showing all 1000 genes.*

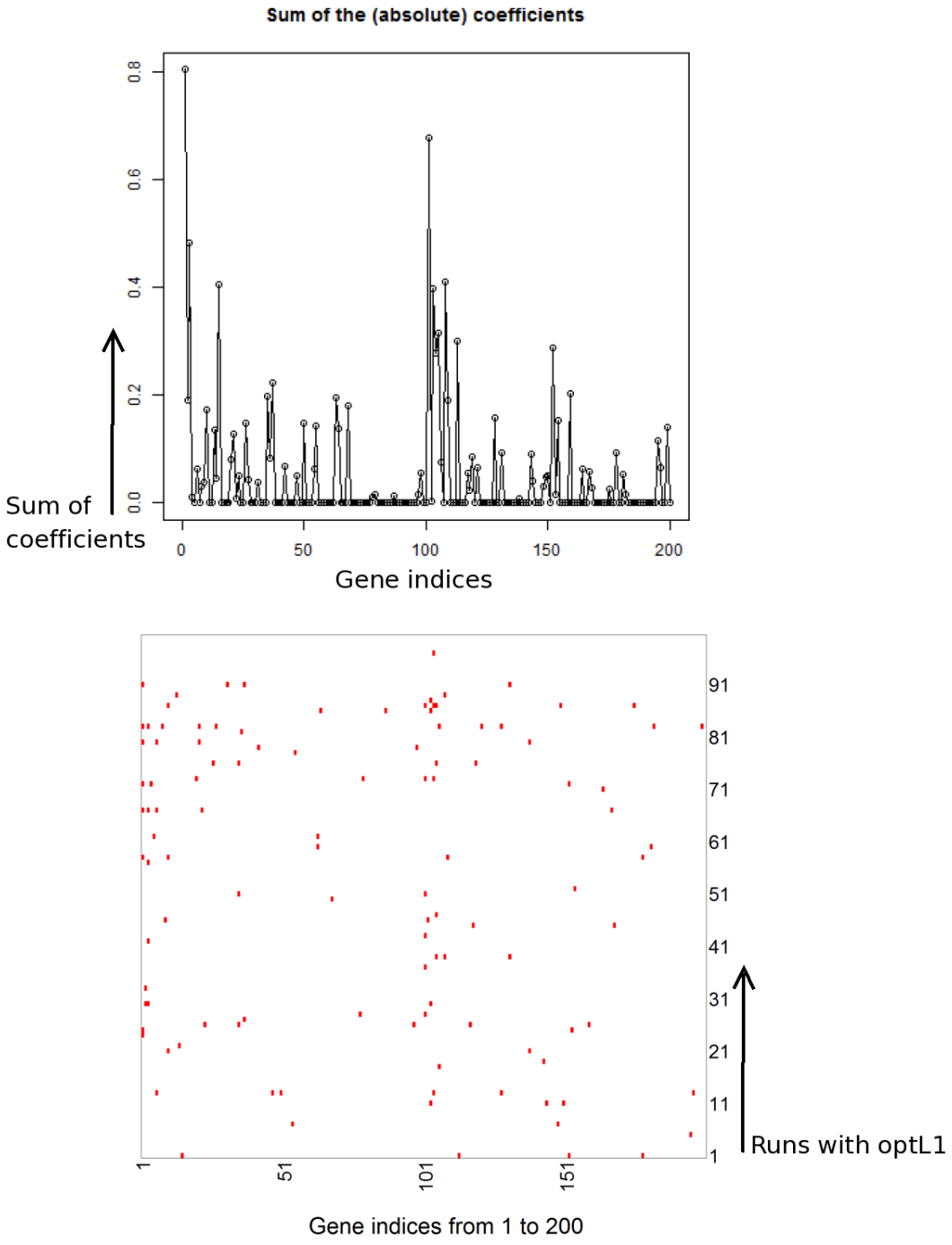

Figure A.5: 100 observations, LOOCV. *Larger version of right hand side of Figure 4.4, showing just the 200 first genes.*

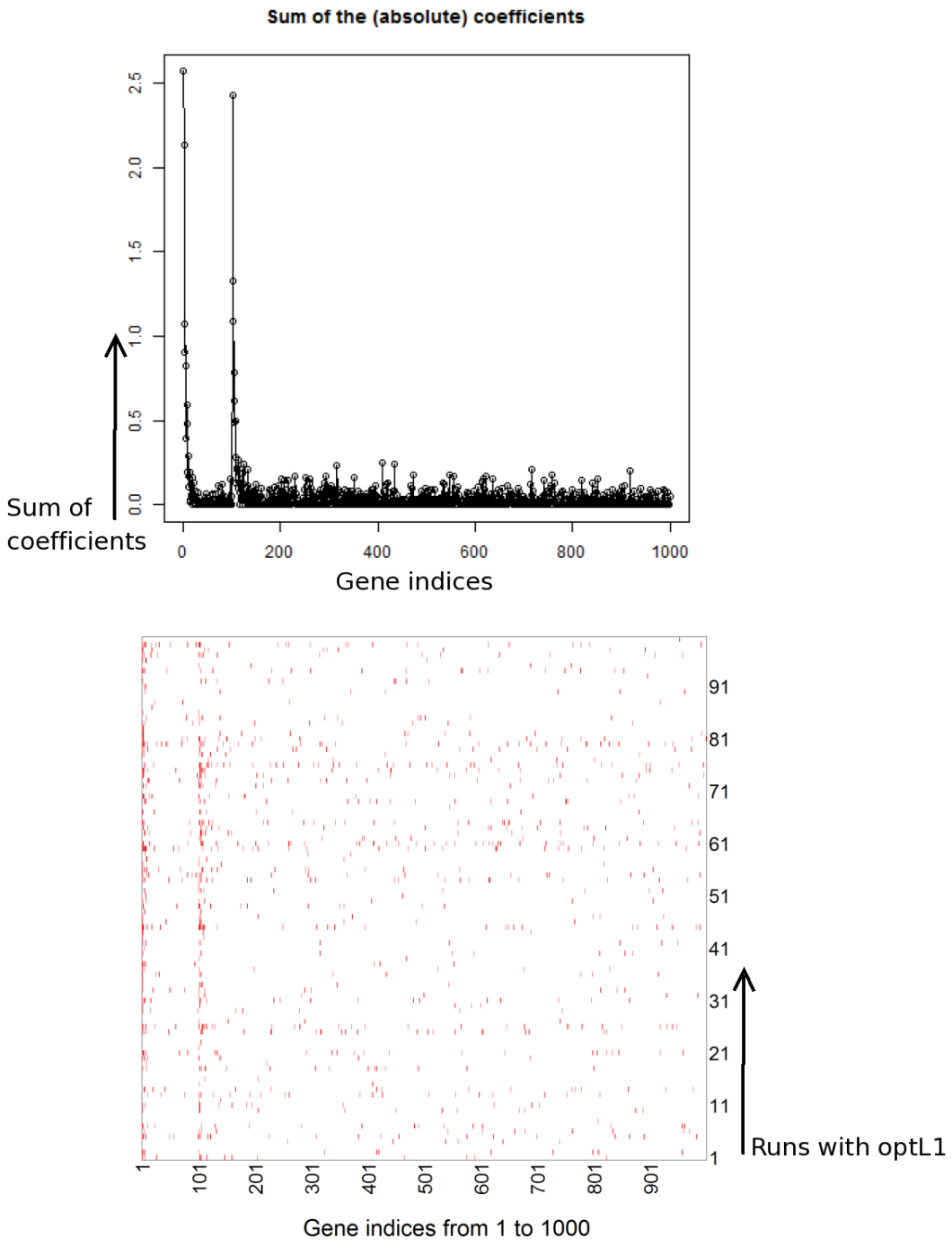

Figure A.6: 300 observations, LOOCV. *Larger version of the left hand side of Figure 4.5, showing all 1000 genes.*

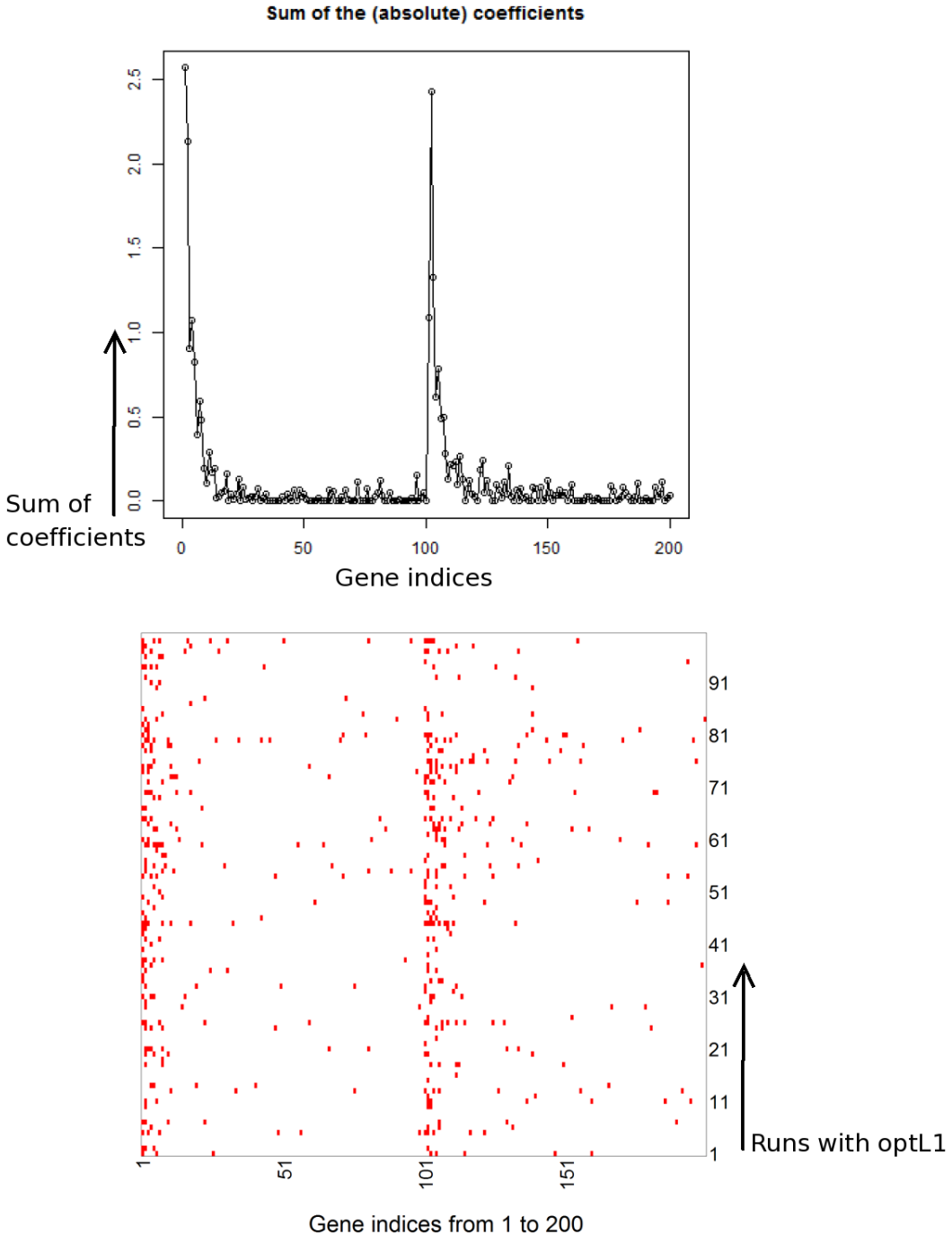

Figure A.7: 300 observations, LOOCV. *Larger version of the right hand side of Figure 4.5, showing just the 200 first genes.*

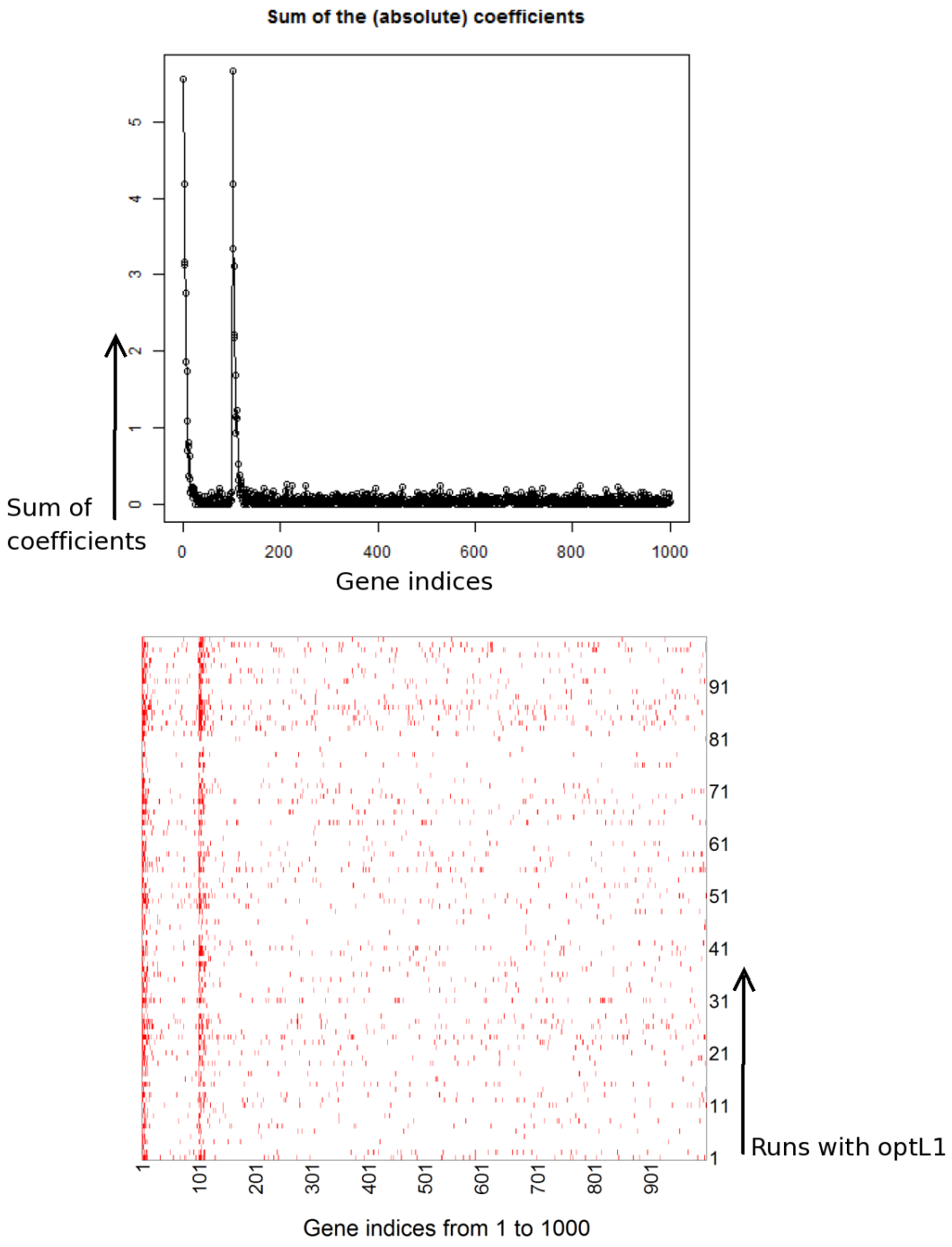

Figure A.8: 500 observations, LOOCV. *Larger version of left hand side of Figure 4.7.*

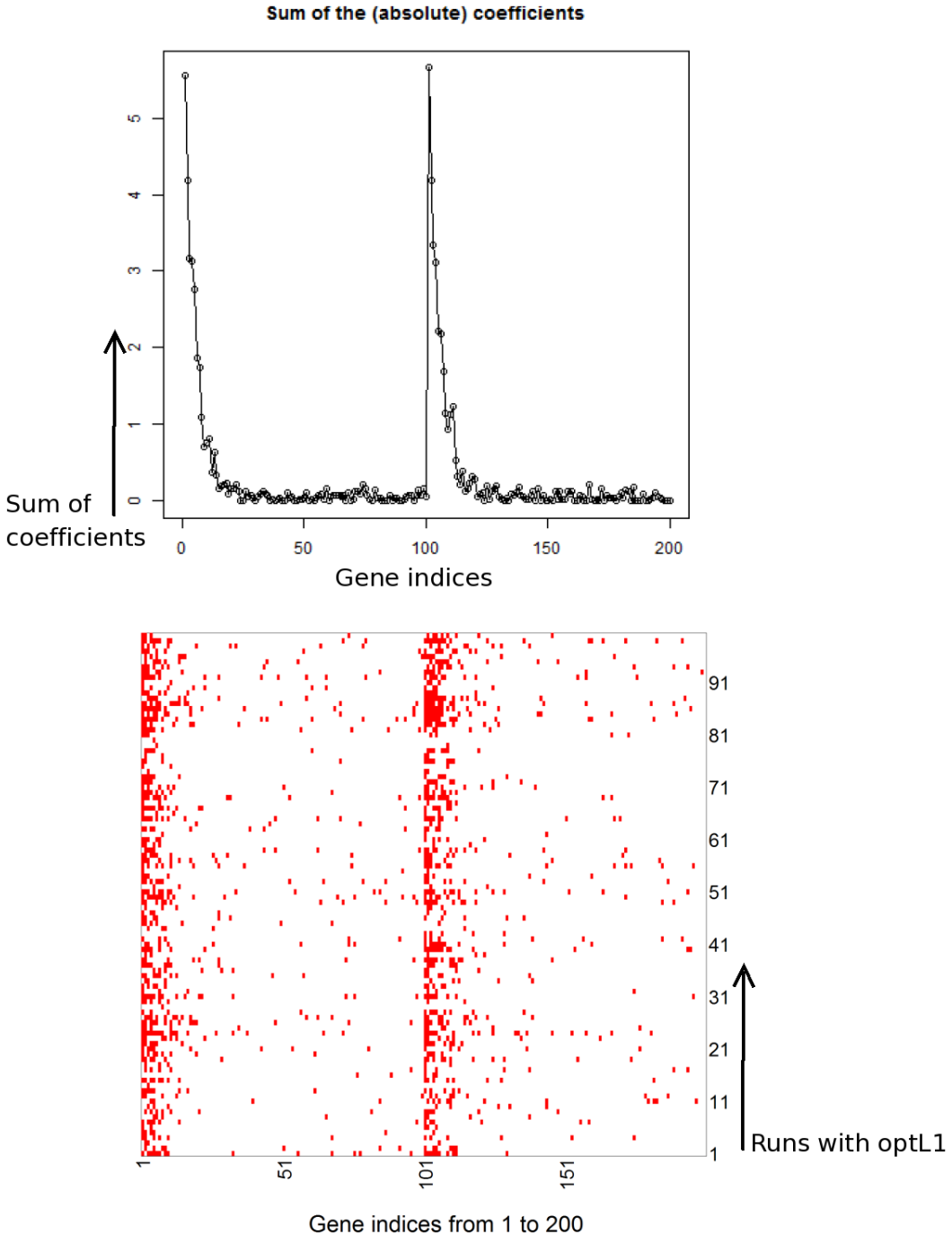

Figure A.9: 500 observations, LOOCV. *Larger version of right hand side of Figure 4.7.*

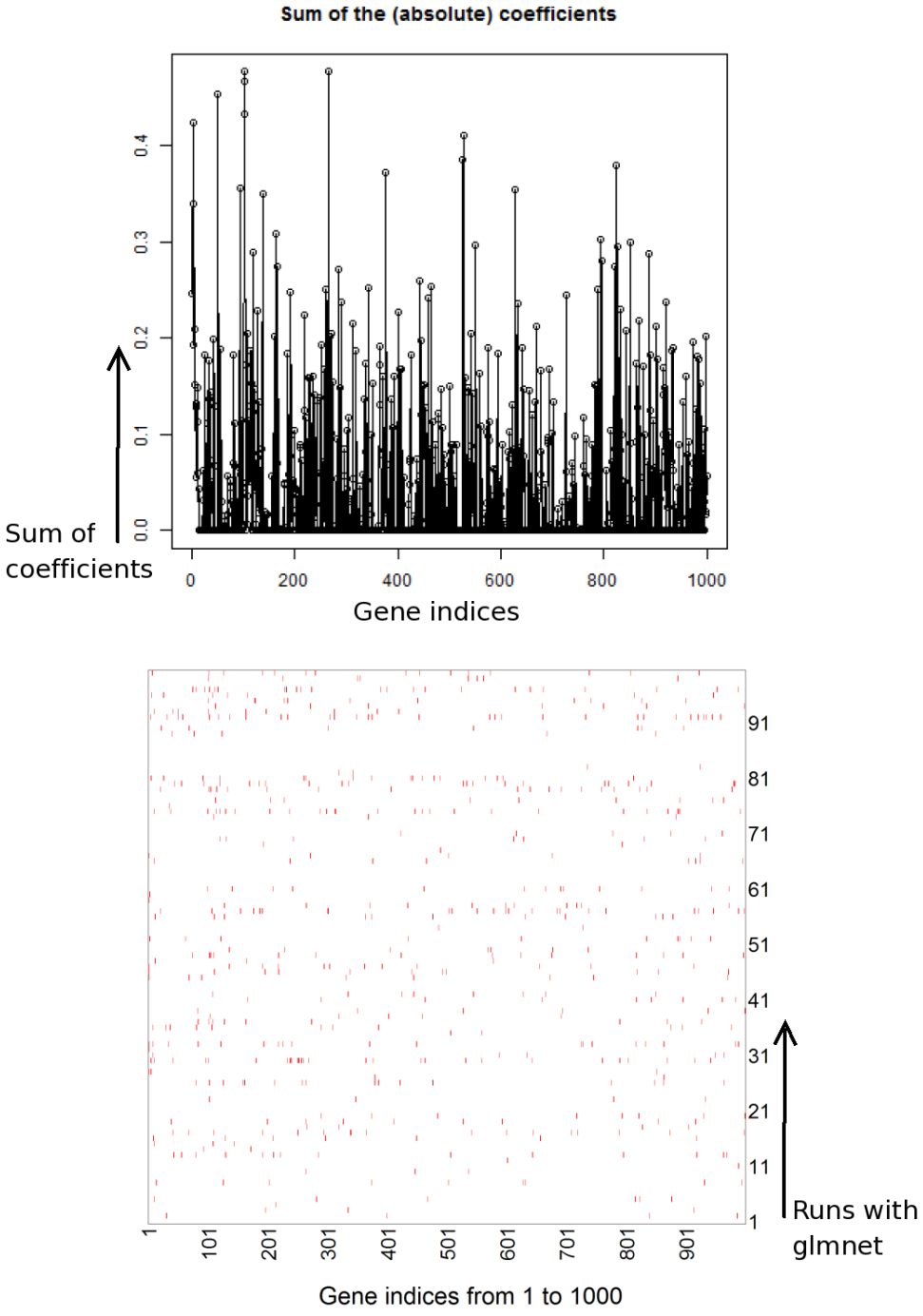

Figure A.10: Glmnet, n=100 observations. *From Figure 4.18, left hand side.*

A.3 More examples

More examples of some of the computations shown in Figures in the thesis are shown here, in Figures A.12-A.16.

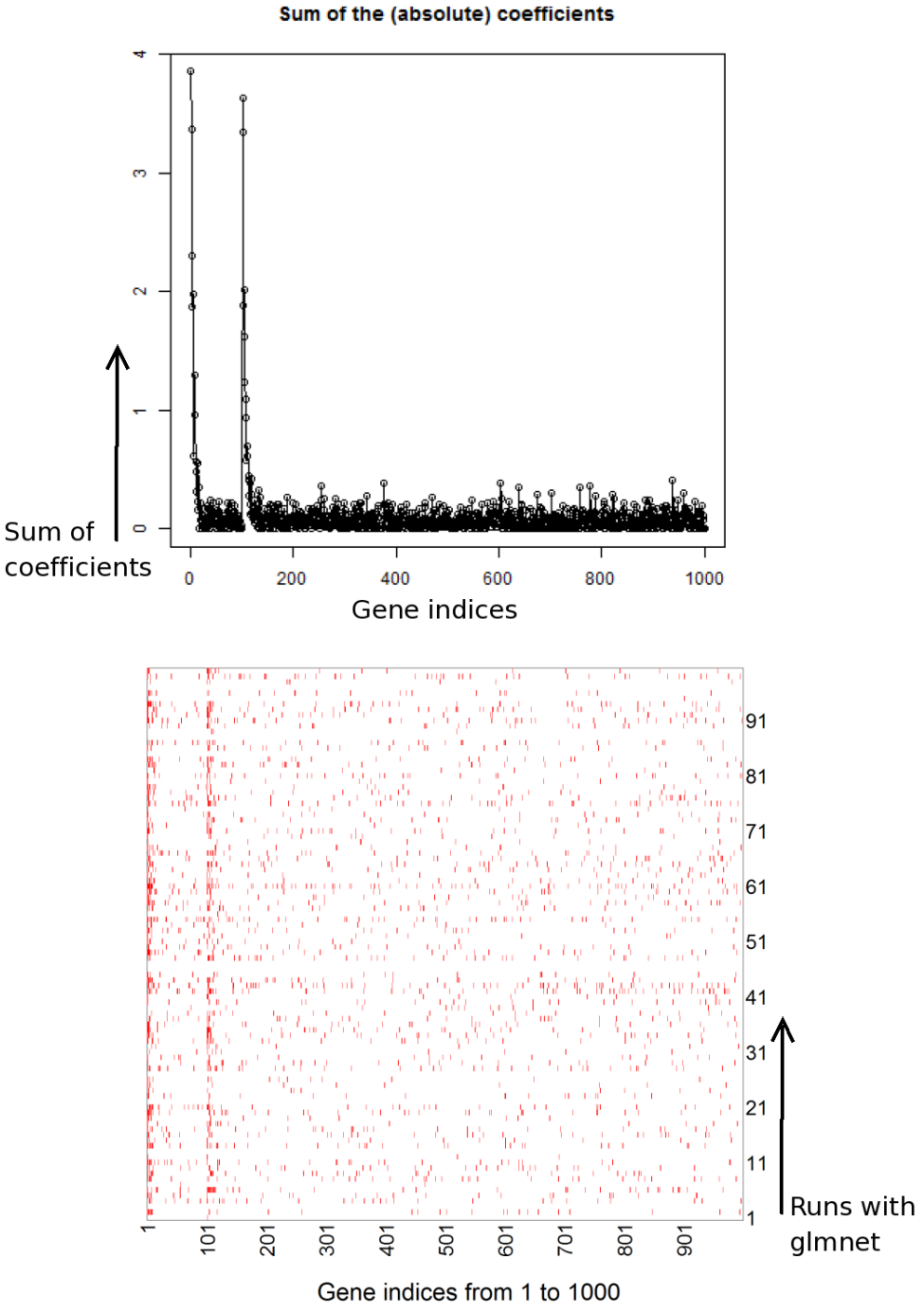

Figure A.11: Glmnet, n=300 observations. *From Figure 4.18, right hand side.*

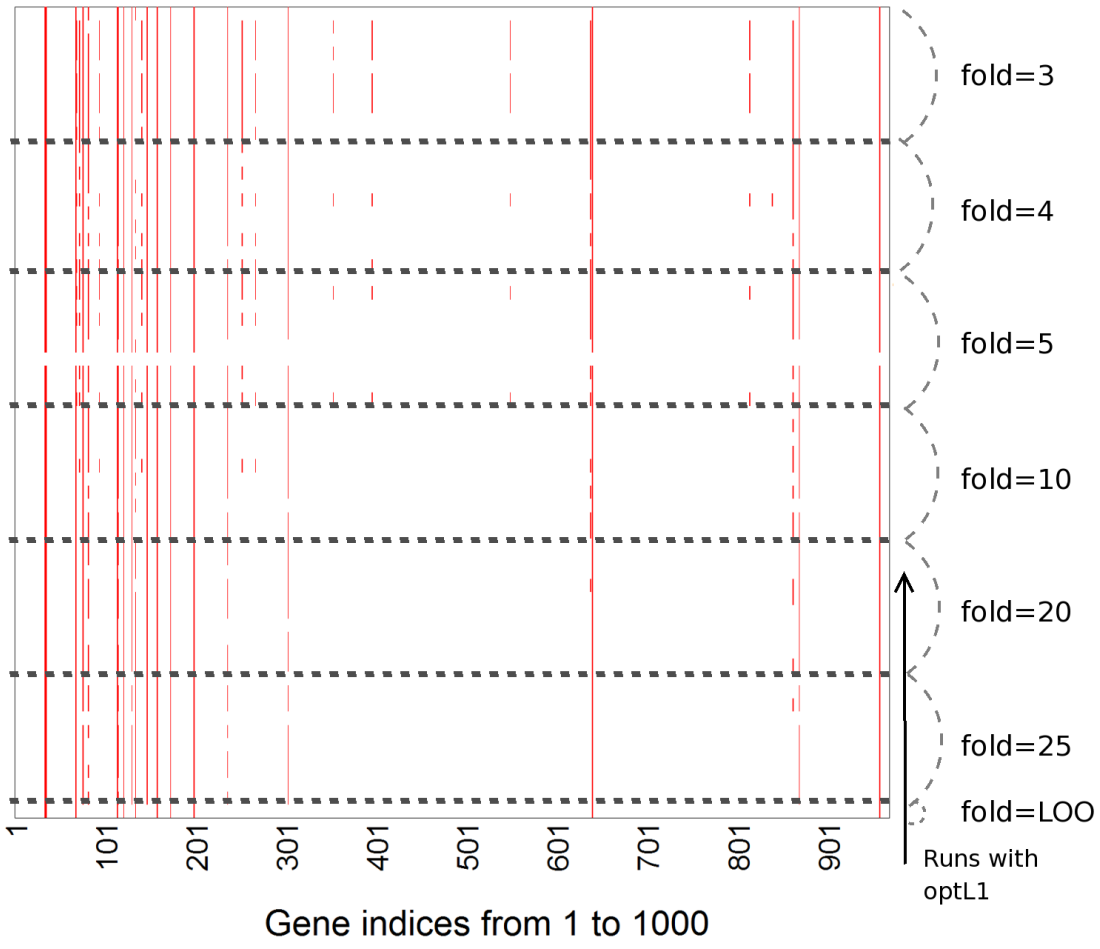

Figure A.12: One dataset, different folds. *As in Figure 4.3, but a new data set.*

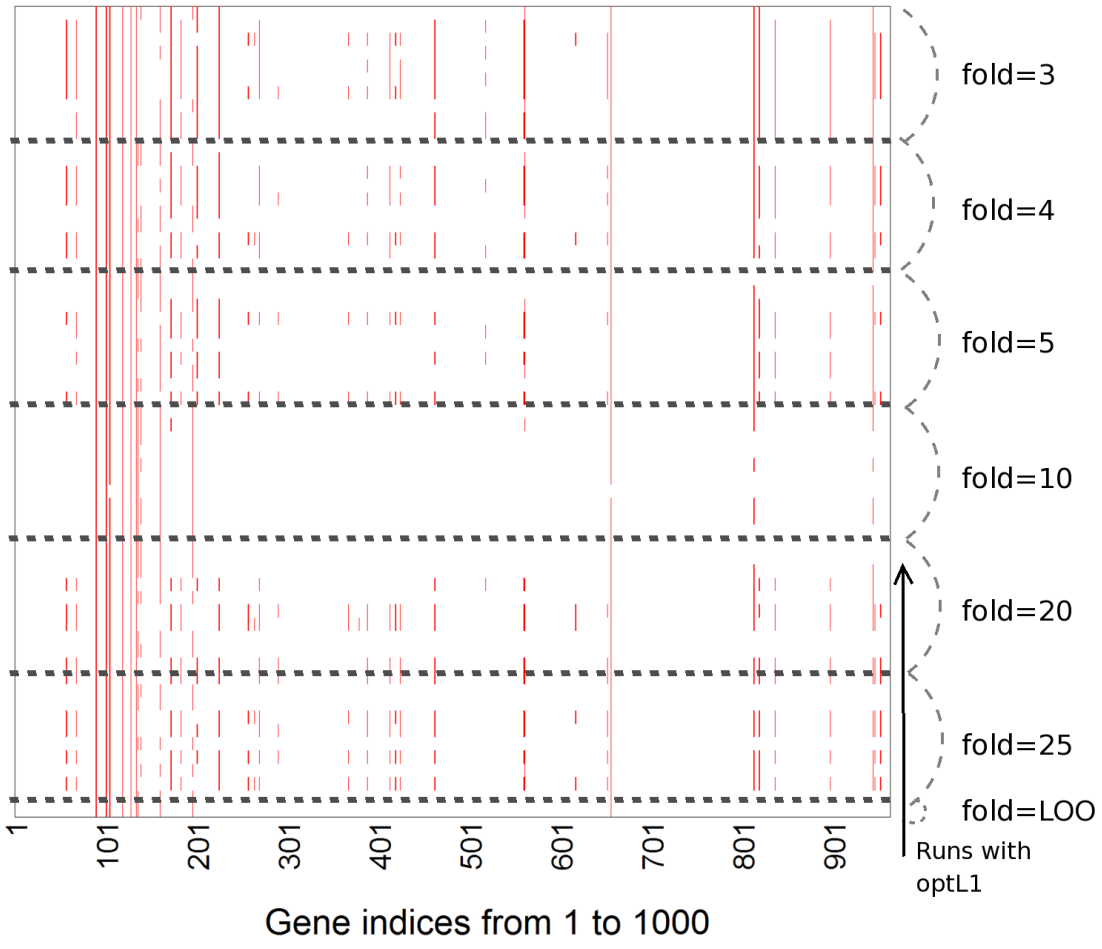

Figure A.13: One dataset, different folds. *As in Figure 4.3, but a new data set.*

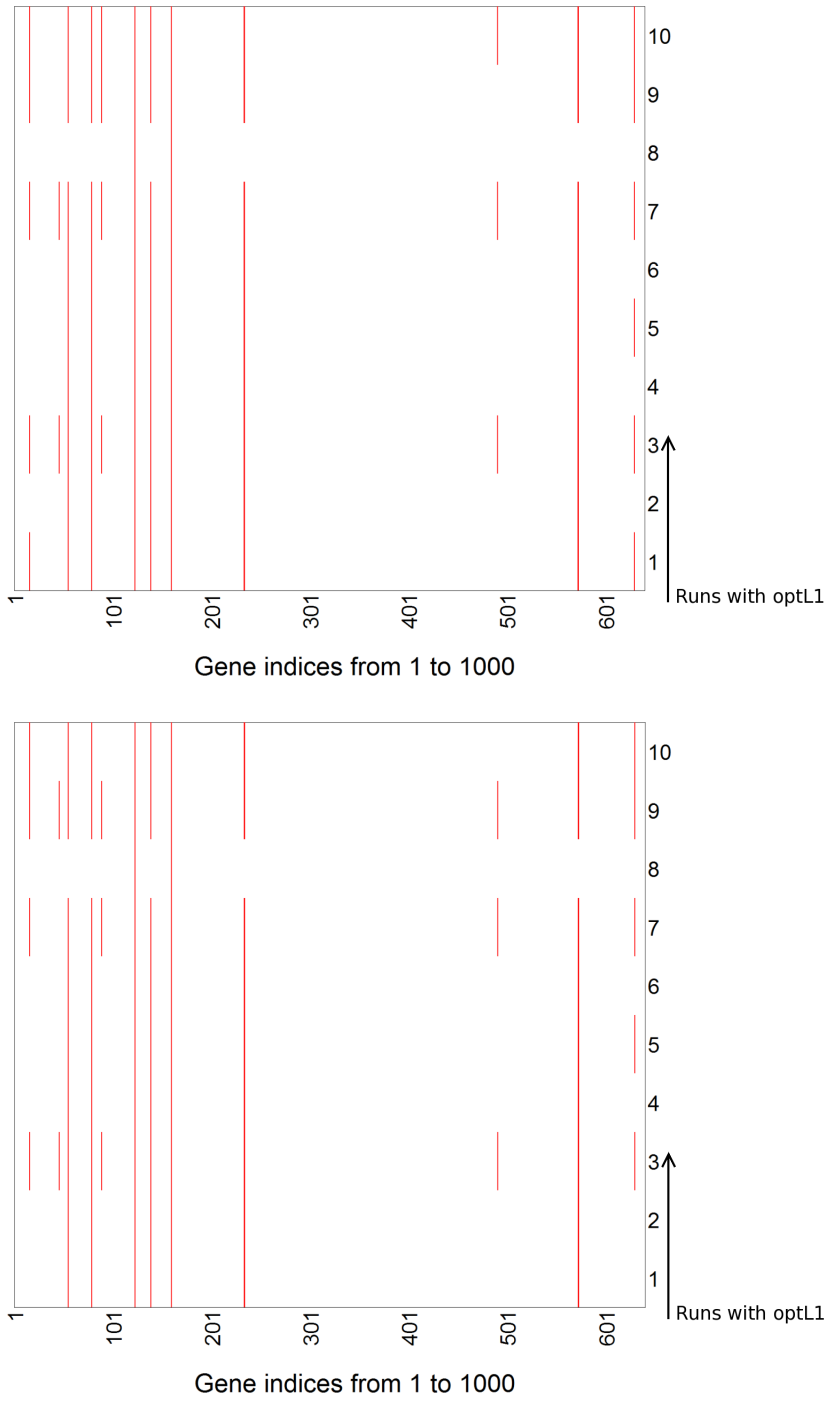

Figure A.14: Checking for global minimum error. *Like Figure 4.15, but with a new data set.*

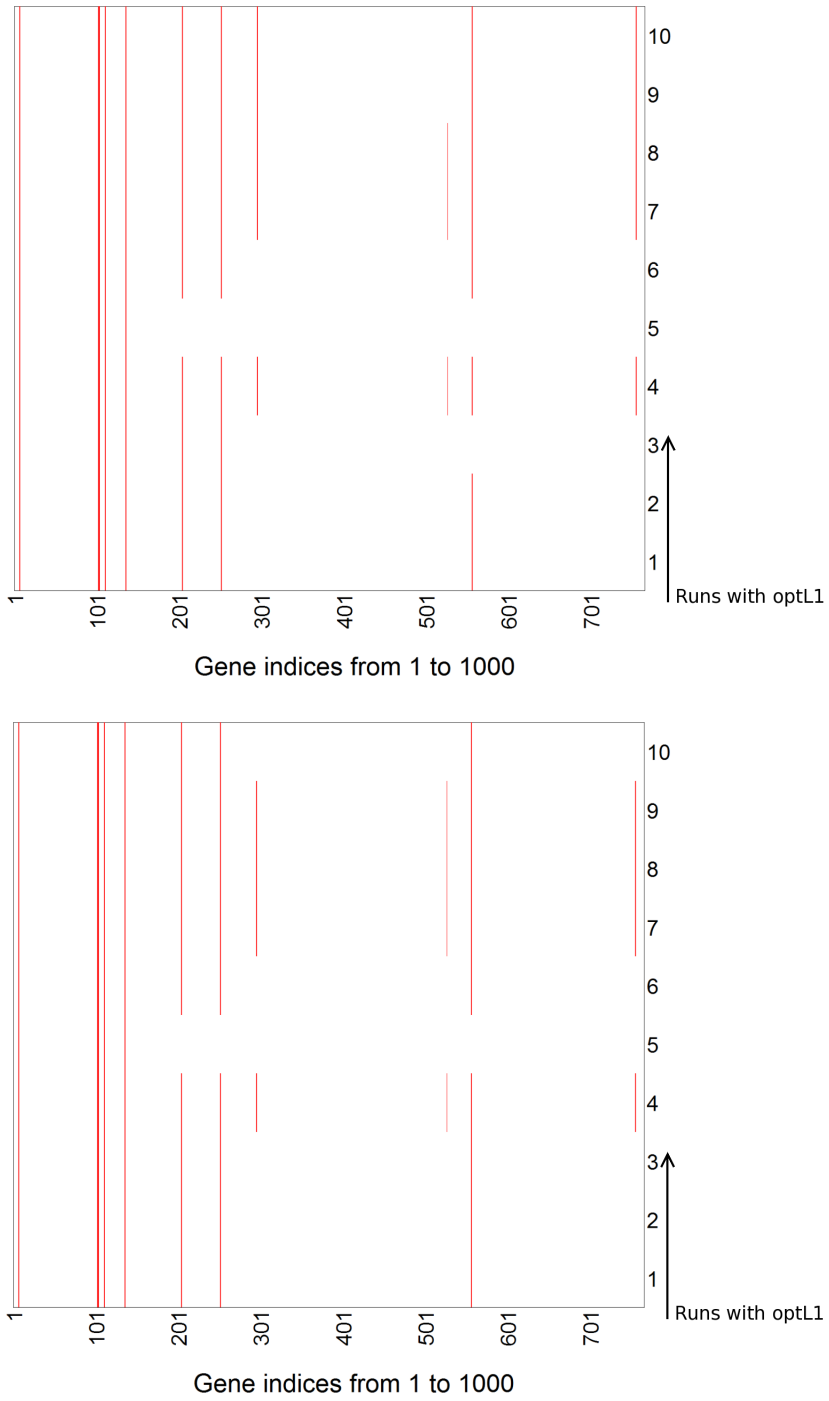

Figure A.15: Checking for global minimum error. *Like Figure 4.15, but with a new data set.*

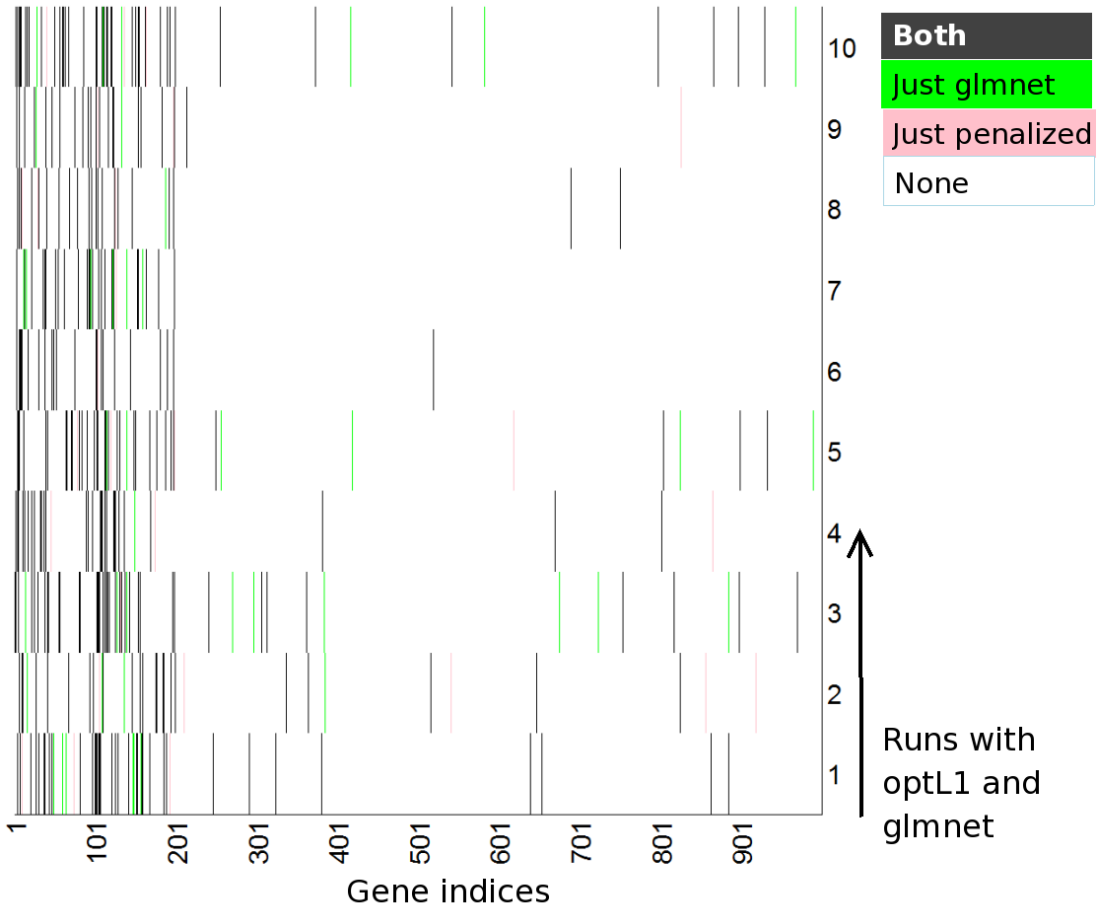

Figure A.16: Comparison of `glmnet` and `penalized`. *Zoom out of Figure 4.19.*