René Luijk

The group lasso in the proportional hazards model with an application to multiply imputed high-dimensional data

Master thesis, defended on July 26, 2012

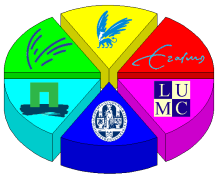
Thesis advisors:

Dr. Jelle J. Goeman

Prof. Dr. Hein Putter

Prof. Dr. Jacqueline J. Meulman

Specialisation: Statistical Science



Abstract

The increasing dimensionality of data, for example in genetics, requires additional assumptions on the regression models in order to obtain estimates of the regression coefficients, known as regularization. Lasso, one of the regularization methods available, imposes an L_1 constraint on the vector of coefficients, leading to the desirable feature of setting some coefficients to zero. This results in sparse, and more importantly, estimable models. However, setting some coefficients becomes problematic when one would want a group of variables to be in or out of the model, e.g. a factor coded as several dummy variables. This problem is solved by the group lasso. This paper presents an extension to the algorithm presented by Goeman for optimizing the penalized log likelihood under the proportional hazards model. This algorithm combines the gradient ascent algorithm and the Newton-Raphson algorithm. This methodology is then applied to data obtained from the Carema case-cohort study, which aim is to assess the additional predict 10-year risk of coronary heart disease.

Contents

1 Introduction

With the increasing dimensionality of data, for example in genetics, more ill-posed problems arise. These are problems where a unique solution to a certain problem does not exist. In order to solve these ill-posed problems, additional constraints are needed, known as regularization. One type of constraint is used in the lasso. Tibshirani (1996) proposed the lasso for the general linear model, with the purpose of building sparse prediction rules. The lasso does so by imposing an L_1 constraint on the regression coefficients $\boldsymbol{\beta}$, causing some coefficients to be set to zero, while simultaneously shrinking the other coefficients toward zero. The coefficients $\boldsymbol{\beta}$ are given by

$$\boldsymbol{\beta}^{lasso} = \arg\max_{\boldsymbol{\beta}} \{l(\boldsymbol{\beta}) - \lambda ||\boldsymbol{\beta}||_1\}$$

where $l(\boldsymbol{\beta})$ is the log likelihood and $||\cdot||_1$ the L_1 norm. An advantage of this method is that it leads to sparse models, thus enhancing the interpretability of the model. The lasso has become a popular method, especially in high-dimensional data, where the number of variables may greatly exceed the number of subjects. Sometimes this high dimensionality of the data makes a constraint on the coefficients necessary in order to obtain a solution at all. Improved prediction accuracy is also an advantage, besides enhanced interpretability. While ordinary least squares (OLS) produces estimates with no bias, the coefficients tend to have large variances. We may use the bias-variance trade-off in order to reduce the variance substantially by sacrificing some bias (see Appendix A), hence improving prediction accuracy.

When applying lasso on a group of variables the property of setting some coefficients to zero becomes problematic. An example of a group of variables may be an additive polynomial model, or a factor coded as several dummy variables. In these cases one would want to either include all variables in the model, or exclude all variables. The lasso can still be applied, but it would pose some problems. Not only does setting an individual coefficient part of a group of variables to zero hamper interpretability, in the case of a factor the results also depend on the coding of this variable. Choosing one category as baseline produces different results than when another category is chosen.

In order to tackle these problems, Yuan and Lin (2006) introduced the group lasso, which penalizes groups of variables, as opposed to single variables. Not only does this help interpretability of the model, it also leads to sparser prediction rules. The estimation of the coefficients is a little different from the regular lasso, in that the constraint is now applied to each group of variables. In regular lasso it is possible to have a different constraint for each coefficient. Just as with regular lasso, it is also possible to have different constraints for each different group of variables in the group lasso.

The above methodology has been extended to models other than the general linear model, e.g. to the logistic model (Meier et al., 2008), but not to the proportional hazards model (Cox, 1972).

In the current paper we extend the group lasso to this model and apply it to a data set obtained from a study by Vaarhorst et al. (2011). This case-cohort study focuses on the prediction of the risk of future coronary heart disease (CHD), where the response is survival time. This prediction was previously done using a risk score based on several known risk factors: age, gender, smoking status, systolic blood pressure and cholesterol level. However, since heritability is known to have an impact on predicting the risk of future CHD, Vaarhorst et al. also collected data on 105 single nucleotide polymorphisms (SNPs) known to be associated with CHD in order to increase prediction accuracy.

Due to missing data in these SNPs the data had to be imputed, which was done using multiple imputation. This procedure generated 20 differently imputed data sets. When applying lasso to each of these data sets one finds, unsurprisingly, that the results differ. Some regression coefficients are set to zero in one data set, while being nonzero in the other. This leads to the awkward situation where the model depends on which imputed data set is being used. The solution to this problem used here is considering the same SNP in each data set to be a group of variables. This results in 105 groups of each 20 variables, since we have 105 SNPs per data set, and we have 20 imputed data sets. Group lasso would then either leave a group of the same SNPs out by setting their coefficients to zero or leave the entire group in. This way we would avoid the problem caused by the regular lasso.

The rest of the paper is organized as follows. In the next section we explain the above mentioned methods in detail and combine them to develop the group lasso in the proportional hazards model. Next, in section 3, we describe the estimation of the regression coefficients and the algorithm to achieve this. The application of this methodology to the described data is in section 4. Lastly, the discussion is found in section 5.

2 Group lasso for the proportional hazards model

2.1 Proportional hazards model

Consider survival data for subject i: $(t_i, \mathbf{x}_i, \delta_i)$, where $\mathbf{x}_i$ a vector of covariates $(x_{1i}, \ldots, x_{pi})^T$ and $\delta_i = I\{y_i \leq c_i\}$ the status indicator, $i = 1, \ldots, N$. Further, let $t_i = \min(y_i, c_i)$, where y_i is the survival time, and c_i the censoring time. The proportional hazards model estimates the hazard $\lambda(t; \mathbf{x}_i)$ for subject i at time t conditional on the vector of covariates $\mathbf{x}_i$ by

$$\lambda(t; \mathbf{x}_i) = \lambda_0(t) \exp\{\mathbf{x}_i \boldsymbol{\beta}\} \tag{2.1}$$

where $\lambda_0(t)$ is the baseline hazard. The vector $\boldsymbol{\beta}$ is obtained by maximizing the partial likelihood

$$\prod_{i=1}^{N} \left(\frac{\exp\{\mathbf{x}_i \boldsymbol{\beta}\}}{\sum_{j=1}^{N} y_j(t_i) \exp\{\mathbf{x}_j \boldsymbol{\beta}\}} \right)^{\delta_i} \tag{2.2}$$

e.g. by means of a Newton type algorithm. Here $y_j(t_i)$ indicates the status indicator for person j at time t_i. Note that the above model can be easily adjusted to allow for time-dependent covariates. The model given in (2.1) can also be extended to handle stratification, giving a slightly different model. In the case of stratification of our data on a variable Z having C levels, the hazard for subject i from stratum c at time t is given by

$$\lambda(t|\mathbf{x}_i, Z_c) = \lambda_{0c}(t) \exp\{\mathbf{x}_i\boldsymbol{\beta}\}, \quad c = 1, \ldots, C \tag{2.3}$$

This may be feasible when one wants to control for a predictor that violates the proportional hazards assumption. In this case each stratum has a different baseline hazard. We will later see that this is not desirable for our current problem.

2.2 Case-cohort design

The case-cohort design is a mixture of a cohort design and a case-control design. A cohort design follows a randomly chosen group of subjects, the cohort, who have not experienced the event of interest (e.g. death) over time to establish the risk factors in the development of this particular event. A case-control design matches randomly chosen subjects who already have experienced the event with comparable subjects who have not. In the case-cohort design a random sample of the cohort, the so-called subcohort, is followed over time and some features measured, including the event of interest. Note that this random sample is chosen independently of any information known about the subjects, i.e. the subcohort is chosen completely at random. The covariates of the subjects outside the subcohort are not measured unless they experience the event. The advantage of this design is that it reduces the costs of measuring the features of interest. For example, genotyping a large number of controls may be very expensive while not substantially increasing the precision of the estimates. A disadvantage of the case-cohort design is that the subcohort risk set may be small in cases where the subjects are at risk for only a short period of time, leading to small risk sets at some failure times (Prentice, 1986). In such cases the efficiency of the case-cohort design is not good, relative to the case-control design.

For the case-cohort design the partial likelihood in (2.2) cannot be used (Prentice, 1986). Hence, an alternative is used: the pseudo-likelihood. Let S be the set of subjects in the subcohort, and $w(t_i)$ the weight given to a subject at time t_i, then the pseudo-likelihood is given by

$$\prod_{i=1}^{N} \left(\frac{\exp\{\mathbf{x}_i\boldsymbol{\beta}\}}{w_i(t_i)\exp\{\mathbf{x}_i\boldsymbol{\beta}\} + \sum_{\substack{k\in S \\ k\neq i}} y_k(t_i) w_k(t_i) \exp\{\mathbf{x}_k\boldsymbol{\beta}\}} \right)^{\delta_i} \tag{2.4}$$

The weight given to each subject depends on what weighting scheme is used. Several popular weighting schemes exist. However, since the results are almost identical, we will only use the scheme by

Prentice (1986). Other weighting schemes are given by Self and Prentice (1988) and Barlow (1994). Prentice gives weight $w(t_i) = 0$ if the subject is going to have an event in the future but is outside the subcohort and $w(t_i) = 1$ otherwise. Self and Prentice also give weight $w(t_i) = 0$ to cases outside the subcohort at time of failure and $w(t_i) = 1$ otherwise. The three schemes are given in Table 1.

Outcome type and timing	**Prentice**	**Self & Prentice**	**Barlow**
Case outside subcohort before failure	0	0	0
Case outside subcohort at failure	1	0	1
Case in subcohort before failure	1	1	$1/\alpha$
Case in subcohort at failure	1	1	1
Subcohort control	1	1	$1/\alpha$

Table 1: Weighting schemes by Prentice and Self & Prentice

2.3 Lasso

The lasso (Tibshirani, 1996, 1997) was introduced as a way to select a subset of variables in a model while simultaneously shrinking the other regression coefficients toward zero. This is done is by imposing an L_1 constraint on the regression coefficients $\boldsymbol{\beta}$. Note that this is the same as specifying a prior distribution on the parameters, as is done in Bayesian statistics (see Appendix B). There are two alternative ways of defining this constraint and obtaining a solution. Tibshirani (1996) proposed to optimize the log likelihood $l(\boldsymbol{\beta})$ subject to a constraint on the regression coefficients, obtaining the estimates $\boldsymbol{\beta}^{lasso}$.

$$\boldsymbol{\beta} = \arg\max_{\boldsymbol{\beta}} \{l(\boldsymbol{\beta})\} \quad \text{subject to} \sum_{k=1}^{p} |\beta_k| \leq z \tag{2.5}$$

Here, z is some tuning parameter and $l(\boldsymbol{\beta})$ is the pseudo-likelihood given in (2.4). Alternatively, we can write this in the Lagrangian form so that the solution $\boldsymbol{\beta}$ is obtained by optimizing the penalized log likelihood $l_{pen}(\boldsymbol{\beta})$,

$$\begin{aligned} \boldsymbol{\beta} &= \arg\max_{\boldsymbol{\beta}} \{l_{pen}(\boldsymbol{\beta})\} \\ &= \arg\max_{\boldsymbol{\beta}} \left\{ l(\boldsymbol{\beta}) - \lambda \sum_{k=1}^{p} |\beta_k| \right\} \end{aligned} \tag{2.6}$$

where we have λ as a tuning parameter. Though each coefficient may be penalized differently by having different values for each β_k, often one value for all different β_k is used. Optimizing (2.6) does not mean that all predictors are included in the model. In particular, optimizing (2.6) means that some parameters may be set to zero if that variable is assumed to be not predictive of the response.

The motivation for this penalization is two-fold. Firstly, selecting a few variables to be included in the model enhances interpretability. Secondly, in some cases the number of predictors may be large, even larger than the number of observations, causing traditional methods to fail, e.g. ordinary least squares. Especially in research involving for example SNPs, the number of predictors may be very large, much larger than the number of observations. The lasso introduces some bias in the estimation of the parameters, thereby reducing the variance, which in turn leads to better prediction accuracy.

Another regularization method, often used as an alternative to lasso, is ridge regression, introduced by Hoerl and Kennard (1970). Ridge is similar to lasso in that it constrains the regression coefficients, forcing them toward zero, but it does so in a slightly different way. The estimates of the regression coefficients in ridge are found by maximizing

$$\arg\max_{\boldsymbol{\beta}} \left\{ l(\boldsymbol{\beta}) - \lambda \sum_{k=1}^{p} \beta_k^2 \right\} \tag{2.7}$$

Note that the penalty is different from the penalty used in (2.6). Also, as opposed to lasso, ridge has the following closed-form solution for the general linear model, which is very similar to the regular least-squares solution.

$$\boldsymbol{\beta}^{ridge} = (\mathbf{X}^T\mathbf{X} + \boldsymbol{\Lambda})^{-1}\mathbf{X}^T\mathbf{y} \tag{2.8}$$

Here $\boldsymbol{\Lambda}$ is a matrix with the penalty parameter λ on its diagonal and zeros elsewhere. Again, note that we may use different values for λ, corresponding to the different penalties for the coefficients β_k.

Having a closed-form solution, ridge has the advantage of not requiring computationally intensive algorithms to find the solution to (2.7), though this is true only for the general linear model. For nonlinear models there is no closed-form solution. Also, ridge generally does better than lasso in terms of prediction accuracy. However, ridge regression does have one important disadvantage. Ridge does shrink coefficients toward zero, but never sets them exactly to zero, hence leaving all variables in the model. For high-dimensional data where the goal is to build a sparse and easily interpretable model with as few variables as possible, this property is not desirable. Note that for both regularization methods no standard errors can be computed. See Appendix C for more on standard errors in regularization.

2.4 Group lasso

In some cases one may want to include or exclude a group of variables, instead of single variables. As mentioned in the introduction, the current problem deals with multiple imputation, where regular lasso yields different results for the differently imputed data sets. Here a group of variables is defined as a SNP in the different imputed data sets. The group lasso, first introduced by Yuan and Lin (2006),

generalizes the lasso to grouped variables and solves the aforementioned problem, while maintaining the same prediction accuracy as the regular lasso.

Denote by $\boldsymbol{\beta}_g$ the vector of regression coefficients belonging to group $g = 1, \ldots, G$. We then optimize the penalized log likelihood to obtain the estimates

$$\boldsymbol{\beta}^{glasso} = \arg\max_{\boldsymbol{\beta}} \left\{ l(\boldsymbol{\beta}) - \sum_{g=1}^{G} \lambda_g \left|\left|\boldsymbol{\beta}_g\right|\right|_2 \right\} \tag{2.9}$$

where $||\cdot||_2$ is the L_2 norm. Note that that (2.9) reduces to (2.6) when all λ_g are equal and all group sizes are equal to one. Also note that instead of having different values of λ_g for each group g, one may also use the notation of both Meier et al. (2008) and Yuan and Lin (2006), who write the penalty term in (2.9) as $\lambda \sum_{g=1}^{G} \sqrt{p_g} \left|\left|\boldsymbol{\beta}_g\right|\right|_2$. The term $\sqrt{p_g}$ takes into account the different group sizes. We choose to have differently scaled values for each λ_g where we define each λ_g as $\lambda\sqrt{p_g}$.

This way the group lasso model includes only groups of variables, leading to a better interpretable model. In our example the group lasso helps us avoiding the problem of obtaining different models, depending on which imputed data set we use.

2.5 The optimal value for the tuning parameter λ

The tuning parameter λ needs to take some value. We would like it to take an optimal value, in whatever way the optimum is defined. There is no easy or universally best way to find the optimal value for λ, or for any tuning parameter; various methods exist to find the optimal value. Also, what is considered to be the best depends on several factors. In general, what value is considered to be optimal is based on optimizing some function, typically a loss function $\sum_{i=1}^{n} L(y_i, \hat{f}(\mathbf{x}_i))$, where $\hat{f}(\mathbf{x})$ is a prediction model fitted, or trained, on a training set. Different choices for the loss function exist, depending on the type of data at hand. For example, for survival outcomes the Brier score (Graf et al., 1999) is an often used measure to evaluate a model. Finding the value for λ that optimizes the metric of choice can be done through several methods, of which K-fold cross-validation (CV) is the most common. In K-fold CV we randomly split the data into K so-called folds. For every fold $k = 1, \ldots, K$, we train the data (i.e. we fit a model) on all available data, leaving out the data in that particular fold (see Figure 1, where we choose $K = 5$ and the current fold $k = 3$). We call this the training set. With that model, we try to predict the data in the left out fold, the test set. For each fold we obtain an estimate of some metric to evaluate our model, i.e. for some loss function. As a final estimate of how our model performs, we take the average over all folds.

An important problem of K-fold CV is the computational burden. Fitting a penalized proportional hazards model is computationally intensive, especially if the model has to be fit 5 times for each value of λ we want to evaluate. In this paper we will use the cross-validated partial likelihood (CVPL) as

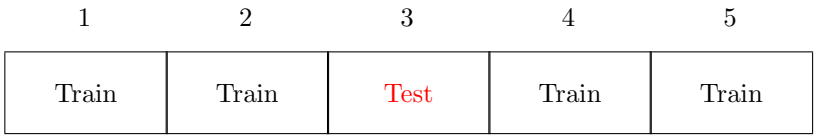


Figure 1: Observed data randomly divided into K folds

developed by Bøvelstad et al. (2007), which is an adaptation of the cross-validated criterion developed by Verweij and Van Houwelingen (1993). If we consider the CVPL to be a function of λ, we may find its root by some optimization procedure, instead of fitting the model for a grid of values. This may decrease computation time considerably. The function we would like to optimize can be differentiated, but calculating the gradient is very difficult. As a consequence, we are forced to resort to a method that does not require this gradient. With this in mind we choose to use Brent's method to find this root, i.e. the value for λ for which the CVPL is maximized. We do not give a description of this method here. Instead, the reader is referred to Brent (1973) for a description. Brent's method is faster than fitting the model for a grid of values for λ, but Brent's method may converge to a local maximum instead of the global maximum. Hence it is advised to also search a grid around the optimal value found by Brent's method to ensure the found optimum is a global maximum, not a local maximum.

As mentioned above, we will use the CVPL (Bøvelstad et al., 2007). As before, the penalized log likelihood is denoted by $l_{pen}(\boldsymbol{\beta})$. The penalized log likelihood where the kth fold is left out is denoted by $l_{pen}^{(-k)}(\boldsymbol{\beta})$. Further, let $\hat{\boldsymbol{\beta}}^{(-k)}(\lambda)$ be the estimate of the parameter vector $\boldsymbol{\beta}(\lambda)$ where the kth fold is left out of our data, and λ is the tuning parameter. We then define the cross-validated log likelihood $CV(\lambda)$ as

$$CV(\lambda) = \sum_{k=1}^{K} \left\{ l_{pen}(\hat{\boldsymbol{\beta}}^{(-k)}(\lambda)) - l_{pen}^{(-k)}(\hat{\boldsymbol{\beta}}^{(-k)}(\lambda)) \right\} \tag{2.10}$$

So we sum over the differences between the log likelihood using all data and the log likelihood using the data where the kth fold is left out. For both terms we calculate the log likelihood using the regression coefficients $\hat{\boldsymbol{\beta}}^{(-k)}$. We then find the optimal value for λ by optimizing $CV(\lambda)$.

3 Computation

In this section we describe two optimization methods for estimating the regression coefficients $\boldsymbol{\beta}^{glasso}$ by optimizing the target function described below. The first method is an extension of the algorithm as developed by Goeman (2010) for estimating L_1 penalized regression coefficients. While this method

is not yet fully implemented we will use a limited-memory quasi-Newton method to analyze the data in section 4. This method will also be described in the current section.

3.1 The target function

As stated in section 2.4, we would like to obtain estimates $\boldsymbol{\beta}^{glasso}$, as defined in (2.9), by optimizing the penalized log partial likelihood. Let us denote this likelihood by $l_{pen}(\boldsymbol{\beta})$.

$$\begin{aligned} l_{pen}(\boldsymbol{\beta}) &= l(\boldsymbol{\beta}) - \sum_{g=1}^{G} \lambda_g \left|\left|\boldsymbol{\beta}_g\right|\right|_2 \\ &= l(\boldsymbol{\beta}) - P(\boldsymbol{\beta}) \end{aligned} \tag{3.1}$$

Our target function $l_{pen}(\boldsymbol{\beta})$ consists of two parts: the log partial likelihood $l(\boldsymbol{\beta})$ and the penalty function $P(\boldsymbol{\beta})$. The log partial likelihood is a concave function that is continuous and twice differentiable at each point and is the log of (2.2),

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{N} \left[\delta_i \left(\mathbf{x}_i\boldsymbol{\beta} - \log \left(\sum_{j=1}^{N} y_j(t_i) \exp(\mathbf{x}_j\boldsymbol{\beta}) \right) \right) \right] \tag{3.2}$$

The first and second order partial derivatives of $l(\boldsymbol{\beta})$ are, respectively

$$l'(\boldsymbol{\beta}) = \sum_{i=1}^{N} \left[\delta_i \left(\mathbf{x}_i - \frac{\sum_{j=1}^{N} y_j(t_i)\mathbf{x}_j \exp(\mathbf{x}_j\boldsymbol{\beta})}{\sum_{j=1}^{N} y_j(t_i) \exp(\mathbf{x}_j\boldsymbol{\beta})} \right) \right] \tag{3.3}$$

and

$$l''(\boldsymbol{\beta}) = -\sum_{i=1}^{N} \left[\delta_i \left(\frac{\sum_{k=1}^{N} y_k(t_j)\mathbf{x}_i\mathbf{x}_i^T \exp(\boldsymbol{\beta}\mathbf{x}_i)}{\sum_{k=1}^{N} y_k(t_j) \exp(\boldsymbol{\beta}\mathbf{x}_i)} - \frac{\sum_{k=1}^{N} y_k(t_j)\mathbf{x}_i \exp(\boldsymbol{\beta}\mathbf{x}_i) \sum_{k=1}^{N} y_k(t_j)\mathbf{x}_i^T \exp(\boldsymbol{\beta}\mathbf{x}_i)}{\left(\sum_{k=1}^{N} y_k(t_j) \exp(\boldsymbol{\beta}\mathbf{x}_i)\right)^2} \right) \right] \tag{3.4}$$

The penalty function $P(\boldsymbol{\beta})$ is also a continuous, concave function, but is not differentiable at all points. More specifically, it is not differentiable if $\boldsymbol{\beta} = \mathbf{0}$. We will later see how this is dealt with.

3.2 Extension of Goeman algorithm

One method to calculate the estimates $\boldsymbol{\beta}^{glasso}$ is based on an algorithm proposed by Goeman (2010), which makes use of both gradient ascent and the Newton-Raphson algorithm. Both algorithms are described below in its general form. The algorithm by Goeman is used for the regular lasso, and is adapted to the group lasso for multinomial logistic regression by Sanders (2009). Here we adapt this methodology for the proportional hazards model.

3.2.1 Gradient ascent

Consider a function $f(\theta)$ that we would like to optimize, i.e. we want to find its root. Further, let $u(\theta)$ be the gradient of this function. We would like to calculate the gradient vector $\mathbf{u}(\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is

a vector of p parameters. The gradient vector $\mathbf{u}(\boldsymbol{\theta})$ also consists of p elements and is defined as

$$\mathbf{u}(\boldsymbol{\theta}) = [u_1(\boldsymbol{\theta}), \ldots, u_p(\boldsymbol{\theta})]^T \tag{3.5}$$

where

$$u_j(\boldsymbol{\theta}) = \frac{\partial f(\boldsymbol{\theta})}{\partial \theta_j}, \quad j = 1, \ldots, p.$$

Gradient ascent is makes use of the fact that when at a point $\mathbf{a}$, the function f will increase fastest if one goes from that point in the direction of the gradient $\mathbf{u}(\mathbf{a})$. Let us suppose that we follow the gradient of $\mathbf{a}$ to arrive at point $\mathbf{b}$,

$$\mathbf{b} = \mathbf{a} + c \cdot \mathbf{u}(\mathbf{a})$$

It then follows that $\mathbf{u}(\mathbf{a}) \geq \mathbf{u}(\mathbf{b})$, given that c is a sufficiently small positive number, i.e. $c > 0$.

With the above in mind we turn to our function $f(\theta)$. We state that, in its simplest form, the gradient ascent algorithm updates our parameter vector $\boldsymbol{\theta}$ each iteration s by taking a step proportional to the gradient, optimizing the target function locally.

$$\boldsymbol{\theta}^{(s+1)} = \boldsymbol{\theta}^{(s)} + c \cdot \mathbf{u}(\boldsymbol{\theta}^{(s)}), \quad s \geq 0 \tag{3.6}$$

This gives a sequence of estimates $\boldsymbol{\theta}^{(0)}, \boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \ldots$, with ever smaller gradients $\mathbf{u}(\boldsymbol{\theta}^{(0)}) \geq \mathbf{u}(\boldsymbol{\theta}^{(1)}) \geq \mathbf{u}(\boldsymbol{\theta}^{(2)}) \ldots$, so hopefully the sequence of estimates will converge to the global optimum.

The algorithm iterates until some convergence criterion is met. Note that c should be sufficiently small in order to reach the optimum, if the optimum indeed exists, otherwise the algorithm will get close, but it will circle around the optimum indefinitely. A value of c that is too small, however, will cause the algorithm to take very small steps toward the optimum, taking many iterations to get there. Also, the convergence criterion may be reached before getting to the optimum, depending on how this criterion is defined. To avoid these problems one could also change the value of c per iteration.

Although this algorithm is very simple and easy to use, it has its disadvantages. Initially, the optimum is approached quite fast, but it then takes very small steps as it gets closer to the optimum, causing the algorithm to converge very slowly. In addition, gradient ascent requires the target function to be differentiable at each point. This may not always be the case, as we will see later.

3.2.2 Newton-Raphson

Again, consider the function $f(\theta)$ with its gradient $u(\theta)$ and the second-order derivative $H(\theta)$. The Newton-Raphson algorithm uses the gradient vector $\mathbf{u}(\boldsymbol{\theta})$ as defined in (3.5), as well as the Hessian matrix of the target function to find its root. In general, the Hessian matrix is defined as

$$\mathbf{H}(\boldsymbol{\theta}) = (H_{ij}(\boldsymbol{\theta})), \quad i, j = 1, \ldots, p \tag{3.7}$$

where

$$H_{ij}(\boldsymbol{\theta}) = \frac{\partial^2 f(\boldsymbol{\theta})}{\partial\theta_i \partial\theta_j}$$

The main idea underlying the Newton-Raphson algorithm is that the function $f(\boldsymbol{\theta})$ can be approximated by a quadratic function which is then minimized. This quadratic function is given by a second-order Taylor series. Say our current estimate is the vector $\boldsymbol{\theta}^{(s)}$, our next estimate is $\boldsymbol{\theta}^{(s+1)}$ and the difference between these two is denoted by $\Delta\boldsymbol{\theta}$, then this truncated Taylor series is given by

$$f(\boldsymbol{\theta}^{(s+1)}) = f(\boldsymbol{\theta}^{(s)}) + \Delta\boldsymbol{\theta}^T \cdot \mathbf{u}(\boldsymbol{\theta}) + \frac{1}{2}\Delta\boldsymbol{\theta} \cdot \mathbf{H} \cdot \Delta\boldsymbol{\theta}^T \tag{3.8}$$

The derivative of this function, denoted by $f'(\boldsymbol{\theta}^{(s+1)})$, is given by

$$f'(\boldsymbol{\theta}^{(s+1)}) = \mathbf{u}(\boldsymbol{\theta}) + \frac{1}{2}\mathbf{H} \cdot \Delta\boldsymbol{\theta} + \frac{1}{2}\mathbf{H}^T \cdot \Delta\boldsymbol{\theta} \tag{3.9}$$

Noting that the Hessian matrix is symmetric if the function is twice differentiable, this can be reduced to

$$f'(\boldsymbol{\theta}^{(s+1)})) = \mathbf{u}(\boldsymbol{\theta}) + \mathbf{H} \cdot \Delta\boldsymbol{\theta} \tag{3.10}$$

Remember we want to find the minimum of $f(\boldsymbol{\theta}^{(s+1)}$, and hence we want its derivative $f'(\boldsymbol{\theta}^{(s+1)}$ to equal zero. Solving the equation $\mathbf{u}(\boldsymbol{\theta}) + \mathbf{H} \cdot \Delta\boldsymbol{\theta} = 0$ for $\boldsymbol{\theta}^{(s+1)}$, we see that each iteration the current parameter vector $\boldsymbol{\theta}^{(s)}$ is updated by $\boldsymbol{\theta}^{(s+1)}$,

$$\boldsymbol{\theta}^{(s+1)} = \boldsymbol{\theta}^{(s)} - \mathbf{H}^{-1}(\boldsymbol{\theta}^{(s)})\mathbf{u}(\boldsymbol{\theta}^{(s)}), \quad s \geq 0 \tag{3.11}$$

until some convergence criterion is met.

One disadvantage of the Newton-Raphson algorithm is that it needs the initial guess, the starting values, to be relatively close to the optimum, otherwise this method may go in the wrong direction. When it does go in the right direction, it may overshoot the optimum. If the latter occurs the step size may be divided by 2 to prevent overshooting. Also, this method needs the second-order derivative to be defined. As we will see, this may not always be the case. Furthermore, evaluating the Hessian matrix may be difficult if $f(\theta)$ is a very complicated function of θ. Despite these disadvantages this method is attractive, because of its quick convergence once the optimum is approached.

3.2.3 Algorithm

Given the advantages and disadvantages of gradient ascent and Newton-Raphson, it may be obvious to use both in order to optimize the target function. Even though gradient ascent is relatively inefficient in that is needs lots of iterations to reach the optimum, it does approach the optimum fast, even passing through points where the usual gradient $\mathbf{g}(\boldsymbol{\beta})$ is not defined. Once gradient ascent approaches the optimum and stays within a subdomain of gradient continuity, one may switch to

Newton-Raphson using the current estimate $\boldsymbol{\beta}^{(s)}$ as starting values. As noted, this method is quick to converge, especially if the initial values are close to the optimal values, which is the case when first using gradient ascent.

Before turning to the final algorithm we first look at the way the penalty function $P(\boldsymbol{\beta}) = \sum_{g=1}^{G} \lambda_g \left|\left|\boldsymbol{\beta}_g\right|\right|_2$ is handled by this particular algorithm. Recall that the penalty function $P(\boldsymbol{\beta})$, like the log partial likelihood $l(\boldsymbol{\beta})$, is a continuous, concave function, but is not differentiable at all points. More specifically, it is not differentiable if $\boldsymbol{\beta} = \mathbf{0}$. Also, the penalty function causes the Euclidean space spanned by the predictors to be divided in subdomains. The boundaries are defined as the positions where a group of variables $\boldsymbol{\beta}_g = \mathbf{0}$. It is only within such a subdomain that the approximation used by the Newton-Raphson algorithm is meaningful.

If the gradient of the penalized log partial likelihood $l_{pen}(\boldsymbol{\beta})$ is indeed defined, then the gradient of $l_{pen}(\boldsymbol{\beta})$ may be calculated using the gradient for the unpenalized log partial likelihood, $l(\boldsymbol{\beta})$. The gradient for group g, $\mathbf{g}_g(\boldsymbol{\beta})$, is then calculated as

$$\mathbf{g}_g(\boldsymbol{\beta}) = \begin{cases} \mathbf{u}_g(\boldsymbol{\beta}) - \lambda_g \frac{\boldsymbol{\beta}_g}{\left|\left|\boldsymbol{\beta}_g\right|\right|_2} & \text{if } \boldsymbol{\beta}_g \neq \mathbf{0} \\ \mathbf{u}_g(\boldsymbol{\beta}) - \lambda_g \frac{\mathbf{u}_g(\boldsymbol{\beta})}{\left|\left|\mathbf{u}_g(\boldsymbol{\beta})\right|\right|_2} & \text{if } \boldsymbol{\beta}_g = \mathbf{0} \text{ and } \left|\left|\mathbf{u}_g(\boldsymbol{\beta})\right|\right|_2 > \lambda_g \\ \mathbf{0} & \text{otherwise} \end{cases} \tag{3.12}$$

where $\mathbf{u}(\boldsymbol{\beta})$ is the gradient of $l(\boldsymbol{\beta})$.

Remember that gradient ascent updates the parameter vector $\boldsymbol{\beta}$ at iteration s by taking a step proportional to the gradient as described by 3.6. When updating the parameter vector $\boldsymbol{\beta}$ using gradient ascent we choose the value for c such that we optimize the target function locally. The optimum within a subdomain is given by t_{opt}, which we define by

$$t_{opt} = \frac{\mathbf{g}(\boldsymbol{\beta})^T \mathbf{g}(\boldsymbol{\beta})}{\mathbf{g}(\boldsymbol{\beta})^T \mathbf{X}^T \mathbf{W} \mathbf{X} \mathbf{g}(\boldsymbol{\beta})}$$

This leads to the following algorithm, where the coefficients on iteration s are updated as follows.

$$\boldsymbol{\beta}^{(s+1)} = \begin{cases} \boldsymbol{\beta}_{NR}^{(s+1)} & \text{if } \operatorname{sign}(\boldsymbol{\beta}_{NR}^{(s+1)}) = \operatorname{sign}(\boldsymbol{\beta}_{+}^{(s)}) \\ \boldsymbol{\beta}^{(s)} + t_{opt} \cdot \mathbf{g}(\boldsymbol{\beta}^{(s)}) & \text{otherwise} \end{cases} \tag{3.13}$$

Update until convergence. Here $\boldsymbol{\beta}_{NR}$ indicates an update using Newton-Raphson, and $\boldsymbol{\beta}_{+}$ indicates the set of active variables, i.e. the set of nonzero coefficients.

3.3 A quasi-Newton method: L-BFGS

As stated at the beginning of this section, the focus of this paper is on the extension of Goeman's algorithm as described above. However, this method is not yet fully implemented. Due to time

constraints we have used the L-BFGS algorithm to analyze the data in section 4. This algorithm is a quasi-Newton method and is a limited-memory variant of the Broyden-Fletcher-Goldfarb-Shannon (BFGS) method. Quasi-Newton methods, like the regular Newton-Raphson method, find the root of a function, i.e. the point where the gradient is zero. Recall that the Newton-Raphson algorithm updates our parameter estimates at iteration s by

$$\boldsymbol{\theta}^{(s+1)} = \boldsymbol{\theta}^{(s)} - \mathbf{H}^{-1}(\boldsymbol{\theta}^{(s)})\mathbf{u}(\boldsymbol{\theta}^{(s)}), \quad s \geq 0$$

Note that this method requires the computation of the inverse of the Hessian matrix $\mathbf{H}^{-1}$. As mentioned earlier, the computation of the Hessian matrix may be difficult or costly, and so may not always be done. Quasi-Newton methods do not compute the Hessian matrix directly. Instead, these methods approximate it, or the inverse of the Hessian matrix, by using successive gradient vectors.

At every iteration s several vectors are stored. We will denote the gradient vector at iteration by $\mathbf{u}^{(s)}$, and the parameter vector at the same iteration by $\boldsymbol{\theta}^{(s)}$. Quasi-Newton methods store the difference between parameter vectors on iterations $s+1$ and s, denoted by $\boldsymbol{\delta}^{(s)}$

$$\begin{aligned}\boldsymbol{\delta}^{(s)} &= \boldsymbol{\theta}^{(s+1)} - \boldsymbol{\theta}^{(s)} \\ &= \mathbf{H}^{-1}(\boldsymbol{\theta}^{(s)})\mathbf{u}(\boldsymbol{\theta}^{(s)})\end{aligned}$$

and the difference in gradient vectors on iterations $s+1$ and s, denoted by $\boldsymbol{\eta}^{(s)}$.

$$\boldsymbol{\eta}^{(s)} = \mathbf{u}(\boldsymbol{\theta}^{(s+1)}) - \mathbf{u}(\boldsymbol{\theta}^{(s)})$$

An estimate of the Hessian matrix at the next iteration $s+1$ would be the solution to the set of linear equations

$$\mathbf{H}^{(s+1)}\boldsymbol{\delta}^{(s)} = \boldsymbol{\eta}^{(s)}$$

Several solutions have been proposed, but the solution used in the BFGS method is considered the best. Here we will consider the limited memory version BFGS as described by Liu and Nocedal (1989). The difference between the two is that the normal BFGS method computes and stores an $p \times p$ approximation to the Hessian matrix, while the limited memory BFGS method only stores a few pair of vectors that ought to represent this approximation implicitly. For an in-depth discussion of the algorithm the reader is referred to Liu and Nocedal (1989).

4 Application: the CAREMA case-cohort study

4.1 Description

The case-cohort study by Vaarhorst et al. (2011) is a prospective study that focuses on predicting the risk of future coronary heart disease (CHD). More specifically, predicting the risk of having a

fatal CHD event in the next 10 years. This enables identifying people at risk for future CHD and treating them accordingly. Traditionally, prediction has been done using a risk score based on age, gender, smoking status, systolic blood pressure, total cholesterol and high-density lipoprotein (HDL) cholesterol. Research shows that heritability also plays a role in the prediction of CHD (e.g. Levy et al. (2009)). Since this is known to be true, the prediction of CHD might be improved by taking into account genetic markers that are associated with CHD or its intermediate risk factors.

While several SNPs are known to be associated with CHD, their individual effects on the outcome generally are relatively small. As is custom in the field of epigenetics, we will combine the information contained in these SNPs by constructing and comparing several genetic risk scores (GRS). Several risk scores have been proposed that are associated with CHD. However, this relation seems to disappear when adjusted for the traditional risk factors. Also, these risk scores often do not improve risk classification, which is the main focus of this study. Therefore, we turn to different risk scores. We try to construct a GRS using several methods. One way of doing so is by summing up the number of risk alleles of all SNPs. Denote this genetic risk score by *GRS All*. Obviously, this is a crude way of constructing a GRS, for it assumes that all SNPs have the same impact on CHD, which is highly unlikely. Another way of constructing a GRS is by creating a weighted sum of the risk alleles. The weights may be given by the regression coefficients obtained by performing a regular Cox regression, obtaining a GRS denoted by *GRS Cox*. This has two disadvantages. Firstly, when there are many SNPs, i.e. the data are high-dimensional, then this method fails to work, because the coefficients may not be estimated. Also, we would like a sparse solution. I.e. we want to include as few predictors as possible. Therefore, we prefer using the lasso or the group lasso to determine the weights given to each SNP, giving the genetic risk scores *GRS Lasso* and *GRS gLasso*. Hence, we will compare four different genetic risk scores.

So the aim of this study is to see whether including information on genetic markers improves prediction accuracy, independent of the already established risk score based on the traditional risk factors. Several methods are available to assess this, e.g. the net reclassification index (NRI) and the integrative discriminatory index (IDI). We will use the NRI, developed by Pencina et al. (2008). For an overview of the computation of the NRI, see Appendix D. The NRI requires that we specify *a priori* categories of 10-year risk of CHD. We will follow Pencina et al. and Vaarhorst et al. by choosing four categories: 0-5, 5-10, 10-20, >20 percent 10-year risk.

In order to investigate the research question a cohort of 21,148 subjects was established from the Cardiovascular Registry Maastricht (CAREMA) study population. This population comprises subjects of two monitoring projects: the monitoring project on cardiovascular disease risk factors (PPHVZ) and the monitoring project on chronic disease risk factors (Morgen project). At the time

of follow-up the number of subjects eligible for the cohort was 15,236. From the eligible cohort a sub-cohort of 2421 eligible subjects (15.9 %) was selected. This sub-cohort consists of 1285 men and 1163 women. The median time to follow-up was 12.11 years.

Using the GWAS Catalog, available at www.genome.gov/gwastudies, 126 SNPs have been selected, all of which have been showed to have an association with CHD or its intermediate risk factors. Some SNPs were in complete linkage disequilibrium, i.e. were strongly correlated with other SNPs in the selection, and were subsequently removed from this selection. Other SNPs were excluded due to failed genotyping, a success rate lower than 90%, or being out of Hardy-Weinberg equilibrium for subjects without an event, leaving 113 SNPs.

4.2 Missing data

As is often the case, some of the data for the SNPs were missing. Several methods exist to deal with missing data, such as mean substitution, complete-case analysis, available-case analysis, and others. Some of these methods are very easy to use, but may not yield statistically valid results. Multiple imputation (MI) is a method that sometimes may be difficult to implement, but does yield valid results and has the advantage that it allows the researcher to use the full data. This causes MI to be efficient, in that it uses all information, including the information from subjects who do not have complete data (Kenward and Carpenter, 2007). Using all available information also corrects for bias.

The general idea of multiple imputation is that one fills in the missing data with several values (say, m values per missing value in the data), leading to m differently imputed data sets. Normally the gathered data set is used to impute the missing observations, but one may also use external information to improve accuracy of these imputations. With a case like the current one, i.e. when one needs to impute SNP data, the HapMap may be used. This is data on over 3,1 million SNPs measured for four different panels of subjects, where each panel consists of a number of subjects from different parts of the world, though most subjects are of northern and western European ancestry. The HapMap is used to infer the allele count for the missing SNP data. However, this was not possible here.

Specifically, we need to generate a posterior predictive distribution of the missing values conditional on the values that we have observed. We then draw values from this distribution which we call the imputed values. An algorithm for doing this is described by Raghunathan et al. (2001). We will not discuss this method here, since the researchers decided to use random imputation.

Random imputation is a method where the imputed data for a single variable are randomly sampled from the observed data for this variable. Three reasons for doing this are given. Firstly, in

the current data set not many values were missing after standard quality control. The results would thus not be affected greatly. Furthermore, the genotyped SNPs were not in linkage disequilibrium, so that they would not contain much information about any other SNPs anyway. Lastly, any external information like the HapMap could not be used here.

We apply the proportional hazards model with an L_1 constraint on the regression coefficients on each data set, obtaining m different estimates of the parameters of interest. Rubin (1987) developed the rules for combining these estimates. Say we have m estimates of our parameter θ, then our overall estimate for this parameter is the simple mean $\bar{\theta} = \frac{1}{m}\sum_{j=1}^{m}\theta_j$. In the case of the lasso we do not calculate variances of our estimates. The reason for this is that the estimates in regularized methods such as lasso are strongly biased. In order to obtain the estimates we had to reduce the variance of these estimates by introducing a lot of bias. Surely, variances may be calculated, e.g. using bootstrap, but the results may be misleading in that this method may produce small variances. Again, see Appendix C. In cases where we would calculate variances, we would also obtain m estimates of the variance $Q^{(j)}$ of each θ_j. Then the variance of our overall estimate $\bar{\theta}$, denoted T, would be a function of the between imputation variance B and the within imputation variance U.

$$T = \left(1 + \frac{1}{m}\right) B + \bar{U}$$

where $B = \frac{1}{m-1}\sum_{j=1}^{m}(Q^{(j)} - \bar{Q})^2$ and $\bar{U} = \frac{1}{m}\sum_{j=1}^{m} U^{(j)}$.

4.3 Problem

The process of imputation was performed 20 times, yielding 20 differently imputed data sets. Due to the high dimensionality of the data lasso is preferred to construct a GRS. However, applying regular lasso for the proportional hazards model yields different selections of variables and reasonably different regression coefficients for those few variables that indeed are selected in all data sets. This is depicted in Figure 2. In this figure the SNPs are on the horizontal axis and the 20 data sets on the vertical axis, going from bottom to top. The darker the color, the higher the absolute value of the coefficient. White areas indicate a SNP that is not selected, i.e. its coefficient is equal to zero. The figure illustrates the problem at hand. We can see that some SNPs indeed are included or excluded in all data sets, but some SNPs seem to be included in the model when using one data set, while being excluded from the model using another data set. In models without regularization we may just take the average of the different estimates (Kenward and Carpenter, 2007, also see this paper for a justification of this approach). However, the lasso produces biased results, causing this simple method not to be valid anymore. Using the group lasso we could resolve this problem by defining the same SNP in the 20 different data sets to be one group of variables.

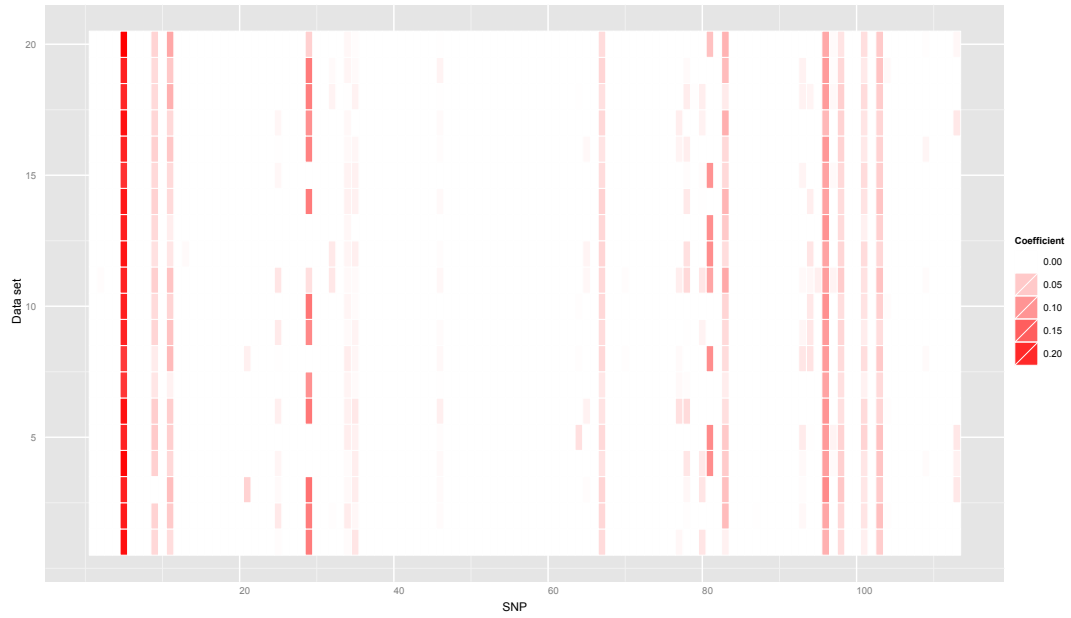


Figure 2: Absolute values of SNP regression coefficients

4.4 Combining data

As seen in Figure 2 the results differ, depending on what data set we use. To overcome this, we use the group lasso, where each group consists of the same SNP in each of the imputed data sets. This way we either exclude or include a particular SNP in all data sets and obtain an estimate of its regression coefficient by averaging the coefficients within each group. Not only should this lead to sparser models, thus enhancing interpretability, but it should also predict the data at least as good as regular lasso. Furthermore, it avoids the awkward situation where the model depends on the imputation of the data.

Before we go to the group lasso, we shall look into a different option. One could consider doing other analyses using the regular lasso, where the goal is to use the lasso to obtain one vector of regression coefficients. To achieve this, we have to manipulate the data. The option considered here is to stack the data sets on top of each other. For computational reasons we choose to use only five data sets. This leads to a data set of dimension $5n \times p$, as depicted below. Let us denote this data set by $\tilde{\mathbf{X}}$, then $\tilde{\mathbf{X}}$ looks as follows:

$$\tilde{\mathbf{X}} = \begin{bmatrix} \mathbf{X}^{(1)} \\ \vdots \\ \mathbf{X}^{(5)} \end{bmatrix}$$

Here each data set $\mathbf{X}^{(j)}$, $j = 1, \ldots, 5$, is a submatrix of dimension $n \times p$.

$$\mathbf{X}^{(j)} = \begin{bmatrix} x_{11} & \ldots & x_{1p} \\ \vdots & \ddots & \vdots \\ x_{n1} & \ldots & x_{np} \end{bmatrix}$$

Note that this matrix $\tilde{\mathbf{X}}$ contains the predictors only. The survival data is stacked in a similar fashion.

Applying the regular penalized proportional hazards model indeed leads to only one vector of coefficients. The optimal value for λ is obtained by cross-validation. In creating the folds, each individual subject is in a particular fold, taking into account that each subject is in the large data set 5 times. I.e. when a particular subject is left out, that means that 5 rows are left out of the data matrix. With this newly created data set we may apply the regular lasso, for in this way we obtain one estimate for each SNP.

While in this case using the regular lasso may work, we would like a more elegant, and more general solution to our problem. Hence we will use the group lasso. Another option, besides creating the $5n \times p$ data matrix, is to create a large $5n \times 5p$ data set, where each of the five original data sets $\mathbf{X}^{(j)}$ are on the diagonal, giving a block-diagonal matrix, as depicted below. All entries off the blocks along the diagonal are zero. Again, the survival data are also stacked, but are not in the data set depicted below. We could then average the 5 coefficients for each SNP in order to obtain one estimate for each coefficient.

$$\begin{bmatrix} \mathbf{X}^{(1)} & & \emptyset \\ & \ddots & \\ \emptyset & & \mathbf{X}^{(5)} \end{bmatrix}$$

When having such data sets, as the above two artificially created data sets, we may consider stratifying the data, taking each data set to be a separate stratum. As showed in Figure 2, analyzing each data set separately yields different coefficients, but also different baseline hazards for each data set. Remember that in the proportional hazards model, the effect of the predictors are assumed to act multiplicatively on the baseline hazard. Simply averaging the coefficients of all data sets would not make sense, since they all act on different baseline hazards. When considering each data set to be a stratum, which indeed would give only one vector or regression coefficients, we are still left with the problem of having several coefficients, in this case several baseline hazards. We opt to have one baseline hazard, for then the coefficients are comparable.

4.5 Analysis

As mentioned earlier, we try to investigate whether the genetic risk scores have any additional predictive value over the traditional risk scores using the NRI. First, we performed a Cox regression using the traditional risk factors only. The results are given in Table 2. We use these results to compute the hazard for each subject, which we will later compare to the hazard based on a larger model. As

Risk factor	Beta	SE	Z-value	P-value	HR	Lower bound	Upper bound
Age	-0.0265	0.0109	-2.4232	0.0154	0.9738	0.9532	0.9949
Systolic blood pressure	0.0210	0.0024	8.5743	0.0000	1.0212	1.0163	1.0261
Smoking status	0.6300	0.0813	7.7495	0.0000	1.8777	1.6011	2.2021
Gender	0.9150	0.0998	9.1699	0.0000	2.4967	2.0532	3.0360
Total cholesterol	0.2839	0.0309	9.1937	0.0000	1.3284	1.2503	1.4113
HDL cholesterol	-2.0958	0.1702	-12.3153	0.0000	0.1230	0.0881	0.1717
Diabetes	1.0918	0.2158	5.0604	0.0000	2.9796	1.9521	4.5479
Family history	0.3467	0.0651	5.3234	0.0000	1.4143	1.2449	1.6069

Table 2: Coefficients of traditional risk factors

can be seen from the table, all traditional risk factors have reached statistical significance at the 5% level, i.e. they are all associated with future CHD. As a next step, we fit several other models, each including the above risk factors and one of the GRS described above. The results of these analyses are given in Table 3. Note that the results for the risk factors are not included here. However, they did not change dramatically from those given in Table 2 and all remained statistically significant. Table 3 shows that all genetic risk scores, except for GRS All, are statistically significant at the 5%

GRS	Coef	SE	Z-value	P-value	HR	Lower bound	Upper bound
GRS All	0.3002	0.6308	0.4760	0.6341	1.3502	0.3922	4.6485
GRS Cox	0.8804	0.0650	13.5524	0.0000	2.4118	2.1235	2.7393
GRS Lasso	1.7073	0.2837	6.0187	0.0000	5.5139	3.1623	9.6142
GRS gLasso	1.5841	0.1722	9.1988	0.0000	4.8749	3.4784	6.8320

Table 3: Coefficients of the genetic risk scores

level, even when adjusted for the other risk factors. Remember that GRS All was created by taking the sum over the number of risk alleles per subject. Since this is a very crude way of combining the

information in the SNPs, this result is not surprising. These results suggest that at least some SNPs are associated with CHD, and have some predictive value in addition to the other, traditional risk factors.

While these results seem very promising, we must keep in mind that we are particularly interested in risk classification. I.e. we would like to be able to predict what risk category a subject belongs to. We calculate the net reclassification index for the different models including a genetic risk score using the four categories 0-5, 5-10, 10-20, and >20 percent 10-year risk. For every GRS we calculate the 10-year risk of experiencing CHD and assign each subject to one of the above risk categories. We then calculate the NRI as defined by Pencina et al. (2008) for all different GRS. The NRI has the problem of not distinguishing between reclassification of cases and non-cases, so we will give the NRI for both groups separately. The results are given in Table 4. Notice that the values for all NRIs are fairly low, the largest being 8.02%, for GRS Cox. As mentioned earlier, this method is not preferred since this type of data often is high-dimensional. We see that the next best GRS is GRS gLasso, with an NRI of 6.71%, outperforming the GRS Lasso. When looking closer, we see that the group lasso outperforms the lasso for both cases and non-cases. Hence, based on these results we would prefer the group lasso over the regular lasso.

From a biological perspective these results may mean that the SNPs indeed have some additional predictive value over the traditional risk factors. One thing that does stand out though, is that the cases are less frequently reclassified correctly than the non-cases. Indeed, we see that almost none of the risk scores significantly contribute to the reclassification of the cases. So while all but the first GRS reach statistical significance at the 5% level, this is mostly due to the reclassification of the non-cases. Although any correct reclassification is desirable, we would mostly like the risk prediction of the cases to be improved, for it are those subjects that are likely to experience coronary heart disease.

5 Discussion

In this paper we developed an extension to the algorithm for L_1 penalized estimation by Goeman (2010), where we adjusted the algorithm such that we may obtain estimates for grouped variables, instead of the single variables as is the case in the regular lasso. Other authors already presented different algorithms for L_1 penalized estimation for grouped variables (e.g. the block-coordinate gradient descent algorithm by Meier et al. (2008) and the group least angle regression selection and the group non-negative garotte by Yuan and Lin (2006)), but they have done so only for the general linear model and for the logistic model. This paper presented an adaptation of the model developed by

Genetic risk score		Reclassification		NRI		
		Up	Down	Value	SE	P-value
Simple sum of	Cases	0.0000	0.0047	-0.0047	0.0027	0.0833
allele counts	Non-cases	0.0039	0.0062	0.0022	0.0032	0.3841
	Total	-	-	-0.0025	0.0025	0.3200
Cox regression	Cases	0.0611	0.0705	-0.0094	0.0146	0.3346
	Non-cases	0.0903	0.1817	0.0914	0.0126	0.0000
	Total	-	-	0.0820	0.0125	0.0000
Regular lasso	Cases	0.0376	0.0376	0.0000	-0.0141	0.0146
	Non-cases	0.0538	0.0735	0.0196	0.1004	0.0126
	Total	-	-	0.0196	0.0103	0.0576
Group lasso	Cases	0.0643	0.0533	0.0110	0.0117	0.2850
	Non-cases	0.0695	0.1256	0.0561	0.0091	0.0007
	Total	-	-	0.0671	0.0124	0.0000

Table 4: The net reclassification index per GRS

Sanders (2009) for the multinomial logistic model to fit the proportional hazards model for survival data. The current method is easily adapted to fit the general linear model and generalized linear models, making it a widely applicable method.

The application of the current method presented in this paper is not a typical one. In this case the groups of variables were defined as the differently imputed SNPs from different imputed data sets. Normally one would have variables that are considered a group for other reasons. For example, because they are dummy variables used to represent a categorical variable. However, though not a typical application, this does show the potential uses for the group lasso.

The use of the NRI to assess risk predictions is quite popular in medical research. However, this metric does not come without its disadvantages. Firstly, as Pepe (2011) laid out, the interpretation of the NRI is problematic. The NRI is a sum of two separate reclassification measures: for events and non-events. Hence, reporting both components is necessary to understand what subjects are reclassified correctly. Furthermore, improved risk reclassification does not necessarily mean that the new model is better at predicting the event of interest. One of the motivations for this claim is that the NRI depends on arbitrarily chosen risk categories. This will inevitably influence the NRI, so that any conclusions drawn based solely on the NRI may be inaccurate, as Pencina et al. (2011) himself shows. Hence, the use of various measures of risk reclassification is advised.

A The bias-variance decomposition

As mentioned in the introduction, ill-posed problems require additional restrictions about our model in order to obtain a solution to our problem. In the lasso we have the following optimization problem:

$$\boldsymbol{\beta} = \arg\max_{\boldsymbol{\beta}} \{l(\boldsymbol{\beta})\} \quad \text{subject to} \, ||\boldsymbol{\beta}||_1 \leq z$$

We restrict our model such that the L_1-norm of our coefficient vector does not exceed some predefined number z. This causes our estimates to be biased, hence leading to smaller variances, which makes the model estimable. For the sake of simplicity, we will consider To see why introducing bias leads to reduced variances, we will consider the mean-squared error (MSE). Or, more specifically, the bias-variance decomposition of the MSE. Let us defined the MSE as the expectation of the squared difference between the true value of our parameter, say, θ, and its estimate, $\hat{\theta}$:

$$\text{MSE} = E\left\{\left(\hat{\theta} - \theta\right)^2\right\}$$

The MSE may be decomposed into two terms: the squared bias of our parameter estimate and its variance. Since the MSE equals the sum over all expectations of our parameters, it is sufficient to show that the MSE may be decomposed into the bias and variance of a single parameter θ.

$$\begin{aligned}
E\left\{\left(\hat{\theta} - \theta\right)^2\right\} &= E\left\{\left(\theta - E(\hat{\theta})\right) + \left(E(\hat{\theta} - \theta)\right)^2\right\} \\
&= E\left\{\left(\hat{\theta} - \theta\right)^2 + \left(E(\hat{\theta}) - \theta\right)^2 + \left(\hat{\theta} - E(\hat{\theta})\right)\left(E(\hat{\theta}) - \theta\right)\right\} \\
&= \text{Var}(\hat{\theta}) + \text{Bias}^2(\hat{\theta}) + E\left\{\hat{\theta}E(\hat{\theta}) - E(\hat{\theta})^2 - \theta\hat{\theta} + E(\hat{\theta})\theta\right\} \\
&= \text{Var}(\hat{\theta}) + \text{Bias}^2(\hat{\theta}) + E(\hat{\theta})^2 - E(\hat{\theta})^2 - \theta E(\hat{\theta}) - \theta E(\hat{\theta}) \\
&= \text{Var}(\hat{\theta}) + \text{Bias}^2(\hat{\theta})
\end{aligned} \tag{A.1}$$

This shows that if we want to reduce the variance, as in regularization, we may do that by simply increasing the bias. Note that the MSE does not change, only the parts of which it consists.

B A Bayesian view on regularization

Suppose we have some data, $S = \{(x_i, y_i)\}_{i=1}^n$, which is the set of pairs of predictors and the response. We may collect the data in two matrices, $\mathbf{X}$ and $\mathbf{Y}$, of dimensions $n \times p$ and $n \times 1$, respectively. Further, let $\boldsymbol{\beta}$ be a vector of regression coefficients in $\mathbb{R}^p$, then we denote the joint distribution over the responses $\mathbf{Y}$, conditional on the predictors $\mathbf{X}$ and the parameters $\boldsymbol{\beta}$ by $P(\mathbf{Y}|\mathbf{X}, \boldsymbol{\beta})$. We have a model that relates the observed quantities to some unobserved quantities, say the regression coefficients $\boldsymbol{\beta}$. Now, we want an estimator of our parameters of interest that best maps back our

observed data to an estimate of $\boldsymbol{\beta}$. So far, this is standard practice, we have not seen anything new yet. Now let us suppose that we have the following regression model

$$y_i = f(x_i) + \epsilon, \quad \epsilon \sim \mathcal{N}\left(0, \sigma_\epsilon^2\right) \tag{B.1}$$

where we have multivariate normally distributed responses.

$$\mathbf{Y}|\mathbf{X}, \theta \sim \mathcal{N}\left(\mathbf{X}\theta, \sigma_\epsilon^2 \mathbf{I}\right) \tag{B.2}$$

In frequentist statistics we usually obtain maximum likelihood estimates $\hat{\boldsymbol{\beta}}$ is by maximizing the likelihood of our data under some model. The likelihood of any fixed parameter vector θ is

$$L(\theta|\mathbf{X}) = P(\mathbf{Y}|\mathbf{X}, \theta). \tag{B.3}$$

This means that the likelihood is equal to the conditional distribution of the response variable, given that we have a fixed θ and fixed observed data $\mathbf{X}$. The Bayesian way of obtaining our estimates is by finding the mode of the posterior distribution $P(\boldsymbol{\beta}, \mathbf{X})$. This is decomposed as

$$P(\theta, \mathbf{X}) = P(\mathbf{X}|\theta)P(\theta) \tag{B.4}$$

where $P(\theta)$ is some prior distribution for our parameter vector θ.

Now let us assume the same as in (B.1), i.e. we assume a general regression model with all its well-known assumptions. We then have the following likelihood $L(\theta|\mathbf{X})$

$$\begin{aligned} L(\theta|\mathbf{X}) &= \mathcal{N}\left(\mathbf{Y}; \mathbf{X}\theta, \sigma_\epsilon^2 \mathbf{I}\right) \\ &\propto \exp\left(-\frac{1}{2\sigma_\epsilon^2} ||\mathbf{Y} - \mathbf{X}\theta||_2^2\right) \end{aligned} \tag{B.5}$$

In this case the maximum likelihood estimator is given by:

$$\arg\min_\theta \left\{ \frac{1}{2} ||\mathbf{Y} - \mathbf{X}\theta||_2^2 \right\} \tag{B.6}$$

When imposing some constraint on the model, e.g. an L_2 constraint on the parameters θ, we obtain the following maximum likelihood estimator:

$$\arg\min_\theta \left\{ \frac{1}{2} ||\mathbf{Y} - \mathbf{X}\theta||_2^2 + \lambda ||\theta||_2^2 \right\} \tag{B.7}$$

There is a model for $\mathbf{Y}$ and θ that yields this regularized least squares solution, and is very similar to the model used previously, namely

$$\begin{aligned} &\exp\left(-\frac{1}{2\sigma_\epsilon^2} ||\mathbf{Y} - \mathbf{X}\theta||_2^2 + \lambda ||\theta||_2^2\right) \\ =&\exp\left(-\frac{1}{2\sigma_\epsilon^2} ||\mathbf{Y} - \mathbf{X}\theta||_2^2\right) \exp\left(\lambda ||\theta||_2^2\right) \end{aligned} \tag{B.8}$$

This may look familiar, because it has the same form as (B.4), $P(\mathbf{X}|\theta)P(\theta)$. Indeed, we now see that we may view the regularized least squares problem as a Bayesian problem by noting that the constraint placed on the model is much like a prior distribution in Bayesian statistics.

C Standard errors and confidence intervals in regularization

In statistics one is usually interested in the accuracy of the estimates obtained of the parameters. Classically, this accuracy is represented by a standard error of that estimate. With that standard error some $(1-\alpha)\times 100\%$ confidence interval can be created, where α represents the probability of a Type I error, which is closely linked to the standard practice of testing in frequentist statistics. When placing additional constraints on the model, as is done in lasso, it is hard to say something about accuracy. Firstly, because the exact standard errors are hard to obtain. The standard errors are obtained from the Hessian matrix, but the variances in this matrix underestimate the true variances. We have sacrificed some bias in order to obtain a lower variance, so the standard errors will be smaller than they truly are. One could consider bootstrapping to obtain an estimate of the standard error. The disadvantage of this procedure is that it takes a lot of time, especially when having large data sets and thus when regularization is needed. If one were to obtain standard errors using the bootstrap, then these standard errors would still be of little use, since our estimates of the regression coefficients are biased to begin with. Having a confidence interval around a biased estimate gives little information when one is interested in testing some hypothesis about this estimate.

D Net reclassification index

One way to evaluate the additional predictive ability of some additional predictor or predictors is by using the net reclassification improvement (NRI) by Pencina et al. (2008). In our case we would like to assess the additional predictive ability of the SNPs. We have predicted 10-year risk of CHD based on two models: one only including the traditional risk score, and one model including the risk score and the SNPs. The NRI assesses reclassification tables constructed separately for subjects who have experienced the event of interest and those who have not, and quantifies the proper movement between categories. For event subjects one would want them to be in the highest class possible, so upward movement is desired, i.e. going from a low-risk category, based on the first model, to a high-risk category, based on the second, larger model. For non-event subjects the reverse is desired. The NRI is then calculated as

$$\text{NRI} = [P(\text{up} \mid D=1) - P(\text{down} \mid D=1)] - [P(\text{up} \mid D=0) - P(\text{down} \mid D=0)] \tag{D.1}$$

where the above probabilities are estimated by

$$\begin{aligned}
\hat{P}(\text{up} \mid D = 1) &= \frac{\text{number of events moving up}}{\text{number of events}} \\
\hat{P}(\text{down} \mid D = 1) &= \frac{\text{number of events moving down}}{\text{number of events}} \\
\hat{P}(\text{up} \mid D = 0) &= \frac{\text{number of non-events moving up}}{\text{number of non-events}} \\
\hat{P}(\text{down} \mid D = 0) &= \frac{\text{number of non-events moving down}}{\text{number of non-events}}
\end{aligned}$$

The estimated standard error of the NRI is given by

$$\text{SE}_{\text{NRI}} = \sqrt{\frac{\hat{P}(\text{up} \mid D = 1) + \hat{P}(\text{down} \mid D = 1)}{\text{number of events}} + \frac{\hat{P}(\text{up} \mid D = 0) + \hat{P}(\text{down} \mid D = 0)}{\text{number of non-events}}} \tag{D.2}$$

An asymptotic test for the NRI is then given by

$$z = \frac{\hat{\text{NRI}}}{\text{SE}_{\text{NRI}}}$$

where, under the null hypothesis of no improvement of predictive accuracy, $z \sim \mathcal{N}\left(0, \sigma^2\right)$. We thus reject the null hypothesis for large values of z.

References

Barlow, W. E. (1994). Robust variance estimation for the case-cohort design. *Biometrics 50*, 1064–1072.

Bøvelstad, H., S. Nygard, H. Storvold, M. Aldrin, O. Borgan, A. Frigessi, and O. Lingjaerde (2007). Predicting survival from microarray data - a comparative study. *Bioinformatics 23*(16), 2080–2087.

Brent, R. P. (1973). *Algorithms for minimization without derivatives*. Prentice Hall.

Cox, D. (1972). Regression models and life-tables. *Journal of the Royal Statistical Society Series B-Statistical Methodology 34*(2), 187–220.

Goeman, J. J. (2010). L_1 penalized estimation in the cox proportional hazards model. *Biometrical Journal 52*(1), 70–84.

Graf, E., C. Schmoor, W. Sauerbrei, and M. Schumacher (1999). Assessment and comparison of prognostic classification schemes for survival data. *Statistics in Medicine 18*(17), 2529–2545.

Hoerl, A. E. and R. W. Kennard (1970). Biased estimation for nonorthogonal problems. *Technometrics 12*(1), 55–67.

Kenward, M. G. and J. Carpenter (2007). Multiple imputation: current perspectives. *Statistical Methods in Medical Research 16*(3), 199–218.

Levy, D., G. B. Ehret, K. Rice, G. C. Verwoert, L. J. Launer, A. Dehghan, N. L. Glazer, A. C. Morrison, A. D. Johnson, T. Aspelund, Y. Aulchenko, T. Lumley, A. Köttgen, R. S. Vasan, F. Rivadeneira, G. Eiriksdottir, X. Guo, D. E. Arking, G. F. Mitchell, F. U. S. Mattace-Raso, A. V. Smith, K. Taylor, R. B. Scharpf, S. J. Hwang, E. J. G. Sijbrands, J. Bis, T. B. Harris, S. K. Ganesh, C. J. O'Donnel, A. Hofman, J. I. Rotter, J. Coresh, E. J. Benjamin, A. G. Uitterlinden, G. Heiss, C. S. Fox, J. C. M. Witteman, E. Boerwinkle, T. J. Wang, V. Gudnason, M. G. Larson, A. Chakravarti, B. M. Psaty, and C. M. Van Duijn (2009). Genome-wide association study of blood pressure and hypertension. *Nature Genetics 41*(6), 677–687.

Liu, D. C. and J. Nocedal (1989). On the limited memory bfgs method for large scale optimization. *Mathematical Programming 45*, 503–528.

Meier, L., S. van de Geer, and P. Bühlmann (2008). The group lasso for logistic regression. *Journal of the Royal Statistical Society Series B-Statistical Methodology 70*, 53–71.

Pencina, M. J., R. B. D'Agostino Sr., R. B. D'Agostino Jr., and R. S. Vasan (2008). Evaluating the added predictive ability of a new marker: From area under the roc curve to reclassification and beyond. *Statistics in Medicine 27*(2), 157–172.

Pencina, M. J., R. B. D'Agostino Sr., and E. W. Steyerberg (2011). Extensions of the net reclassification improvement calculations to measure usefulness of new biomarkers. *Statistics in Medicine 30*(1), 11–21.

Pepe, M. S. (2011). Problems with risk reclassification methods for evaluating prediction models. *American Journal of Epidemiology 173*(11), 1327–1335.

Prentice, R. (1986). A case-cohort design for epidemiologic cohort studies and disease prevention trials. *Biometrika 73*(1), 1–11.

Raghunathan, T. E., J. M. Lepkowski, and J. Van Hoewyk (2001). A multivariate technique for multiply imputing missing values using a sequence of regression models. *Survey Methodoly 27*(1), 85–95.

Rubin, D. B. (1987). *Multiple imputation for nonresponse in surveys*. Wiley: New York.

Sanders, M. (2009, October). Sparse multi-class prediction based on the group lasso in multinomial logistic regression. Master's thesis, Technical University of Delft.

Self, S. and R. Prentice (1988). Asymptotic distribution theory and efficiency results for case-cohort studies. *Annals of Statistics 16*(1), 64–81.

Tibshirani, R. (1996). Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society Series B-Methodological 58*(1), 267–288.

Tibshirani, R. (1997). The LASSO method for variable selection in the Cox model. *Statistics in Medicine 16*(4), 385–395.

Vaarhorst, A., Y. Lu, B. Heijmans, S. Böhringer, H. Putter, J. Boer, T. Gorgels, A. Merry, P. Van den Brandt, L. Schouten, E. Feskends, M. Muller, W. Jukema, and E. Slagboom (2011). A literature-based genetic risk score for predicting coronary heart disease; the carema case-cohort study. To appear.

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

Yuan, M. and Y. Lin (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society Series B-Statistical Methodology 86*(1), 49–67.