UiO : Department of Mathematics
University of Oslo

Dynamic survival prediction for high-dimensional data

Simon Boge Brant

Master's Thesis, Spring 2018

This master's thesis is submitted under the master's programme *Modelling and Data Analysis*, with programme option *Statistics and Data Analysis*, at the Department of Mathematics, University of Oslo. The scope of the thesis is 60 credits.

The front page depicts a section of the root system of the exceptional Lie group E_8, projected into the plane. Lie groups were invented by the Norwegian mathematician Sophus Lie (1842–1899) to express symmetries in differential equations and today they play a central role in various parts of mathematics.

Dynamic survival prediction for high-dimensional data

Simon Boge Brant

Abstract

In this thesis, we consider models for survival data with a high-dimensional covariate space. Most models used for such datasets are based on the Cox regression model, of which a critical assumption is that the hazard functions are proportional between individuals. The purpose of this thesis is to develop a way of analysing these datasets that does not require that the proportional hazards assumption is valid. In search of such a method, we study the concept of *landmarking* and try to develop a way of fitting what van Houwelingen and Putter [2011] refers to as *sliding landmark models* that works when we have a high number of covariates. An essential part of our strategy is the 'bet on sparsity principle' [Hastie et al., 2001], where one assumes that only some of the variables in the dataset have an effect on the outcome. We seek out to implement this using regularisation techniques, such as penalised regression and boosting. In particular, we develop a boosting algorithm for sliding landmark models, based on the *likelihood boosting algorithm* for Cox regression [Binder and Schumacher, 2008]. The thesis is concluded by a simulation study, where the different models and methods of estimation we consider are used to analyse different simulated datasets, and are compared via a dynamic Brier score [van Houwelingen and Putter, 2011].

Acknowledgements

I would first and foremost like to thank my supervisor, Ørnulf Borgan, for introducing me to an interesting topic, for suggesting a good structure on the process of writing, and for generously sharing his plentiful knowledge and experience. I would also like to thank my fellow students that I have shared discussions and late dinners on campus with, without whom I would not have had such a good time, or learned as much. In addition, I would like to thank Tristan and Vegard for spell-checking and comments. I would also like to thank my family, my mother in particular, for taking an interest in whatever it is that I do. Lastly, I want to thank Andrea for being patient and for always making me laugh.

Oslo, 2018
Simon Boge Brant

Contents

Chapter 1

Introduction and outline of the thesis

A lot of time and effort has been invested over the past couple of decades into trying to utilise genetic information to make predictions of survival. The hope of this is that we can use these very high quantities of information to make more accurate predictions than if we were merely using standard clinical variables such as the patients age, gender, biomarkers, et cetera. If one were able to create statistical models that incorporate genetic variables that yields more accurate predictions, then one could use these for a wide variety of applications. One could for instance give more accurate prognoses of, say 5 year survival, for a cancer patient. These models could potentially also be used to better understand, or perhaps discover, relationships between genetic variables and as it were, the risk of dying for an individual with a certain condition. To model survival data, the by far most widely used model is the *Cox regression model* [Cox, 1972]. Due to the omnipresence of the Cox model, many models that research statisticians attempt to develop to model survival data are based on the Cox model, and we too will here consider models that are extensions of the Cox regression model.

One particular problem that arises when working with statistical techniques for high dimensional data, is that which Bellman [1961] refers to as the curse of dimensionality, which essentially is that in a high dimensional space virtually any two points in a dataset will be very far apart. This makes generalisation from observed data hard, and using standard statistical techniques will in all likelihood lead to models that have little, if any predictive utility. In addition to this we also realistically need to be able

to fit models where the number of observations is outnumbered by the number of covariates. In these situations we cannot use the statisticians arsenal of maximum likelihood methods and least squares model fitting in the direct sense, but must adapt and extend them. All these possible extensions, at least the ones we are discussing here, can be referred to by the all-encompassing term *regularisation*. What we mean by this term is effectively either restricting the number of dimensions of the covariate space that the model makes use of, forcing the estimates to be closer to zero, or a combination of the two.

A group of such extensions is called penalisation, and involves adding a term called a *penalty function* to the objective function, i.e. the residual sum of squares, likelihood function, or in our case the partial log likelihood. Of these, we in particular discuss the *lasso* [Tibshirani, 1994], which involves subtracting a term proportional to the L_1 norm of the regression coefficients from the partial log likelihood. The effect of the lasso is that it restricts the absolute value of the estimates, or *shrinks* them. In addition, not all dimensions of the covariate space are guaranteed to be used, and we say that the model *selects* a subgroup of the effects. An alternative way of fitting models that can also cope with the same problems that penalised methods are designed to do, is the method of *boosting*. The idea of boosting is to estimate our model in an iterative fashion, by adding together small increments to the estimates for a given number of iterations. Boosting algorithms are usually designed to overcome the problem we discussed above concerning situations with more covariates than observations by letting the algorithm only update the coefficient of a single covariate in each iteration, and stopping the algorithm before it converges. We discuss two different boosting algorithms for the Cox model, which are called *model-based boosting* [Bühlmann and Hothorn, 2007] and *likelihood-based boosting* [Binder and Schumacher, 2008].

As mentioned most models used to model survival data are based on Cox regression. A key underlying assumption of the Cox model, is that the hazards are assumed to be proportional between individuals, or alternatively that the effects of the covariates are assumed to be constant in time. This assumption is not always valid, but there are ways to extend the Cox model that allows time varying effects. One such extension of the Cox regression model is known as *landmarking* [van Houwelingen and Putter, 2011], which in its essence involves considering Cox regression models that are local in time. One considers the sequence of these local models, where each individual model is connected to a subset of the follow-up range. This sequence is by van Houwelingen

and Putter [2011] termed a *sliding landmark model*. The purpose of landmarking is to create models that are better suited to make dynamic survival predictions than the Cox model when the proportional hazards assumption fails to hold. By dynamic predictions one here refers to continously making predictions of an individual surviving a given period ahead in time from a certain *landmark point*. I.e, we want to be able to predict for instance 5 year survival for a patient, not only at the time of diagnosis, but at several landmark points during the follow up of a patient.

The main goal of this thesis is to try and expand this method of landmarking to applications for high dimensional datasets, such as datasets where genetic variables are recorded. To this end we will consider two options, namely to extend it using a lasso-like approach, or by boosting. We want this method to behave in such a manner that it selects effects for all landmark points simultaneously. For the lasso based algorithm we will consider a combination of sliding landmarking with the *group lasso* [Yuan and Lin, 2006], treating the landmark effects for a covariate as different levels of a categorical covariate. Essentially we propose to extend the landmarking scheme by estimating the sliding landmark by subtracting a group lasso penalty from the likelihood function we get from adding the individual partial log-likelihoods from each local model in the sliding landmark model together. This aggregate of the individual likelihoods is by van Houwelingen and Putter [2011] referred to as the *integrated partial log likelihood*. For the boosting-based algorithm, we will consider a scheme based on likelihood-based boosting, where we extend this method to the integrated partial log likelihood.

In chapter 2, we will provide some background on survival data and the Cox regression model, and discuss sliding landmarking. In addition, we will also discuss a Brier score [Graf et al.] based method as a way of evaluating the predictive accuracy of landmark models. In chapter 3, we discuss penalised estimation of Cox regression models, and extensions of this approach to sliding landmark models. This is followed by a discussion of boosting in Cox regression models in chapter 4, where we also discuss boosting for sliding landmark models. The thesis is then concluded by a simulation study in chapter 5, and a summary in chapter 6.

Chapter 2

Survival analysis and non proportional hazards

2.1 Right-censored survival data

Time to event data can be encountered in many different fields of scientific study, such as medicine, biology, demography and sociology, to name a few. When the event in question can only happen once for each individual studied, such data are termed *survival data*. An overview of modeling of survival data, and methods concerning such models can for instance be found in Aalen et al. [2008]. Even though the term survival data is used, these data can be recorded times until any event, not just death. The problem with observing such data is that it takes time to observe time, and for various reasons such data will be incomplete. Imagine, for example, a clinical trial of some sort, where one records the time to some event for all the individuals in the study. Such studies usually span some fixed length of time, and all of the individuals will not always experience the event before the end of the study. In addition, there is always a possibility that some of the individuals will drop out before the end of the study, without experiencing the event. This is, of course, a problem that has to be dealt with when analysing such data. One could, somewhat naively, propose to ignore the incomplete data. But, only looking at the complete data will give a biased and less informed view of what we are studying. Survival data that are incomplete as described above are termed *censored survival data* [Aalen et al., 2008]. We can describe these data as two sets of random variables in the following way. For each

observation i, $i = 1, 2, ..., n$, there is one random variable T_i representing the i-th survival time, and one random variable C_i representing the i-th censoring time. The observed, possibly censored survival time is then $\tilde{T}_i = \min(T_i, C_i)$, along with an indicator $D_i = I(\tilde{T}_i = T_i)$ of an event taking place at time $\tilde{T}_i$.

When studying censored survival data we are interested in estimating the probability of the event happening later than a time t, $P(T > t)$. This is known as the *survival function*, and we write $S(t) = P(T > t)$. One may also define a similar *censoring function*, $C(t) = P(C > t)$, as it were, as a survival function for the censoring times. It is worth noting that we can write

$$1 - S(t) = P(T \leq t) = F(t),$$

where $F(t)$ is the cumulative distribution function. Another function that we would like to estimate is the *hazard function*, which, loosely speaking, is the probability of the event happening in a small interval $[t, t + \delta)$, given that the event has not happened before time t. More formally, the hazard function is defined as

$$\alpha(t) = \lim_{\delta \to 0} \frac{P(T < t + \delta | T \geq t)}{\delta}.$$

Estimation of hazard functions, as with densities, is in general quite hard. In particular, we cannot attain the usual $\sqrt{n}$ convergence rate that we get for estimates. By using that

$$P(T < t + \delta | T \geq t) P(T \geq t) = P(T \geq t) - P(T \geq t + \delta)$$

and recognizing the derivative of $S(t)$, one can make the observation that

$$\alpha(t) = \frac{-S'(t)}{S(t)}.$$

By noting that this is a separable differential equation in S and t, we get the relation

$$S(t) = e^{-A(t)},$$

where $A(t) = \int_0^t \alpha(s) ds$ is called the *cumulative hazard function*. Like the cumulative distribution function, both the survival and the cumulative hazard functions are much easier to estimate than densities and hazard functions. Traditional estimators of these are the Kaplan-Meier and the Nelson-Aalen estimators. Assuming a situation where

we have observed censored survival data (t_i, d_i), $i = 1, ..., n$, these estimators are defined as

$$\hat{S}(t) = \prod_{t_i \leq t} \left(1 - \frac{d_i}{Y(t_i)}\right),$$

and

$$\hat{A}(t) = \sum_{t_i \leq t} \frac{d_i}{Y(t_i)},$$

respectively. Here d_i is an indicator of the i-th recorded time being an event time and not a censored observation, and $Y(t)$ is the number at risk at time t. By the number at risk at time t, we mean the number of individuals that have not experienced the event, and have not been censored prior to time t.

In addition to the right censoring described above, where some of the individuals in the study either drop out, or never experience the event of interest, a further complication can be present. In some cases, not all the individuals under study enter the study at the same time, but enter the study at different times. This is known as *left truncation*. We will not deal with these type of data directly in this thesis, but we will in a central topic of this thesis pretend that all of the observations under consideration are left truncated at a specified time. This does not matter for estimation, as we can merely pretend that this point in time is 0, since the left truncation time is the same for all observations.

2.2 Cox regression

A Cox proportional hazards regression model [Aalen et al., 2008] is, as the name suggests, defined through the hazard function, which is required to be proportional between all individuals. This is done by assuming that the hazard consists of some arbitrary non-parametric function, usually referred to as the *baseline hazard*, multiplied by a constant that depends on a linear predictor for each individual. More concretely, the hazard of an individual i is expressed as

$$\alpha(t, \mathbf{x}_i) = \alpha_0(t) e^{\boldsymbol{\beta}^T \mathbf{x}_i},$$

where $\alpha_0(t)$ is the baseline hazard common to all individuals, $\mathbf{x}_i$ is the covariate vector for individual i, and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T$ are regression coefficients. As mentioned

above we do assume that all individuals share a common baseline hazard $\alpha_0(t)$, but we do not make any assumptions about its shape. I.e, the Cox proportional hazards model is a semi-parametric model. Due to the semiparametric nature of the hazard specification in the Cox regression model, it is impossible to use ordinary likelihood methods. Instead one has to resort to a partial likelihood for estimation and inference. The partial likelihood for such a model is

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} \left(\frac{e^{\boldsymbol{\beta}^T \mathbf{x}_i}}{\sum_{\ell \in \mathcal{R}_i} e^{\boldsymbol{\beta}^T \mathbf{x}_\ell}} \right)^{d_i},$$

where $\mathcal{R}_j$ is the risk set at time t_j. Underlying the Cox model are the two main assumptions of log-linearity

$$\log(\alpha(t|\mathbf{x})) = \log(\alpha_0(t)) + \boldsymbol{\beta}^T \mathbf{x},$$

and that the hazards are proportional independently of time

$$\frac{\alpha(t|\mathbf{x}_1)}{\alpha(t|\mathbf{x}_2)} = e^{\boldsymbol{\beta}^T(\mathbf{x}_1 - \mathbf{x}_2)}.$$

One way to check if the first assumption holds for a given covariate x_j is to fit a Cox model where the hazard takes the form $\alpha(t|x_j) = \alpha_0(t)e^{f(x_j)}$, where $f(x)$ is some regression function that is estimated by a spline, and then plot $f(x_j)$ against x_j. To check the second assumption, one option is to plot and perform tests based on the Schoenfeld residuals [Grambsch and Therneau, 1994].

2.2.1 Cox regression on Danish melanoma data

One example that illustrates that these assumptions do not need to be fulfilled, is a dataset of Danish cancer patients with malignant melanoma operated at the Odense University hospital in the period of 1962-1977, which can be found in Andersen et al. [1993]. In this dataset the patients' tumor thickness, age, and sex were recorded, along with an indicator of a patient having ulceration.

Table 2.1: Ordinary Cox-regression for melanoma data.

	$\hat{\beta}$	$e^{\hat{\beta}}$	$\hat{\sigma}_{\hat{\beta}}$	Z	P
ulcer	0.971	2.641	0.321	3.027	0.002
log_2(tumor thickness)	0.423	1.527	0.122	3.470	0.001

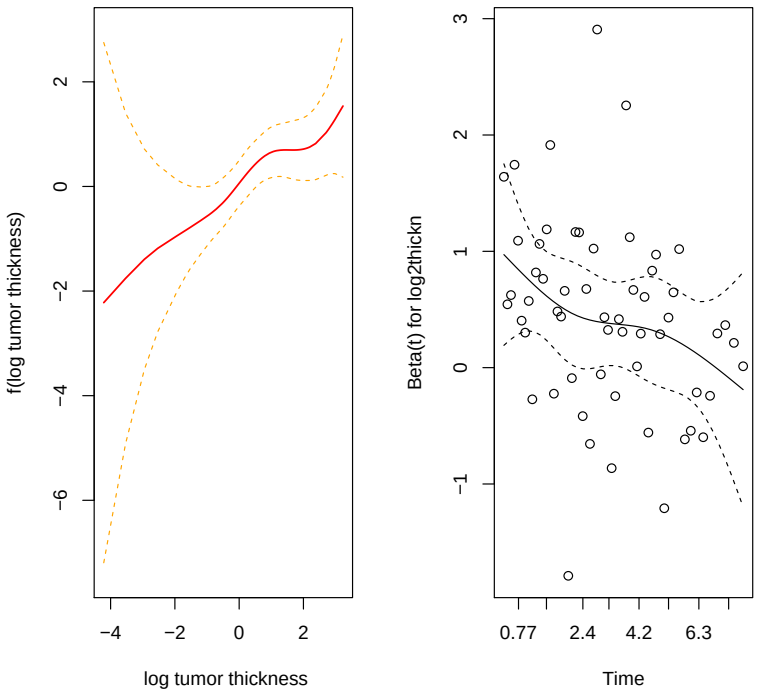

Figure 2.1: Spline fit of the effect of log_2-tumor thickness, and Scoenfeld residuals for the Danish melanoma patients.

A summary of a regular Cox regression model fitted with log_2 of the tumor thickness and ulceration as covariates is given in table 2.1. The fit indicates – given that the model is correct – that the effect of doubling the tumor thickness is an increase in relative risk of 52.7%, and that the effect of a patient having ulceration corresponds to an increase in relative risk of 164.1%. By plotting spline fits (figure 2.1) one can see that the assumption of log-linear effects seems to hold for the log-transformed thickness, except for smaller values of tumor thickness. The results from a formal test based on the Schoenfeld residuals from a fitted Cox model with the log-transformed thickness, and the plot given in figure 2.1 indicates a deviation from the proportional hazards assumption. A problem that arises, is that we do not know what is estimated when the assumption of proportional hazards is violated. In short, it turns out that the Cox-regression estimate is a kind of average of the time-varying effect over the entire study. We will return to this problem with a more in-depth answer in section 2.3.1.

2.2.2 Estimation of cumulative hazards and survival probabilities

For various reasons, we may be interested in the estimated cumulative hazard under the assumptions of the Cox proportional hazards model for a given covariate vector. One approach here is to use the estimator

$$\hat{A}(t|\mathbf{x}) = \hat{A}_0(t)e^{\hat{\boldsymbol{\beta}}^T\mathbf{x}},$$

where $\hat{A}_0(t)$ is the Breslow estimator

$$\hat{A}_0(t) = \sum_{t_i \leq t} \frac{d_i}{\sum_{\ell \in \mathcal{R}_i} \exp\left(\hat{\boldsymbol{\beta}}^T \mathbf{x}_\ell\right)}. \tag{2.1}$$

We can also obtain an estimator of the survival function by transforming the cumulative hazard estimator, i.e

$$\hat{S}(t|\mathbf{x}) = e^{-\hat{A}(t|\mathbf{x})}. \tag{2.2}$$

An application of this is to use the estimated survival function in order to calculate estimates of survival probabilities. For example one can answer questions such as what is the probability of the event not occuring before time t, given that it has not yet occured by time $s \leq t$, by estimating

$$P(T > t|T \geq s) = \frac{S(t|\mathbf{x})}{S(s|\mathbf{x})},\ s \leq t$$

using $\hat{S}(t|\mathbf{x})$.

2.2.3 Survival probabilities for the melanoma data

Using the fitted Cox model from section 2.2.1 together with the Breslow estimator, we can compute estimated 5-year survival probabilities for a given individual as described in section 2.2.2. As an illustration we compute these for an individual with an average tumor thickness, without ulceration. A plot of these probabilities is given in figure 2.2. A simple interpretation of the plot is that the 5-year survival prognoses are increasingly becoming better from around two years after the diagnosis, meaning that a patient is more likely to recover from the disease if he or she survives the first two years after being diagnosed.

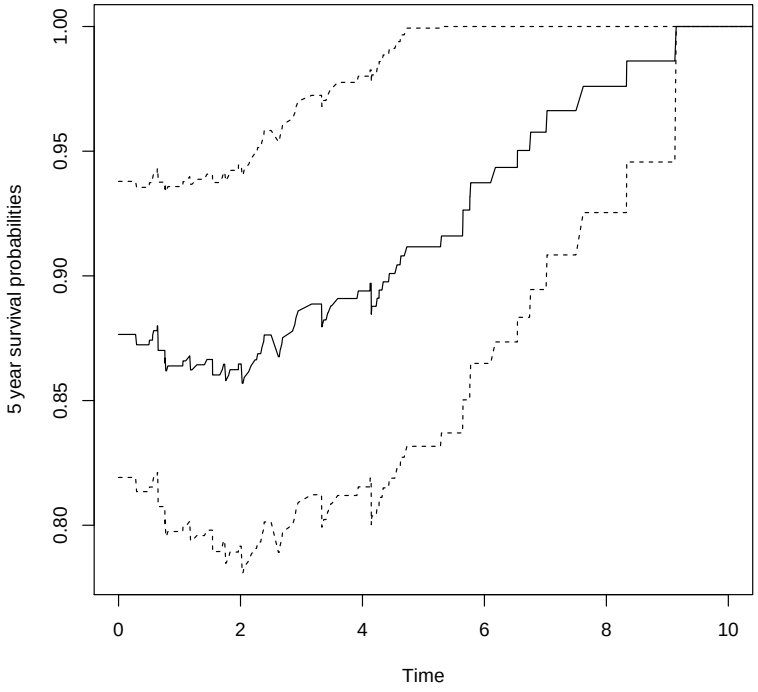

Figure 2.2: 5-year survival probabilities from regular Cox fit, for an individual with an average tumor thickness and no ulceration. The dotted lines represent 95% confidence intervals.

2.3 Landmarking

In this section, we will first consider some robustness properties of Cox regression in a misspecification context. We will assume that there is a time-varying effect of the covariates on the true hazard, i.e. the hazard is of the form

$$\alpha_i(t) = \alpha_0(t)e^{\boldsymbol{\beta}(t)^T\mathbf{x}_i},$$

and consider what happens if we fit a Cox proportional hazards regression model. We will see that if we fit a Cox-model where we treat all observations with a (censored or uncensored) survival time that exceeds some t_{hor} as censored, the coefficient estimates we obtain are a form of averages of the time-varying effects over the interval $[0, t_{hor}]$. Such a scheme, where one treats all observations with a survival time that exceeds some value t as censored is called *administrative censoring*. In the same context, we will also

observe that the Breslow-type estimate of the cumulative hazard is approximately equal to the true cumulative hazard at t_{hor}, given some conditions. In short, these conditions state that the *prognostic index* $\boldsymbol{\beta}^T \mathbf{x}$ should be small, and not vary too much. If we center the covariates we 'move' some of the prognostic index from the relative risk function $e^{\boldsymbol{\beta}(t)\mathbf{x}}$ to the baseline hazard $\alpha_0(t)$ in the Cox regression model, thus the condition that the prognostic index should be small necessitates centering the covariates. When we center the covariates, the hazard instead takes on the form

$$\alpha_i(t) = \alpha_0^*(t) e^{\boldsymbol{\beta}(t)\mathbf{x}_i^*},$$

where $\mathbf{x}_i^* = \mathbf{x}_i - \overline{\mathbf{x}}$, $\alpha_0^*(t) = \alpha_0(t) e^{\boldsymbol{\beta}(t)\overline{\mathbf{x}}}$, and $\overline{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^n \mathbf{x}_j$. We can then relabel $\mathbf{x}_i^*$ as $\mathbf{x}_i$, and $\alpha_0^*(t)$ as $\alpha_0(t)$. This theoretical investigation is a motivation for a way of dealing with time-varying effects when trying to make dynamic survival predictions. In short, this methodology involves fitting Cox regression models on different 'time windows' $[LM_s, LM_s + w]$ for S time-points LM_s and a fixed window size w with left-truncation at LM_s and administrative censoring at $LM_s + w$. Here, one exploits the fact that the estimated cumulative hazard under the Cox model will be close to the true model on each of these subintervals of the total follow up time, which will yield good dynamic predictions of survival probabilities even though there are time-varying effects.

2.3.1 Robustness of Cox-regression

We will go over some theoretical results that are the underpinnings of the landmarking technique for computing dynamic survival predictions in settings with time-varying effects. These results are taken from van Houwelingen [2007], and are here merely stated without justification. Detailed derivations, that are a somewhat embellished version of the appendix of van Houwelingen [2007], can be found in appendix B. Suppose we are in the right-censored survival data situation, described in section 2.1, and that the censoring and survival times are independent given the covariates. Such a censoring mechanism is called *random censoring*. In addition, assume that the covariates $\mathbf{x}_i$ are centered and that the individual hazards take the form

$$\alpha_i(t) = \alpha_0(t) e^{\boldsymbol{\beta}^T(t)\mathbf{x}_i},$$

i.e. there is a time-dependent effect of the covariates. In this case, one can show that given some regularity conditions, the most important being that the covariates are

centered,

$$A(t|\mathbf{x}_i) \approx A_0(t)e^{\overline{\boldsymbol{\beta}}^T(t)\mathbf{x}_i}, \tag{2.3}$$

where

$$\overline{\boldsymbol{\beta}}(t) = \frac{\int_0^t \alpha_0(s)\boldsymbol{\beta}(s)ds}{A_0(t)}. \tag{2.4}$$

One may also show that, provided some regularity conditions are satisfied, if one fits a Cox regression model with administrative censoring at some time t_{hor}, the estimates $\tilde{\boldsymbol{\beta}}_{Cox}$ are approximately given by

$$\tilde{\boldsymbol{\beta}}_{Cox} \approx \overline{\boldsymbol{\beta}}(t_{hor}). \tag{2.5}$$

This leads to, after quite some work, an approximation which is essential to the topic of this thesis, namely that

$$A_{Cox}(t_{hor}|\mathbf{x}) \approx A(t_{hor}|\mathbf{x}), \tag{2.6}$$

if the covariates are centered, the coefficients do not vary to much over $[0, t_{hor}]$, this interval is not too wide, and we have random censoring. This means that if we use the estimator (2.2), provided that we consider a small enough time window, we should obtain approximately correct predictions of surviving up to time t_{hor} even though there might truly be a time-dependent effect of the covariates.

2.3.2 Sliding landmarking

The estimates from a Cox model might give a reasonable prediction of survival time up to some t_{hor}, even if the assumption of proportional hazards fail. However, in this case the Cox model might not be a good choice when it comes to making dynamic predictions. I.e, the estimates obtained by the method presented in section 2.2.2 might be inaccurate, as the Cox model does not capture dynamic differences. Instead we may assume that we are in the misspecification situation presented in subsection 2.3.1, and rather use weighted averages of $\boldsymbol{\beta}(t)$ computed over the intervals $[LM_s, LM_s + w]$, $0 = LM_1 < LM_2 < \cdots < LM_S = t_{hor}$ instead of the average over the whole follow-up range $[0, t_{hor}]$. van Houwelingen and Putter [2011] call this a sliding landmark

model. To make predictions from some time-point $t = LM$ to $t = LM + w$, one fits a Cox model to a data set that is left truncated at $t = LM$ with administrative right censoring at $t = LM + w$. The sliding landmark model can be written as

$$\alpha(t|\mathbf{x}, LM, w) = \alpha_0(t|LM, w) \exp(\boldsymbol{\beta}^T(t)\mathbf{x}).$$

The landmark points LM_s can, for example, be chosen as a grid of equidistant points over a desired interval $[0, t_{end}]$. Alternative choices are possible but should not depend on the actual event times. van Houwelingen and Putter [2011] suggest that a grid of between 20 and 100 points should be sufficient. As outlined above, one estimates the model by fitting a sequence of Cox regression models, one for each landmark point LM_s on the grid, where all observations are left truncated at the landmark point LM_s, and right censored at $LM_s + w$. Using the notation

$$A_s = \{i|t_i \in [LM_s, LM_s + w]\},$$

we can write the individual partial log likelihood for the s-th landmark point as

$$\sum_{i \in A_s} \boldsymbol{\beta}^T(LM_s)\mathbf{x}_i - \log\left(\sum_{\ell \in \mathcal{R}_i} \exp\left(\boldsymbol{\beta}^T(LM_s)\mathbf{x}_\ell\right)\right).$$

Since all these S partial log-likelihoods depend on different sets of regression coefficients, $\boldsymbol{\beta}(LM_s)$, if we maximise them independently this is equivalent to maximising them all at once. Therefore, using the notation

$$l_i(\boldsymbol{\beta}(LM_s)) = \boldsymbol{\beta}^T(LM_s)\mathbf{x}_i - \log\left(\sum_{\ell \in \mathcal{R}_i} \exp\left(\boldsymbol{\beta}^T(LM_s)\mathbf{x}_\ell\right)\right) \tag{2.7}$$

we can write the likelihood we are maximising as

$$ipl(\boldsymbol{\beta}(\mathbf{LM})) = \sum_{s=1}^{S} \sum_{i \in A_s} l_i(\boldsymbol{\beta}(LM_s)). \tag{2.8}$$

van Houwelingen and Putter [2011] calls this expression an *integrated partial log-likelihood*, although they define it in seemingly different, but equivalent terms. We shall from here on occasionally refer to this expression by the name *integrated partial log-likelihood*, or by the abbreviation *ipl*.

2.3.3 Survival predictions from sliding landmark models

As discussed in the previous section, the motivation for the sliding landmark model is rooted in the problem of giving dynamic survival predictions for a given individual. By this we mean predicting the probability of an individual surviving, say 5 years from some point in time given that the individual has survived up to then. We do this in the natural way, keeping the approximation (2.6) in mind, by computing the conditional baseline cumulative hazard from LM_s to $LM_s + w$ as

$$\hat{A}_0(LM_s + w|LM_s) = \sum_{i \in A_s} \frac{d_i}{\sum_{\ell \in \mathcal{R}_i} \exp\left(\boldsymbol{\beta}(LM_s)^T \mathbf{x}_\ell\right)}.$$

The estimate of the corresponding conditional baseline hazard estimate for an individual with covariate vector $\mathbf{x}_0$ is then

$$\hat{A}(LM_s + w|LM_s, \mathbf{x}_0) = A_0(LM_s + w|LM_s) \exp(\boldsymbol{\beta}(LM_s)^T \mathbf{x}_0).$$

Therefore the estimate of the conditional survival function used to predict survival for an individual with covariate vector $\mathbf{x}_0$ is

$$\hat{S}(LM_s + w|LM_s, \mathbf{x}_0) = \exp(-A(LM_s + w|LM_s, \mathbf{x}_0)).$$

2.3.4 Sliding landmarking analysis of Danish melanoma data

An illustration of the sliding landmarking approach to computing regression coefficients can be given using the data from the Danish melanoma study. A grid of 76 equidistant points over the interval $[0, 7.5]$ were chosen as landmarks. Regression coefficients were computed for log-transformed tumor thickness and ulceration at each landmark point LM_s using the corresponding landmark datasets with a window size of $w = 5$, i.e. with left truncation at LM_s and administrative censoring at $LM_s + 5$. A plot of the landmark estimates of the regression coefficients is given in figure 2.3. From the plot, it seems like the effect of increasing log-transformed tumor thickness is decreasing on the interval $[0, 6]$, and increasing on the interval $[6, 7.5]$. It also seems like the effect of ulceration being present is somewhat decreasing on the interval $[0, 4]$, and increasing on $[4, 7.5]$. An application of these fitted coefficient curves is to produce estimates of 5-year survival probabilities as described in the previous section. A plot of these computed for each landmark point using both the landmark coefficients and

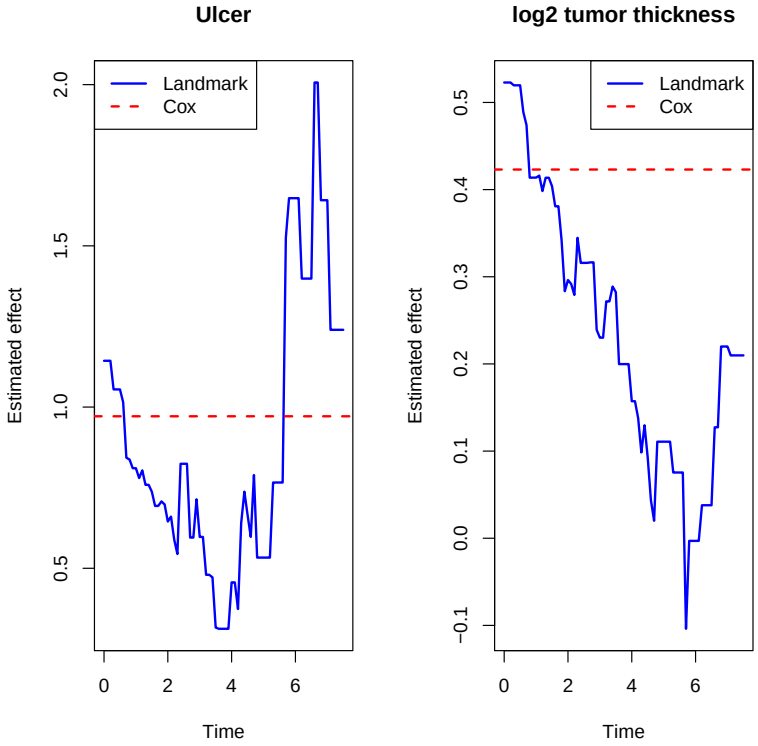

Figure 2.3: Sliding landmark estimate of the coefficients for log-transformed thickness and ulceration. The dotted lines indicate the values of the Cox estimates over the entire study.

the Cox estimates, for both an individual without ulceration with average log2 tumor thickness (0.89), and an individual with ulceration and log2 tumor thickness equal to one is given in figure 2.4. Judging by the plot, the Cox model overestimates the 5-year survival probabilities compared to the sliding landmark model for the individual without ulceration and with an average log2 tumor thickness, and underestimates for the individual with ulceration and log2 tumor thickness equal to 1.

2.4 Assesment of survival predictions

A major ambition of this thesis is to develop a new method for making survival predictions that we hope is well suited for situations where we have a high dimensional covariate space, and non-proportional hazards. Specifically, we wish to investigate if

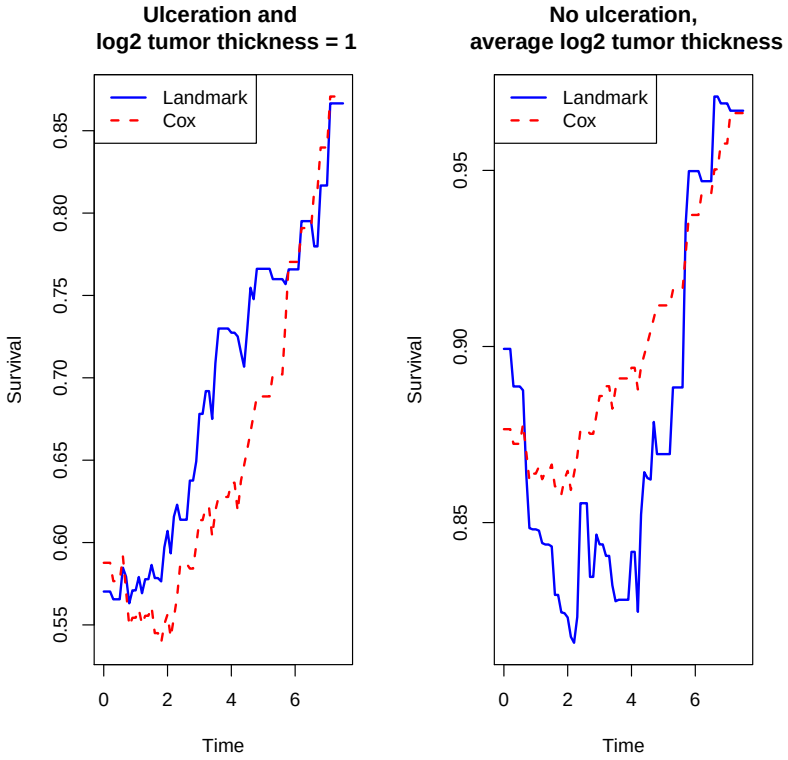

Figure 2.4: 5 year survival probability predictions computed from the sliding landmark model and from the Cox-model, for an individual with ulceration and log2 tumor thickness equal to 1, and an individual without ulceration and an average log2 tumor thickness.

our method does any better than standard approaches based on methods that implicitly assume proportional hazards such, as penalised Cox regression methods which we will discuss later in this thesis. To be able to answer such questions, we need a conceptual contraption that allows for such comparisons, which is not as straightforward as in other situations as for example binary classification problems. There are a number of available methods developed specifically for the assessment of survival predictions, of which a number are summarised and discussed in the paper by Bøvelstad and Borgan [2011].

Of the methods discussed in this paper we will here focus on the Brier score and a related R^2 measure based on it, and apply this to the evaluation of dynamic survival predictions based on landmarking. The reason we choose the Brier score approach, is

mainly that the Brier scores are not specific to the Cox likelihood. By R^2 measure, one means a score between 0 and 1 that says something of the proportion of variation in the dependent variable, i.e the survival time, that is predictable from the independent variables. In other words, the closer the R^2 measure for a given method is to 1, the better the predictive capabilities of the model. It is important to note that when we are to evaluate how a well a method of prediction works, we need to 'hide' some of the data from the from the estimation procedure. If the data we use to evaluate the goodness of the models predictions also are used to estimate the parameters of the model, then we may grossly overestimate its predictive power. This includes the tuning of what are sometimes referred to as hyperparameters, a topic we shall return to when we discuss penalised regression methods in the next chapter. The subset of the data that we omit when estimating the regression coefficients is often referred to as a *test set*, and the remaining data that is used in the estimation procedure is often referred to as a *training set*.

2.4.1 Brier scores for survival data

The Brier score was first introduced in Brier [1950] as a method of measuring the inaccuracy of probabilistic weather forecasts. In the paper by Graf et al., the Brier score was adapted as a measure for evaluating the accuracy of survival predictions, and an R^2 measure based on the Brier score was developed. The Brier score is purpose specific of predicting survival up to some time t^*. In a situation where there are no censorings in the test set, the Brier score of predicting survival up to t^* is defined as

$$\begin{aligned} BS(t^*) &= \frac{1}{m}\sum_{j=1}^{m}\left(I(t_{i_j} > t^*) - \hat{\pi}(t^*|\mathbf{x}_{i_j})\right)^2 \\ &= \frac{1}{m}\sum_{j=1}^{m}\left[\hat{\pi}(t^*|\mathbf{x}_{i_j})^2 I(t_{i_j} \leq t^*) + (1-\hat{\pi}(t^*|\mathbf{x}_{i_j}))^2 I(t_{i_j} > t^*)\right] \end{aligned}$$

where m is the number of individuals in the test set, i_j, $j = 1, 2, ..m$ are the indices of the individuals in the test set, and $\hat{\pi}(t^*|\mathbf{x}_{i_j})$ is the estimated probability of the j-th individual in the test set surviving up to t^*. This probability can be anything, but it could here be natural to imagine that it is estimated by (2.2). Looking at the expression for the Brier score with no censorings, we can see that the idea of the Brier score is to dichotomise the survival times by looking at the variable $I(t_i > t^*)$, and then

computing a mean squared measure from the prediction of the survival time exceeding t^*. In the event that not all the individuals in the test set experience the event and that the censoring and survival times are independent given the covariates, Graf et al. propose that the Brier score is expressed as

$$BS^c(t^*) = \frac{1}{m}\sum_{j=1}^{m}\left[\frac{\hat{\pi}(t^*|\mathbf{x}_{i_j})^2 I(t_{i_j} \leq t^*, d_{i_j} = 1)}{\hat{G}(t_{i_j})} + \frac{(1-\hat{\pi}(t^*|\mathbf{x}_{i_j}))^2 I(t_{i_j} > t^*)}{\hat{G}(t^*)}\right],$$

where

$$\hat{G}(t) = \prod_{t_{i_j} \leq t}\left(1 - \frac{1-d_{i_j}}{\sum_{k=1}^{m} Y_{i_k}(t_{i_j})}\right)$$

is the Kaplan-meier estimate of the censoring function for the individuals in the test set, where $Y_i(t)$ is an indicator of the i-th individual being in the risk set at time t. This score may be used to define an R^2 measure, benchmarking the performance of a model against the null model, in the following way

$$R^2_{Brier}(t^*) = 1 - \frac{BS^c(t^*)}{BS^c_0(t^*)}.$$

By BS^c_0 we mean the expression for BS^c where the predictions are made with a model where all the regression coefficients are set to zero. Graf et al., and Bøvelstad and Borgan [2011] extend these concepts of the Brier score, and the related R^2 measure to integrated versions. That is, instead of evaluating them at some single time-point t^*, they are evaluated on a grid, and averaged in a suitable fashion. We will see in the next section that these extensions are not so relevant for our application, and we will therefore not go through them in detail.

2.4.2 A dynamic Brier score approach to assesing landmark predictions

The Brier score and the related R^2 measure attempts to estimate the goodness of a set of survival predictions at some time t^*, or over a time-period via their integrated versions. The goal is to assess the predictive performance of a model, specifically of predictions concerning survival from time $t = 0$ to time $t = t^*$. In this thesis however, we are studying landmarking, which is motivated by a desire to produce good dynamic survival predictions. In other terms we are working with estimates of a conditional

survival function $P(T > t|T \geq s)$, and not with estimates of a survival function $S(T > t)$. To be even more specific, we are estimating a sequence of S models connected to different points in time (landmarks) LM_s, where the purpose of each model is to predict survival w ahead in time. I.e., we are trying to give as good as possible an estimate of $P(T > LM_s + w|T \geq LM_s)$, where these are achieved with models built for each purpose, each model having its own set of coefficients, and separate baseline hazard. To evaluate such models that give dynamic predictions, we need a dynamic measure of prediction error. Such a measure is provided by van Houwelingen and Putter [2011], and can, in the notational style of this thesis, be expressed as

$$DBS^c(t_0, t^*) = \frac{1}{Y_{test}(t_0)} \sum_{j \in \mathcal{R}_{test}(t_0)} \frac{\hat{\pi}(t^*|\mathbf{x}_{i_j}, t_0)^2 I(t_{i_j} \leq t^*, d_{i_j} = 1)}{\hat{G}(t_{i_j}|t_0)} + \frac{(1 - \hat{\pi}(t^*|\mathbf{x}_{i_j}, t_0))^2 I(t_{i_j} > t^*)}{\hat{G}(t^*|t_0)},$$

where t_0 is the time-point we are predicting the survival up to t^* from, $\pi(t^*|\mathbf{x}_{i_j}, t_0)$ is the estimate of the survival up to t^*, conditional of survival up to t_0, and $\hat{G}(t^*|t_0)$ is the Kaplan-Meier based estimate of the conditional censoring function, $\mathcal{R}_{test}(t_0)$ is the set of indices belonging to the individuals in the test set at time t_0, and $Y_{test}(t_0)$ is the number of individuals in the test set at risk at time t_0. It is worth noting that van Houwelingen and Putter [2011] propose that the weights here represented by the Kaplan-Meier based estimates $\hat{G}$ are instead estimated in an analogue fashion to (2.2), but we will here stick to the Kaplan-Meier, in keeping with Graf et al. and Bøvelstad and Borgan [2011]. This conditional version of the censoring function is computed as

$$\hat{G}(t|s) = \frac{G(t)}{G(s)} = \prod_{t_{i_j} \in (s,t]} \left(1 - \frac{1 - d_{i_j}}{\sum_{k=1}^{m} Y_{i_k}(t_{i_j})}\right).$$

As mentioned, we are interested in the dynamic preditions w ahead in time at each landmark point LM_s, therefore the scores we will use can be written as

$$DBS^c(LM_s, LM_s + w) = \frac{1}{Y_{test}(LM_s)} \times \sum_{j \in \mathcal{R}_{test}(LM_s)} \frac{\hat{\pi}(LM_s + w|\mathbf{x}_{i_j}, LM_s)^2 I(t_{i_j} \leq LM_s + w, d_{i_j} = 1)}{\hat{G}(t_{i_j}|LM_s)} + \frac{(1 - \hat{\pi}(LM_s + w|\mathbf{x}_{i_j}, LM_s))^2 I(t_{i_j} > LM_s + w)}{\hat{G}(LM_s + w|LM_s)},$$

which will yield a vector of S scores when evaluated for each landmark point. These can, in a similar manner as before, be used to define a dynamic R^2 measure for our landmarking predictions as

$$DR^2_{Brier}(LM_s + w, LM_s) = 1 - \frac{DBS^c(LM_s, LM_s + w)}{DBS^c_0(LM_s, LM_s + w)},$$

which will also yield a vector of S scores, one for each landmark point.

2.4.3 Comparison of the predictive performance of landmarking and Cox regression on Danish melanoma data

To measure ability of the landmarking model to predict the survival of melanoma patients, we will use the strategy outlined above. Specifically, we will compute dynamic R^2, or DR^2_{Brier}, for 5-year predictions for each method and plot these to compare them. To do this we have to split the data in two parts, into so-called test and training sets. We let the test set consist of approximately a third of the data, and the training set of the remaining data. Both a Cox regression model, and a sliding landmarking model with the same landmarks and interval width as in section 2.3.4 are fitted to the training data. The resulting two models are then used to predict 5 year survival times for the data in the test set. From these, the dynamic Brier scores, and the dynamic R^2 measures are computed as described in the previous section. These are rendered in two separate plots, which are shown in figure 2.5. Since the Brier score is like a mean squared measure, the lower the Brier score, the better the model is at predicting 5 year survival.

As for other R^2 measures, the closer the measure is to 1 the better. Keeping this in mind when we expect the plots in figure 2.5, we see that the Cox and the landmark model in this case are quite similar in performace, up to about 3 years. The landmark model seems to do better than the Cox model, apart from the last time period from about 7 years. Looking at the dynamic R^2, we see that in the interval $[3, 6]$, both the models seem to make worse predictions than if we were to use the null-model, or the model with no effect of the covariates. The performance is here much worse than the null model for the Cox model, while the landmark model seems to vary around the point of being equally bad as the null model in terms of predictive performance. The landmark approach may be better than the Cox model at making dynamic predictions in this setting, but will certainly suffer from the same issues as the Cox model, and

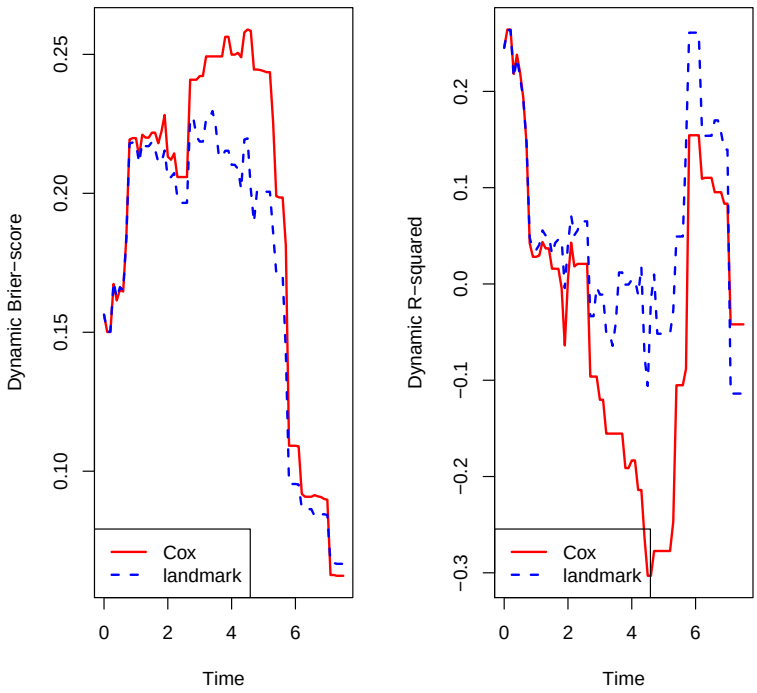

Figure 2.5: Plot of the dynamic Brier score and the dynamic R^2 measure for the Danish melanoma data, computed for the Cox model and the method of sliding landmarking.

other maximum-likelihood related models in higher dimensions. This neccesitates regularisation, or imposing restrictions on the optimisation problem. Regularisation could also yield better predictions in lower dimensions, and much of what we will focus on for the remaining part of this thesis will revolve around such techniques.

Chapter 3

Penalised regression in survival models

For the most part of the history of statistics, the main focus has been on maximum likelihood estimation and least squares model fitting. In this setting, we need to have more observations than the number of parameters in the model to even be able to estimate the model parameters. Usually, this is not a problem. There are however situations where we have more variables than observations, one notable example being medical studies with recorded gene expression data for each individual. In these types of situations, one can easily have several thousand parameters to estimate, and only a few hundred observations or less. Here, most methods of classical statistics are unusable. It may also occur that we have many parameters to estimate in relation to the number of observations, but still more observations than parameters. Here, one can technically fit regression models with traditional techniques, but the estimates are usually too variable to be useful. The problem here may be thought of as that too much of what the model explains is random variation, rather than an actual relationship between the covariates and the dependent variable. We will here focus on a solution to this problem called penalised likelihood estimation, which essentially involves subtracting a function of the parameters, called a penalty function, from the log likelihood.

3.1 Penalisation

Before we get into technical points about penalisation, we would like to take a moment and address the question of which problems we are trying to solve. Firstly, there are numerical issues that can occur as described above. Secondly, if we fit a model with many covariates, we might think that the model is too complex, and therefore is lacking in interpretability. One popular way to deal with these problems, and in particular the second problem, is subset selection. This involves fitting models where one includes and exclude different combinations of the covariates, and picks the 'best' model in terms of some selection criteria such as AIC or BIC [Claeskens and Hjort, 2008]. There are several issues with this. It typically involves fitting many models, which quickly becomes computationally infeasible when the number of covariates is large [Hastie et al., 2001].

Although the latter can be remedied by using greedy algorithms, there is an additional problem in that since the model selection process is discrete, there is very high variability in the resulting models. I.e., small changes in the dataset can lead to a very different model [Breiman, 1996]. By greedy algorithms one means algorithms for solving optimization problems that follow a heuristic of sequentially making locally optimal choices, in the hope of finding a global optimum. For the subset selection problem this can for instance mean starting from a model with no covariates, and including one covariate at a time until no further improvement can be made in terms of the selection criteria, instead of selecting the optimal solution among all possible covariate combinations.

Returning to penalisation, an alternative approach is to fit the model with a *lasso* penalty, introduced by Tibshirani [1994] The idea here is to introduce a penalty term to the objective function to shrink the estimates. The penalty term here is the L_1 norm of the parameter vector. This might help against overfitting. In addition, it often tends to set some of the coefficients to zero, thus also serving as a model selection procedure. More specifically, we obtain the regression coefficients by solving

$$\underset{\boldsymbol{\beta}}{\operatorname{argmax}} \left\{ l(\boldsymbol{\beta}) - \lambda \sum_{j=1}^{p} |\beta_j| \right\},$$

where $l(\beta)$ is the log likelihood of the model, and λ is a parameter that determines how much the parameters should be penalised. This form of the maximization problem is

known as the *Lagrangian form*, and the problem may also be stated in the equivalent way of

$$\underset{\boldsymbol{\beta}}{\text{argmax}}\ l(\boldsymbol{\beta}),\ \text{subject to}\ \sum_{j=1}^{p} |\beta_j| \leq s,$$

where s is a parameter that determines the level of constraint on the coefficients. An alternative to the lasso is the method of ridge regression [Hoerl and Kennard, 1970], which penalises the likelihood by the squared L_2 norm of the coefficient vector. In the Lagrangian form, the ridge problem may be stated as

$$\underset{\boldsymbol{\beta}}{\text{argmax}} \left\{ l(\boldsymbol{\beta}) - \lambda \sum_{j=1}^{p} \beta_j^2 \right\}.$$

Ridge regression only shrinks the coefficients, and does not offer model selection directly like the lasso does.

3.1.1 Cross validation, and how to choose λ

To fit the model, one must choose a specific value for λ. This will typically be done by setting a grid of values, and choosing the optimal by a technique called *cross validation* (CV). Cross validation had previously existed as an idea in statistics, but was first formalised by Stone [1974] and Geisser [1975]. The procedure, now usually termed ordinary cross validation(OCV) — as opposed to generalised cross validation(GCV) [Craven and Wahba, 1978] — involves dividing the dataset into K different subsets of roughly the same size, successively fitting to $K-1$ of the subsets, and computing some measure of fit on the K-th subset. The sum, or average of these is then used to evaluate the model. In the penalised likelihood setting, this may be phrased as

$$CV(\lambda) = \sum_{k=1}^{K} l_k(\hat{\boldsymbol{\beta}}_{(-k)}(\lambda)),$$

where $l_k(\boldsymbol{\beta}_{(-k)}(\lambda))$ is the penalised log likelihood evaluated for the k-th subset, using parameter estimates computed from the full dataset excluding the k-th subset. When $K < n$ (typically 5 or 10), this is referred to as K-fold cross validation. The situation where $K = n$ is referred to as *leave one out* - cross validation (LOO).

3.1.2 Penalisation and the Cox regression model

In this thesis, we concern ourselves with survival data, the Cox regression model, and extensions thereof. Due to the fact that the terms in the Cox partial log likelihood are not independent, we have to adapt the likelihood measure of fit in order to use ordinary cross validation to evaluate Cox regression models. Verweij and Van Houwelingen [1993] provided an extension of the cross validation methodology to the survival setting, and specifically the Cox regression model, which in their later paper [Verweij and Van Houwelingen, 1994] was applied to penalised likelihood in Cox regression, with an L_2 penalty term. In Verweij and Van Houwelingen [1993], they concentrate on an expression for the likelihood for leave one out cross validation. K-fold cross validation for the Cox model has been discussed by Bøvelstad et al. [2007]. It is possible here to find an exact expression for the cross validated partial log likelihood as in Verweij and Van Houwelingen [1993], but in practice what one computes is just

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

where $l_{(-k)}$ is the Cox partial log likelihood where the k-th fold is excluded. One should note that there does not exist a valid formula for the standard errors of the regression coefficients from this model. This is due to the fact that the terms in the partial log likelihood are dependent, and that we have to define the cross validated partial log likelihood in the manner stated above. While first developed for the linear model, Tibshirani [1997] provided an extension to the Cox regression model in the right censored survival data setting. The problem is stated as

$$\underset{\boldsymbol{\beta}}{\text{argmax}}\ l(\boldsymbol{\beta}),\ \text{subject to} \sum_{j=1}^{p} |\beta_j| \leq s,$$

where $l(\boldsymbol{\beta})$ is the Cox log partial likelihood. In the original paper by Tibshirani [1997] the covariates are assumed to be standardised such that $\sum_{i=1}^{n} x_{ij} = 0$, and $\frac{1}{n}\sum_{i=1}^{n} x_{ij}^2 = 1$. We may however not always require this, but instead relax the assumption to that the measurements have to be on the same scale. The ridge method may also be adapted to the Cox regression model in a similar fashion as the lasso.

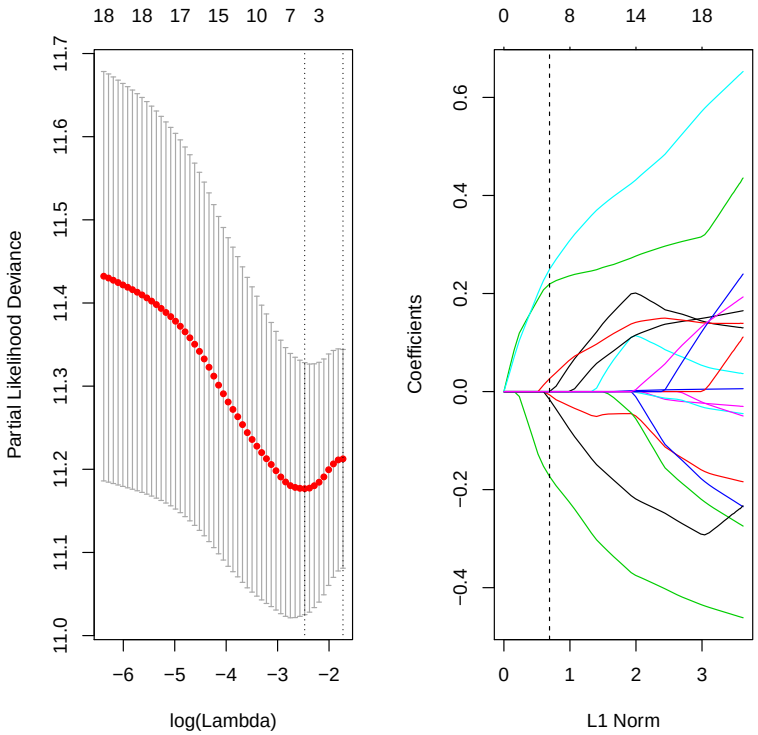

Figure 3.1: Values of the partial likelihood deviance and values of the coefficients plotted for a sequence of λ-values for the carcinoma data.

3.1.3 Carcinoma of the Oropharynx

To illustrate the lasso method for Cox regression, we will consider a dataset of 195 patients with oropharyngeal cancer taken from the book by Kalbfleisch and Prentice [1986]. Here, the patients survival time, an event indicator, the patients sex and age, the institution where the patients were treated, what treatment each patient received, the general condition of each patient, the site of the carcinoma in the oropharynx, the grade of the carcinoma, the t-stage and the n-stage was recorded. The TN staging classification gives a measure of the extent of the tumor at the primary site (T), and at regional lymph nodes (N). The grade variable is a measure of the degree of which the tumor cell resembles the host cell. The variables institution, sex, treatment, grade, t-stage, n-stage and site are all categorical variables with 6, 2, 2, 3, 4 and 4 levels, respectively. Since the effect on the mortality is too dominant, we do not include the general condition of the patients for purposes of illustration. The penalisation parameter λ was chosen

by 10-fold cross validation, where we aim to minimise the partial likelihood deviance (figure 3.1). To see how the lasso method works, it is illustrative to look at a plot of the coefficients for the models corresponding to a range of λ-values, which is shown in figure 3.1. Here we can see how the lasso shrinks the coefficients, and sets some to zero.

Table 3.1: The non-zero coefficients from a Cox model with a lasso penalty, fitted to the carcinoma data.

Variable	β
Institution 3	-0.007
Grade 2	0.026
Grade 3	-0.173
t-stage 2	-0.016
t-stage 4	0.219
n-stage 3	0.250

The coefficients from the fitted model with the optimal value of λ is shown in table 3.1, where we can interpret the coefficients as the effect of the variables having a given level, relative to the base level, which is 1. For example, we can see that institution number 3 has an estimated lower mortality rate than institution 1. One problem with the direct use of the lasso in this situation is that we might want to include or exclude an entire variable, and not just the levels of a categorical variable as individual variables. This problem has a solution in an adjusted version of the lasso method, which we will introduce next.

3.1.4 Group lasso

Yuan and Lin [2006], in their article, derive an extension to the lasso method to situations with categorical covariates, ensuring that grouped coefficients are pushed in and out of the model simultaneously. This is achieved by imposing a ridge penalty within each group, and then weighting the penalty of each group with the square root of the number of levels within each group. All the discussion in Yuan and Lin [2006] has the linear model in mind, with the objective function being the squared error loss function. We will here concentrate on the Cox model, with the objective function being the Cox partial log likelihood. To state the maximisation problem, some notation is needed. We assume that there are G variables, at least one of which is a categorical variable, where each variable has p_g levels. The maximisation problem to

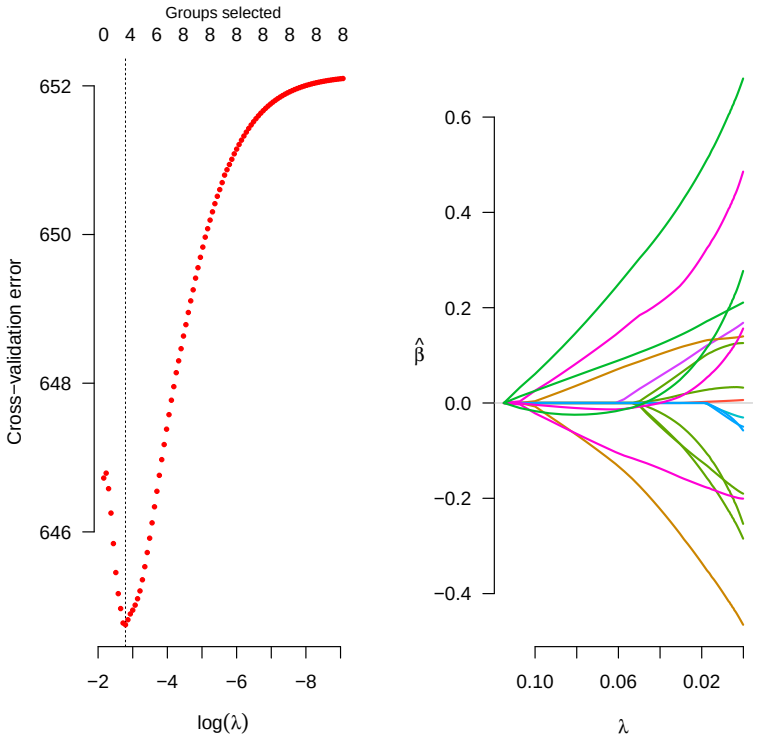

Figure 3.2: Values of the partial likelihood deviance and values of the coefficients plotted for a sequence of λ-values for the carcinoma data, for the group lasso model.

be solved can be expressed as

$$\underset{\beta}{\operatorname{argmax}} \left\{ l(\boldsymbol{\beta}) - \lambda \sum_{g=1}^{G} \sqrt{p_g} \left(\sum_{j=1}^{p_g} \beta_{gj}^2 \right)^{\frac{1}{2}} \right\}$$

where $l(\boldsymbol{\beta})$ is the Cox partial log likelihood.

3.1.5 Carcinoma of the Oropharynx, group lasso

Returning to the dataset of 195 patients with carcinoma of the Oropharynx, we can illustrate the group lasso method, which serves the purpose of ensuring that each of the grouped coefficients are pushed in and out of the model in unison. We here also choose the optimal penalty parameter λ via 10-fold cross validation, and choose the value of λ that yields the smallest cross validation error (see figure 3.2). To see how the grouped lasso behaves in comparison to the regular lasso, we here also plot the

coefficients for each λ to obtain the coefficient paths, which is shown in figure 3.2. We can here see that the individual variables in each group are set to zero at the same time, and the coefficient paths are somewhat smoother than for the regular lasso, due to the ridge-like penalty on the individual coefficients within each group. Concretely, we see that in contrast to the model with an ordinary lasso penalty, the institution variable is now excluded, and all the levels of t-stage and n-stage are included.

Table 3.2: The non-zero coefficients from a Cox model with a group lasso penalty, fitted to the carcinoma data.

Variable	β
Grade 2	0.060
Grade 3	-0.128
t-stage 2	-0.103
t-stage 3	-0.014
t-stage 4	0.143
n-stage 2	-0.017
n-stage 3	0.242
n-stage 4	0.087

The regression coefficients that are not set to zero are given in table 3.2. If we are to interpret this model, we can infer from the coefficients from the fit that patients with grade equal to 2 and 3 have an estimated higher and lower mortality rate compared to those with grade equal to 1, respectively. In addition, patients with t-stage equal to 2 or 3 have an estimated lower mortality rate than those with t-stage equal to 1, while those with t-stage equal to 4 have a higher mortality rate. Lastly, those patients with n-stage equal to 2 have an estimated lower mortality rate than those with n-stage equal to 1, and those with n-stage equal to 3 or 4 have an estimated higher mortality rate.

3.2 Landmarking and penalised Cox regression

A main goal of this thesis is to adapt the Cox model to situations where we have non-proportional hazards (or time-varying effects, if you will), and a high dimensional covariate space. We have looked at two extensions that try to solve these two problems by themselves, namely landmarking, and penalised partial likelihood methods, and our aim forward is to try and combine the two. The first approach we take is a naive one,

where we just maximise the penalised log partial likelihood for each of the landmark datasets.

3.2.1 Penalised sliding landmarking

As mentioned above, we first attempt to estimate coefficients at each landmark point as before, but with a penalised regression technique instead of ordinary Cox regression. Given S landmark points and interval width w, the likelihood that we maximise, omiting penalty terms, is the expression (2.8) given in section 2.3.2, which can be written as

$$ipl(\boldsymbol{\beta}(\mathbf{LM})) = \sum_{s=1}^{S} \sum_{i \in A_s} l_i(\boldsymbol{\beta}(LM_s)),$$

where $A_s = \{j \mid t_j \in [LM_s, LM_s + w]\}$ is the landmark dataset with left-truncation at $t = LM_s$ and right-censoring at $t = LM_s + w$. In the naive approach we first propose, we fit models to each landmark dataset independently, and thus also set a penalty independently for each of these models. Hence, what we are maximising, if we follow this approach is

$$\left\{ \sum_{s=1}^{S} \left(\sum_{i \in A_s} l_i(\boldsymbol{\beta}(LM_s)) - \lambda_s \sum_{j=1}^{p} |\beta_j(LM_s)| \right) \right\}.$$

It should be noted that there usually is a lot of overlap between the different landmark sets, and therefore this likelihood-like expression is not a proper partial likelihood, and the terms in the likelihood, as for the Cox partial likelihood, are not independent.

3.2.2 Primary biliary cirrhosis data

To illustrate the method outlined above, we will fit such a penalised sliding landmarking model to a dataset of 310 patients from the Mayo Clinic trial in primary biliary cirrhosis (PBC) of the liver conducted between 1974 and 1984. We here choose to use a new dataset, because we want a dataset without any categorical variables, with enough variables to illustrate the lasso, that also can be used to illustrate the next natural extension of the above method later. The dataset we use is a subset of the full dataset, where we have only included those individuals that were in the treatment/control groups, and that had measured values of all the covariates we included. The covariates included are

the patients age, sex, treatment received (D-penicillamine/placebo), whether or not the patient had an enlarged liver (hepato), ascites or blood vessel malformations in the skin (spiders), the measured values of aspartate aminotransferase (ast), serum bilirunbin (bili), serum cholesterol (chol), serum albumin, urine copper, alkaline phosphotase (alk.phos), and standardised blood clotting time (protime).

To be able to summarise the model in a simple manner, we fit the model with only a handful of landmarks, more precicely we choose $LM_1 = 0$ years, $LM_2 = 3$ years and $LM_3 = 6$ years. Setting $w = 5$ years yields 3 datasets, the first with administrative censoring at $t = 5$ years, the second and third with left truncation at $t = 3$ years and $t = 6$ years, and administrative censoring at $t = 8$ years and $t = 11$ years, respectively. We then fit a Cox regression model to each of these datasets with an L_1 penalty, where the individual penalty terms are chosen via 10-fold cross validation.

Table 3.3: Landmark estimates computed by individual lasso-penalised regressions for each landmark set, computed for the primary biliary cirrhosis data.

LM:	0 years	3 years	6 years
trt	0	0	0
age	0.013	0	0
sex	0	0.06	0
ascites	0.128	0.071	0.432
hepato	0.052	0.07	0
spiders	0	0	0
bili	0.083	0.174	0.191
albumin	−0.910	−0.895	−0.287
copper	0.003	0.00003	0
alk.phos	0	0	0
ast	0.001	0	0
protime	0.304	0	0

A summary of the model fit is shown in table 3.3. Looking at the table we see that not all of the variables that are set to zero are set to zero at every landmark point. As we would like to have a scheme that pushes covariates in and out of the model in unison, we need to come up with a different approach to fitting our model. If we are to interpret the fit given in table 3.3, we see there is no estimated effect of difference in treatment and that older patients have a higher mortality, but only at the start of the study. There is an estimated higher mortality rate among males at the second landmark, the patients with ascites have a higher mortality rate and patients with an enlargened liver have

an estimated higher mortality rate, but not towards the end of the study. There is no effect of blood vessel malformations in the skin, higher values of serum bilirunbin are associated with a higher mortality rate but the effect is not as strong for the first 5 years, and higher values of serum albumin are associated with a lower mortality rate. In addition, there is a estimated positive effect on the mortality rate at the first and second landmark point of increased urine copper and at the first landmark point of increased standardised blood clotting time.

3.2.3 Sliding landmarking with preselection of variables

A possible strategy when we are to fit sliding landmark models to datasets with a high number of covariates, is to first use a method to select a subset of the covariates, and then to fit a model to this reduced dataset. There are a variety of methods that allows us to select such a subset, but as we have here already introduced lasso regression we will focus on lasso selection. For an overview of some available techniques, see for instance Bøvelstad et al. [2007]. Thus our propoced procedure is here to first fit a penalised Cox regression model to the given dataset using a lasso-penalty, and then fitting a sliding landmark model using only the selected covariates from the lasso-model. For brevity, we will occasionally refer to this scheme by the acronym SL-PS.

3.2.4 Sliding landmark analysis of the PBC data with preselecion of variables

To exemplify, we use the above method to estimate regression coefficients for the PBC data discussed in section 3.2.2. To ensure perfect comparability we use the same folds as in 3.2.2 when choosing the tuning parameter for the lasso model that we use to select covariates. The landmark coefficients are then estimated as discussed in chapter 1, with landmarks at 0, 3 and 6 years, and an interval width of $w = 5$ years.

Table 3.4: The resulting coefficients from the sliding landmark analysis of the primary biliary cirrhosis data using lasso selection.

LM:	0 years	3 years	6 years
age	0.031	0.017	0.039
ascites	0.095	0.229	0.735
hepato	0.438	0.489	0.31
spiders	0.022	0.12	0.34
bili	0.075	0.184	0.187
albumin	-1.12	-1.441	-1.347
copper	0.003	0.002	0.003
ast	0.004	0.001	0.004
protime	0.434	-0.098	0.127

The lasso model selects all of the covariates except treatment, sex, blood vessel malformations in the skin and alkaline phosphotase. The sliding landmark estimates are summarised in table 3.4. If we are to compare with the previous analysis in section 3.2.2, we see that the with the exception of the nonzero estimate at the second landmark of the covariate sex, the same covariates have been set to zero across all the landmarks. Aside from the fact that some of the landmark coefficients are zero for the analysis in 3.2.2, the results seem quite similar, but are larger in absolute value for the model in the present section. This is of course to be expected as we are not penalising the estimates here.

3.2.5 Group penalised sliding landmarking

In the example in 3.2.2, we saw that when we simply fit penalised Cox regression models with a lasso penalty to each of the landmark datasets, the resulting model has an unwanted property. Namely, that the coefficient of a variable can be excluded and included at different landmark points. What we want instead, is that the effect of a variable is either estimated at all the landmark points, or excluded from the model entirely. To achieve this, we want to estimate a model based on all the landmark datasets, with a group penalty on each group of coefficients belonging to the same variable at different landmark points. Specifically, we seek to maximise the expression

$$\left\{ \sum_{s=1}^{S} \sum_{i \in A_s} l_i \left(\boldsymbol{\beta}(LM_s) \right) - \lambda \sum_{j=1}^{p} \left(\sum_{s=1}^{S} \beta_j^2(LM_s) \right)^{\frac{1}{2}} \right\},$$

with respect to λ and $\boldsymbol{\beta}(\mathbf{LM})$. We wish however, to use existing software to do this. This is not possible to do directly with the software packages that are avaialble to us at the present time. The closest approximate solution is to stack the individual datasets created by the *dynpred* software by Putter [2015], and then fit the model using for instance the *grpreg* package [Breheny and Huang, 2015]. One can show that using this strategy, what we actually maximise is

$$\left\{ \sum_{s=1}^{S} \sum_{i \in A_s} l_i^* \left(\boldsymbol{\beta}(LM_s) \right) - \lambda \sum_{j=1}^{p} \left(\sum_{s=1}^{S} \beta_j^2(LM_s) \right)^{\frac{1}{2}} \right\},$$

where

$$l_i^* \left(\boldsymbol{\beta}(LM_s) \right) = d_i \left(\boldsymbol{\beta}^T(LM_s)\mathbf{x}_i - \log \left(\sum_{\ell \in \mathcal{R}_i} \sum_{\{k | t_l \geq LM_k\}} \exp \left(\boldsymbol{\beta}^T(LM_k)\mathbf{x}_\ell \right) \right) \right). \tag{3.1}$$

Thus the risk used when computing the individual likelihood contributions are here incorrect, as the observations often are counted more than once and coefficients belonging to other landmarks also contribute. In addition this approach requires that all the landmark datasets are kept in-memory at the same time. This might impose too large a constraint on the number of landmark points that can be included, especially for high-dimensional datasets. Nevertheless, it offers a way of expanding on the approach outlined in 3.2.1, potentially leading to a more parsimonous and interpretable model.

3.2.6 Group penalised sliding landmarking analysis of PBC data

In an attempt to exemplify and illustrate the approach outlined in section 3.2.5, we will fit such a model to the liver cirrhosis data discussed in section 3.2.2. To choose the penalty parameter λ, we use 10-fold cross validation. Since we here reuse observations in the model, it is now quite important to pay attention to how the folds are generated. We would like that when an observation is excluded from one of the model fits in the cross validation procedure, it is removed from all the landmark datasets at the same time. To ensure this, we define and assign the fold numbers to each of the observations, prior to creating and stacking the landmark datasets.

Table 3.5: Landmark estimates computed by a group lasso-penalised regression with stacked landmark datasets, computed for the primary biliary cirrhosis data.

LM:	0 years	3 years	6 years
trt	0	0	0
age	0.016	0.013	0.011
sex	0	0	0
ascites	0.181	0.057	0.239
hepato	0.235	0.163	0.099
spiders	0	0	0
bili	0.108	0.121	0.138
albumin	−0.927	−0.995	−1.051
copper	0.003	0.002	0.001
alk.phos	0	0	0
ast	0.0009	0.0006	0.0004
protime	0.162	0.147	0.134

The model is fitted with landmark points at $LM_1 = 0$ years, $LM_2 = 3$ years and $LM_3 = 6$ years, and an interval with of $w = 5$ years. A summary of the fitted model is given in table 3.5. From the table, we can see that the variables treatment, sex, spiders (blood vessel malformations in the skin) and alkaline phosphostase are excluded from the final model. Interpreting the coefficients, we see that we estimate that older patients, patients with ascites, enlargened livers, higher values of serum bilirunbin, higher values of copper urine, higher values of aspartate amonotransferate, longer standardised blood clotting time, and lower values of albumin are positively associated with the mortality rate. While studying the coefficient estimates from different models is interesting in itself, it is not clear which estimates are better or worse in any sense. If we are interested in survival predictions from given time-points ahead in time, which we in this thesis are, we could here compare the models using the measures described in section 2.4.2.

To do this we split the data in two, where two thirds of the data are used to estimate the models, and the rest are used for the evaluation of the models. The models we will compare the predictive of performance of the group penalised sliding landmarking model to is the lasso-model, and the sliding landmark model with preselection. For these three models, we compute the dynamic Brier score for predicting survival 3 years ahead in time at the landmarks $LM_1 = 0$, $LM_2 = 3$ and $LM_3 = 6$, and the corresponding dynamic R^2 measures. These are shown in figure 3.3, where we can see that for the first landmark the sliding landmark model with preselection performs

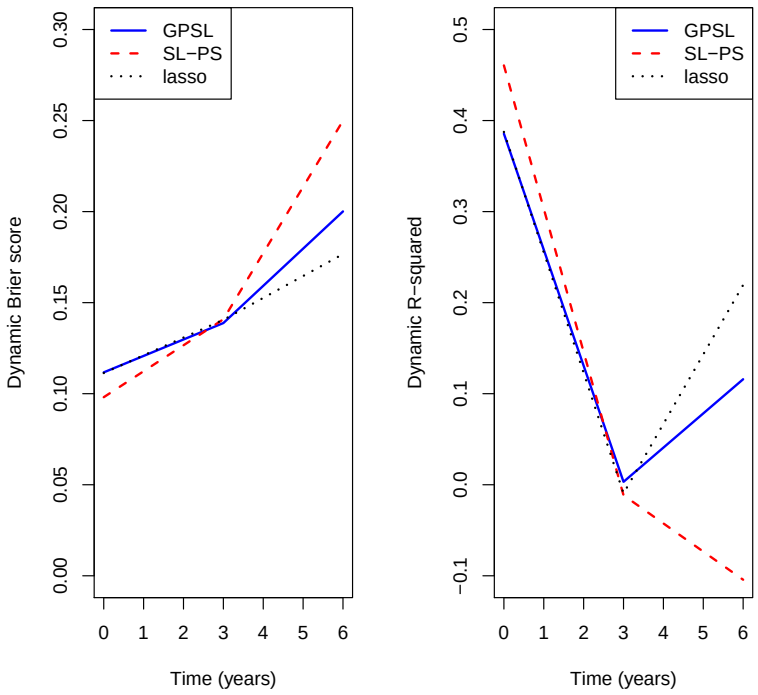

Figure 3.3: Plots of the dynamic Brier score and the dynamic R-squared measure for the group penalised sliding landmarking (GPSL), sliding landmarking with preselection (SL-PS) and the lasso model for the pbc data.

better than the two other methods, where the two other seem to have the exact same score. For the second landmark, all the three models seem to have scores that are very close. At the last landmark, the lasso seems to perform drastically better than the group penalised sliding landmarking, which in turn seems to perform drastically better than the sliding landmarking with lasso selection. This coul perhaps seem contradictive as these measures are specifically designed to measure the predictive accuracy of dynamic predictions, and the sliding landmark model is specifically designed to produce good predictions of this kind. However, there is less data available at the last landmark, and due to this the estimates become somewhat unstable, and are thus outperformed by the methods that incorporate a form of shrinkage, in addition to selecting a subset of the covariates.

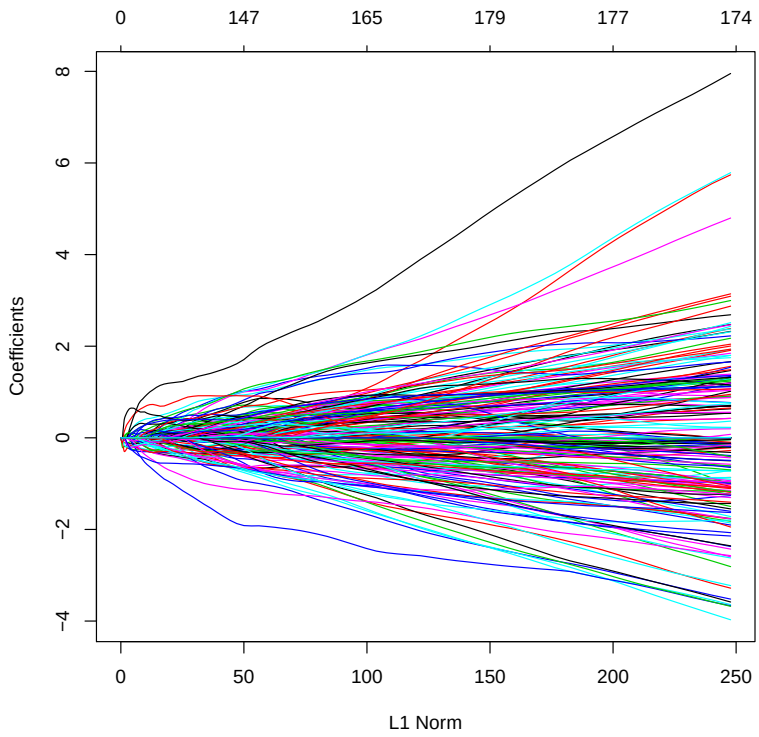

Figure 3.4: Values of the coefficients plotted against the L_1 norm of the coefficient vector, for models fitted to the Dutch breast cancer data.

3.2.7 Dutch breast cancer data

To further illustrate the lasso method for Cox regression, specifically the high dimensional setting, we will consider a dataset containing 4919 gene expression measurements and censored survival times for 295 Dutch women dicussed in the paper of van Houwelingen et al. [2006]. This dataset consists of a subset of the gene-expressions in cDNA arrays containing 24885 genes, which were reduced to 4919 genes [van Houwelingen et al., 2006]. The subjects were selected from the fresh-frozen-tissue bank of the Netherlands Cancer Institute, where the criteria were that the tumour was a primary invasive breast carcinoma, less than 5 cm in diameter, that the age at diagnosis was 52 years or younger, that the calendar year of diagnosis was between 1984 and 1995, and that there was no previous history of cancer, except non-melanoma skin cancer. We fit a Cox regression model with a lasso penalty to the data, where the penalty is chosen by 10-fold cross validation using the R [R Core Team, 2017] package glmnet [Friedman

et al., 2010, Simon et al., 2011].

Table 3.6: Coefficients from Cox model lasso penalty, fitted to the Dutch breast cancer data.

Gene No.	β
128	-0.102
1925	0.003
2042	-0.099
2242	-0.204
2246	-0.164
2363	-0.102
2816	0.092
3154	0.281
3394	-0.237
4175	-0.197
4176	-0.141
4197	0.004
4272	0.093
4309	0.545
4331	-0.091
4616	-0.018

The fitted (non-zero) coefficients from the final model are shown in table 3.6,where the interpretation is that a higher value of the measured gene expression is associated with a higher mortality rate for those genes with an estimated positive coefficient, and a lower mortality rate for those genes with an estimated negative coefficient. A plot showing the relationship between the L_1-norm of the coefficient vector and the individual values of the coefficients, and in addition the number of non-zero coefficients in the model is given in figure 3.4. One would normally standardise data when fitting models with a lasso penalty since it is sensitive to scaling, but we have here chosen not to do so because the gene expression measurements already are on the same scale.

3.2.8 Sliding landmark analysis of the Dutch breast cancer data with preselection of variables

As previously alluded to, a possible solution that allows us to estimate sliding landmark models for high dimensional datasets is to first select a subset of covariates, and then to fit a sliding landmark model to the low-dimensional dataset with only these covariates.

Naturally, our first attempt to estimate a model to a high dimensional dataset utilises this strategy. Therefore, we use the very covariates that are selected in the lasso-analysis of the previous section in our sliding landmark analysis of the same dataset. To fit the model itself we use landmarks at 0, 2.5, 5 and 7.5 years, and an interval width of 2.5 years.

Table 3.7: The resulting landmark coefficients from the sliding landmark analysis of the Dutch breast cancer data, using lasso preselection of variables.

Gene no.	0 years	2.5 years	5 years	7.5 years
128	-0.356	-0.446	0.384	0.819
1925	1.542	1.004	0.144	2.213
2042	-0.484	-0.837	-1.892	-1.314
2242	-0.291	-1.298	-0.336	-2.341
2246	-0.106	-1.699	-0.393	0.793
2363	-1.584	0.052	-0.783	-6.551
2816	2.076	-0.108	-0.198	-3.059
3154	0.577	0.421	0.268	4.732
3394	-1.387	0.071	1.044	3.557
4175	-1.534	0.156	-0.716	-2.832
4176	-1.17	0.19	-1.337	-1.541
4197	-0.706	1.472	1.256	2.464
4272	-0.955	0.483	0.179	3.278
4309	0.916	0.981	1.968	2.999
4331	0.117	-0.264	-1.037	-5.918
4616	1.425	0.045	-0.933	0.811

The coefficients from the analysis are given in table 3.7, where we see that compared to the lasso analysis the many of the coefficients are very large in absolute value, in particular at the last landmark dataset. They also seem to vary greatly in size across the same covariates. While we undoubtedly are able to estimate a model to the data using this technique, some of the resulting coefficients are in this case much to large, perhaps because this strategy is too unrobust.

3.2.9 Group penalised sliding landmarking analysis of Dutch breast cancer data

A main goal of this thesis is to try and develop a method that allows for estimation of time-varying effects in Cox regression, for high dimensional settings. While we could

use the strategy used in the previous example, we desire an approach to estimating the coefficients that does both variable selection and estimation at the same time, and that also penalises the estimates. Our initial candidate for such a method is the group penalised sliding landmarking approach outlined in 3.2.5. In this section, we will take a look at a concrete example, namely the breast cancer data discussed in the two previous sections, and illustrate group penalised sliding landmark analysis on this dataset. To fit the model, we set an interval width of $w = 5$ years, landmark points at $LM_1 = 0$ years, $LM_2 = 2.5$ years, $LM_3 = 5$ years, and $LM_4 = 7.5$ years, and choose the penalty parameter λ by 10-fold cross validation, using the same folds as for the analyses from the two previous sections. The resulting model contains estimated non-zero coefficients for no less than 102 covariates, which needless to say are to many to display in a table here. It is worth mentioning that although the same folds are used, only 6 of the covariates that appeared in the lasso model reappear in this model, despite the fact that it contains as many as 102 covariates.

Table 3.8: Table containing the group penalised sliding landmark estimates of the coefficients of the coefficients that are also estimated to have a non-zero effect by lasso penalised Cox regression.

Gene no.	0 years	2.5 years	5 years	7.5 years
2042	-0.145	-0.147	-0.096	0.022
2363	-0.15	-0.117	-0.113	-0.023
2816	0.448	0.439	0.452	0.672
3154	0.157	0.184	0.269	0.443
4197	0.01	0.024	0.024	0.022
4331	-0.076	-0.129	-0.16	-0.126

The interpretation of these coefficients are as in previous examples, higher values of a measured gene expression is associated with a higher mortality rate if the coefficient is positive, and lower if the coefficient is negative. The estimates that are in common for both the methods are given in table 3.8. These are much smaller and seem more stable for the group penalised method, which is reasonable since this model contains a lot more coefficients in total, and the estimates are penalised.

In a similar way as for the pbc data, we will here also assess the quality of the predictions made from the lasso model, the sliding landmark estimates made using the lasso selection, and the group penalised sliding landmark estimates using the measures introduced in section 2.4.2. We here estimate the models using roughly two thirds of

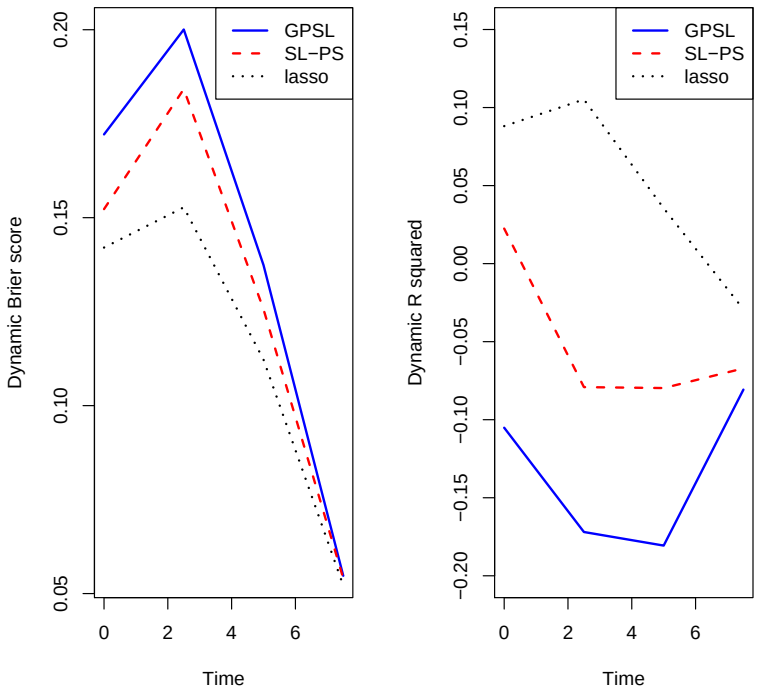

Figure 3.5: Plots of the dynamic Brier score and the dynamic R-squared measure for the group penalised sliding landmarking (GPSL), sliding landmarking with preselection (SL-PS) and the lasso model for the Dutch breast cancer data.

the data, and evaluate the predictions using the remaining third. Plots of the dynamic Brier scores and related R^2 measures are shown in figure 3.5. Here we see that in terms of predictions, the lasso estimates are most accurate, followed by the sliding landmark estimates using the lasso selection, while the group penalised sliding landmark estimates are the least accurate. Concentrating on the dynamic R^2 based on the dynamic Brier scores, we see that the lasso model is better than the null model at the three first landmarks, but worse at the last landmark. The sliding landmark model with preselection is better than the null model at the first landmark, and worse compared to the null model at the three last landmark. The group penalised model is worse than the null model at all of the four landmarks. Thus none of the two considered models seem to outperform the lasso model for this dataset, and these test cases. For the sliding landmark model using the lasso selection, is seems that the problem is a

lack of shrinkage of the estimates, which naturally makes the coefficient estimates less stable, and therefore more inaccurate predictions. For the group penalised sliding landmark analysis, it seems that the error we make in terms of the baseline when trying to estimate the model in the way we do here as discussed in section 3.2.5, is large enough that the estimates become too inaccurate, and useless for predictions.

This example also neatly illustrates the computational disadvantages of the approach outlined in section 3.2.5, as this requires the design matrix of the regression problem to have p times S columns, where p is the number of covariates in the dataset, and S is the number of landmark points, in addition, the number of rows also grows as S increases. With several thousand variables in the dataset, it will be challenging to store the design matrix of a model with more than a handful of landmark points in the memory of a standard computer. This might prompt us to go for the two-stage process we also have discussed in this chapter, but as we have seen, this method also has some undesirable properties. The ideal solution might be the group penalised sliding landmark approach, but limitations of the available software might render further pursuit of this idea to be beyond the scope of this thesis. If we however were to follow this idea we would have to store the design matrix in a memory-efficient way, which of course is no real challenge, but we would also have to develop a a way of estimating the model using a quadratic optimisation algorithm taylored to this specific problem. This is of course a possible solution, but we will instead go down a somewhat different path in our pursuit of truth, as it were, by designing an algorithm using the likelihood function (2.8), and an estimation technique known as boosting.

Chapter 4

Boosting in survival models

An alternative to penalised maximum likelihood estimation is the concept of likelihood-based boosting. Boosting originated in the field of machine learning, and was originally developed for classification, perhaps the most notable example being the AdaBoost algorithm [Freund and Schapire, 1997]. Boosting has later been adapted to a statistical setting, instigated by Friedman et al. [2000] who showed that AdaBoost minimises a certain exponential loss function. They also showed that this loss function is related to the binomial log likelihood, and developed an algorithm called LogitBoost, which fits an additive logistic regression model. The LogitBoost algorithm can be adapted to work for any exponential family, and for proportional hazards models [Ridgeway, 2001]. So, what is boosting? All boosting procedures are iterative procedures that adapt to some measure of error from the data. For AdaBoost, this involves weighting the data with the weights being decided by the misclassification error. An important procedure for regression, and in general, is the gradient boost algorithm by Friedman [2001], which is a general description of a way to iteratively estimate the minimiser of a loss function by moving in the direction of the gradient of the loss function. From a statistical point of view, one usually works with maximising the log likelihood instead of minimising loss functions, but we can adapt the gradient boosting procedure by thinking of the negative log likelihood as a loss function. In this manner the gradient boosting algorithm can be adapted and used as a means to fit for instance exponential family models and proportional hazards regression models, such as the ones we are interested in in this thesis.

4.1 Gradient boosting

In the paper by Friedman [2001], he proposes the general gradient boosting algorithm. The aim of the algorithm is, as previously mentioned, to minimise a loss function. The algorithm is iterative in nature, and the idea of it is, for each iteration m, to compute an increment to a predictor that aims to move in the direction of the negative derivative of the loss function. The first step is therefore to compute this derivative evaluated at the previous estimate

$$\tilde{\mathbf{y}}_m = -\frac{\partial L\left(\mathbf{y}, F_{(m-1)}(\mathbf{x})\right)}{\partial F(\mathbf{x})},$$

where F is a general predictor that maps values of $\mathbf{x}$ to values of $\mathbf{y}$, an example being the linear predictor $F(\mathbf{x}) = \boldsymbol{\beta}^T\mathbf{x}$. One then subsequently fits a model $h(\mathbf{x})$ that predicts $\tilde{\mathbf{y}}$, and regresses on h using the previous estimate F as an offset. The new estimate is then the previous estimate plus the increment of h scaled by a regression coefficient ρ_m. The latter is sometime referred to as a line search, because it involves finding the point on the line $F_{(m-1)}(\mathbf{x}) + \rho_m h(\mathbf{x})$ that minimises the loss function. One can use any model fitting procedure to compute $h(\mathbf{x})$, but Friedman [2001] proposes a least squares approach, due to the computational advantages of least-squares algorithms. A natural adoption of the general gradient boosting procedure for Cox regression is to replace the general loss function with the negative partial log likelihood, and $h(\mathbf{x})$ by a least squares fit [Ridgeway, 2001]. Note that we here work with the prognostic index in place of the general predictor such that we can obtain regression coefficients, and interpret them in the same way as for regular maximum likelihood-fitted Cox regression models. Another thing to note is that one often scales the updates to the regression coefficients (or to F in the general algorithm) by a parameter $\nu \in (0, 1)$. This parameter is usually called the 'learning rate', because it controls how large the updates of the coefficients are in each step, and thus also the rate at which the algorithm 'learns' the relationships between the covariates and the dependent variable. One usually chooses the number of iterations to use by some form of cross-validation, and the controlling the learning rate ν helps in finding a model that minimises the cross-validated loss by making smaller steps in each iteration.

4.2 Model-based boosting for Cox regression

The approach to boosting for Cox regression outlined in the previous section is sometimes refered to as *model-based* boosting [De Bin, 2016]. The procedure usually does not, however, compute the estimates in the way stated above. Instead of simultaneously updating all the regression parameters, only one parameter is usually updated in each iteration. This is done both to be able to fit sparse models, and to be able to handle high-dimensional data. One adapts the algorithm by computing univariate linear fits to the pseudo-observations

$$\tilde{y}_i = \frac{\partial l_i\left(\boldsymbol{\beta}_{(m-1)}\right)}{\partial \boldsymbol{\beta}^T \mathbf{x}_i} = d_i - d_i \frac{\exp(\boldsymbol{\beta}_{(m-1)}^T \mathbf{x}_i)}{\sum_{\ell \in \mathcal{R}_i} \exp(\boldsymbol{\beta}_{(m-1)}^T \mathbf{x}_\ell)},$$

where l_i is the i-th contribution to the partial log likelihood. The univariate linear fits are computed as

$$\hat{\gamma}_j = \frac{\sum_{i=1}^n \tilde{y}_i x_{ij}}{\sum_{i=1}^n x_{ij}^2} \quad j = 1, 2, ...p,$$

where one then chooses to update the coefficient of the covariate j^* such that

$$j^* = \underset{j}{\operatorname{argmin}} \sum_{i=1}^{n} (\tilde{y}_i - \hat{\gamma}_j x_{ij})^2.$$

The boosting estimates are then subsequently updated as

$$\begin{aligned} \beta_{j^*}^{(m)} &= \beta_{j^*}^{(m-1)} + \nu \gamma_{j^*} \\ \beta_j^{(m)} &= \beta_j^{(m-1)}, \quad j \neq j^* \end{aligned}$$

.

This model-based boosting algorithm for the Cox model with a componentwise approach to estimation is implemented in the R package *mboost* [Hothorn et al., 2017].

4.3 Likelihood-based boosting for Cox regression

The alternative to the model-based approach is what De Bin [2016] refers to as *likelihood-based* boosting. The main difference compared to the model-based approach is that

one circumvents the computation of pseudo-observations (negative derivative of the loss). Instead, one computes the updates to the coefficients by a single iteration of the Newton-Raphson algorithm to maximise a penalised version of the partial log likelihood, using the current estimate as an offset. To state these steps in detail, we first need to state the relevant expressions formulated with an offset term. The penalised likelihood that we aim to maximise is

$$l^{pen}(\mathbf{b}|\hat{\boldsymbol{\beta}}) = \sum_{i=1}^{n} d_i \left[(\mathbf{b}+\hat{\boldsymbol{\beta}})^T \mathbf{x}_i - \log\left(\sum_{\ell \in \mathcal{R}_i} \exp((\mathbf{b}+\hat{\boldsymbol{\beta}})^T \mathbf{x}_\ell) \right) \right] - \frac{\lambda}{2} \sum_{j=1}^{p} b_j^2,$$

where $\hat{\boldsymbol{\beta}}$ is the current estimate, and $\mathbf{b}$ is an increment to the current estimate. The expression above can perhaps be written in a neater way by using the notation

$$s^{(0)}(\boldsymbol{\beta}, t_i) = \sum_{\ell \in \mathcal{R}_i} \exp(\boldsymbol{\beta}^T \mathbf{x}_\ell).$$

It will also prove convenient to introduce the notation

$$s_j^{(1)}(\boldsymbol{\beta}, t_i) = \sum_{\ell \in \mathcal{R}_i} x_{\ell j} \exp(\boldsymbol{\beta}^T \mathbf{x}_\ell),$$

and

$$s_j^{(2)}(\boldsymbol{\beta}, t_i) = \sum_{\ell \in \mathcal{R}_i} x_{\ell j}^2 \exp(\boldsymbol{\beta}^T \mathbf{x}_\ell)$$

for the first and second partial derivative of $s^{(0)}(\boldsymbol{\beta}, t_i)$, respectively. Using this notation, we can express the first and second partial derivatives of $l^{pen}(\mathbf{b}|\hat{\boldsymbol{\beta}})$ as

$$\frac{\partial l^{pen}(\mathbf{b}|\hat{\boldsymbol{\beta}})}{\partial b_j} = \sum_{i=1}^{n} d_i \left[x_{ij} - \frac{s_j^{(1)}(\mathbf{b}+\hat{\boldsymbol{\beta}}, t_i)}{s^{(0)}(\mathbf{b}+\hat{\boldsymbol{\beta}}, t_i)} \right] - \lambda b_j,$$

and

$$\frac{\partial^2 l^{pen}(\mathbf{b}|\hat{\boldsymbol{\beta}})}{\partial b_j^2} = -\sum_{i=1}^{n} d_i \frac{s_j^{(2)}(\mathbf{b}+\hat{\boldsymbol{\beta}}, t_i) s^{(0)}(\mathbf{b}+\hat{\boldsymbol{\beta}}, t_i) - s_j^{(1)}(\mathbf{b}+\hat{\boldsymbol{\beta}}, t_i)^2}{s^{(0)}(\mathbf{b}+\hat{\boldsymbol{\beta}}, t_i)^2} - \lambda.$$

The updates are then, as previously mentioned, computed in a similar way to one iteration of the Newton-Raphson algorithm for one covariate at a time. To make later generalisations easier to comprehend we use the notation

$$u_j^{pen}(\boldsymbol{b}|\hat{\boldsymbol{\beta}}) = \frac{\partial l^{pen}(\mathbf{b}|\hat{\boldsymbol{\beta}})}{\partial b_j},$$

and

$$J_j^{pen}(\boldsymbol{b}|\hat{\boldsymbol{\beta}}) = -\frac{\partial^2 l^{pen}(\mathbf{b}|\hat{\boldsymbol{\beta}})}{\partial b_j^2},$$

which can also be written as

$$u_j^{pen}(\boldsymbol{b}|\hat{\boldsymbol{\beta}}) = u_j(\boldsymbol{b}|\hat{\boldsymbol{\beta}}) - \lambda b_j,$$

and

$$J_j^{pen}(\boldsymbol{b}|\hat{\boldsymbol{\beta}}) = J_j(\boldsymbol{b}|\hat{\boldsymbol{\beta}}) + \lambda,$$

where u_j and J_j are the unpenalised counterparts of u_j^{pen} and J_j^{pen}. Because our aim is to later generalise this algorithm to landmark models, it seems worthwile to remind ourselves of how the Newton-Raphson update of the estimate is derived, formulated in our current context. The aim of the algorithm is in each step to try and move towards the maximum of the partial log likelihood in one dimension at a time, but to do this we need to determine what the individual updates should be. The trick we employ is to approximate the offset penalised partial log likelihood with a second order Taylor-expansion about 0 in the direction of each of the p covariates. Phrased somewhat differently, we can say that we compute p different univariate Taylor approximations to the offset penalised partial log likelihood, where we in each of them differentiate with respect to the the regression parameter of one covariate, and treat the rest as constant. This expansion in the direction of the j-th covariate is

$$\begin{aligned} l^{pen}(\mathbf{b}|\hat{\boldsymbol{\beta}}) &\approx l^{pen}(\mathbf{0}|\hat{\boldsymbol{\beta}}) + u_j^{pen}(\mathbf{0}|\hat{\boldsymbol{\beta}})b_j - \frac{J_j^{pen}(\mathbf{0}|\hat{\boldsymbol{\beta}})}{2}b_j^2 \\ &= l(\mathbf{0}|\hat{\boldsymbol{\beta}}) + u_j(\mathbf{0}|\hat{\boldsymbol{\beta}})b_j - \frac{J_j(\mathbf{0}|\hat{\boldsymbol{\beta}}) + \lambda}{2}b_j^2, \end{aligned} \tag{4.1}$$

which is a second degree polynomial in b_j and thus its maximum can be found by finding the value of b_j that satisfies that the derivative of it is zero. Hence, the potential update of the regression coefficient of the j-th covariate is

$$b_j = \frac{u_j(\mathbf{0}|\hat{\boldsymbol{\beta}})}{J_j(\mathbf{0}|\hat{\boldsymbol{\beta}}) + \lambda}.$$

To select which covariate to update the regression coefficient of, the natural solution is to use the penalised partial log likelihood evaluated at the j-th update as a scoring

criteria. However, while this is largely unproblematic for datasets without too many covariates, it can impose a large computational constraint for high-dimensional data. This is due to the form of the partial log likelihood, which takes quite a few steps to compute. This is not an issue for a problem with a few hundred observations and up to around a hundred covariates. But since we have to compute the partial log likelihood for each covariate in each boosting step, this accumulates and becomes infeasible if we have several thousand covariates. The alternative and preferred solution is therefore instead to use a scoring measure based on an approximation of the partial log likelihood. The approximation we choose is precisely the second order Taylor approximation in (4.1). If we insert our potential updates for each covariate into (4.1), we see that we get

$$\begin{aligned} & l(\mathbf{0}|\hat{\boldsymbol{\beta}}) + u_j(\mathbf{0}|\hat{\boldsymbol{\beta}})\frac{u_j(\mathbf{0}|\hat{\boldsymbol{\beta}})}{J_j(\mathbf{0}|\hat{\boldsymbol{\beta}}) + \lambda} - \frac{J_j(\mathbf{0}|\hat{\boldsymbol{\beta}}) + \lambda}{2}\left(\frac{u_j(\mathbf{0}|\hat{\boldsymbol{\beta}})}{J_j(\mathbf{0}|\hat{\boldsymbol{\beta}}) + \lambda}\right)^2 \\ = \; & l(\mathbf{0}|\hat{\boldsymbol{\beta}}) + \frac{1}{2}\frac{u_j(\mathbf{0}|\hat{\boldsymbol{\beta}})^2}{J_j(\mathbf{0}|\hat{\boldsymbol{\beta}}) + \lambda}. \end{aligned}$$

Thus, choosing the coefficient update that maximises the second order approximation to the partial log likelihood is equivalent to scoring them using the scoring measure

$$W^{(j)} = \frac{u_j(\mathbf{0}|\hat{\boldsymbol{\beta}})^2}{J_j(\mathbf{0}|\boldsymbol{\beta}) + \lambda},$$

because $l(\mathbf{0}|\hat{\boldsymbol{\beta}})$ is the same for all covariates, and scaling the measures does not change the rank-order. The regression coefficient we choose to update is then the covariate with index j^* such that

$$j^* = \underset{j}{\text{argmax}}\, W^{(j)}.$$

I.e., the new estimates are

$$\begin{aligned} \beta_{j^*}^{(m)} &= \beta_{j^*}^{(m-1)} + b_{j^*} \\ \beta_j^{(m)} &= \beta_j^{(m-1)} \quad j \neq j^*. \end{aligned}$$

Binder and Schumacher [2008] advocate using the same scoring variable as given above to choose the update in each iterations. But rather than refer to an argument similar to the one given here, they refer to it as a penalised version of the score test statistic, based on a low-order Taylor approximation, which is perhaps because a first order

approximation of the partial log likelihood leads to the same scoring measure as above. While this is of course absolutely true, we think that in this thesis it makes more sense to reason as we have done above. This is mainly because we are to generalise this approach to landmark models, where the notion of a score test statistic may not make as much sense as in the Cox regression setting. Due to this, reasoning in this way might obfuscate rather than clarify the argument. Instead, we should keep in mind that what we merely are doing is choosing the covariate that maximises the same approximation that we use to derive the updates, in the hope that this yields the largest increase of the partial log likelihood. Software that implements the likelihood boosting algorithm for Cox regression is available through the package CoxBoost [Binder, 2013].

4.3.1 Boosted Cox regression analysis of the primary biliary cirrhosis data

The techniques of likelihood-based boosting, model-based boosting as well as the ridge and the lasso are well suited for high dimensional data, but before we consider a high-dimensional case, we look at a simpler example. To this end, we revisit the primary biliary cirrhosis data discussed in section 3.2.2, and fit models to this data using the four aforementioned techniques, as well as ordinary Cox regression. An important thing to remember is that, like the ridge and lasso, the likelihood-based boosting requires centering non-dichotomous covariates, and transforming them such that their empirical variance is equal to one. Like the *glmnet* package, the *CoxBoost* package offers to do this internally. However, unlike the *glmnet* package the *CoxBoost* package does not transform the coefficients back to the original scale of the covariates, therefore we must take care of this ourselves. As we have previously mentioned, and as we may infer from the computation

$$\begin{aligned}\alpha(t|\mathbf{x}) &= \alpha_0(t)\exp(\boldsymbol{\beta}^T\mathbf{x}) \\ &= \alpha_0(t)\exp(\boldsymbol{\beta}^T(\mathbf{x}-\overline{\mathbf{x}}+\overline{\mathbf{x}})) \\ &= \alpha_0(t)\exp(\boldsymbol{\beta}^T\overline{\mathbf{x}})\exp(\boldsymbol{\beta}^T(\mathbf{x}-\overline{\mathbf{x}})),\end{aligned}$$

centering the covariates does not change the value of the estimated coefficients, it merely moves a part of the risk function to the baseline hazard. Scaling the covariates can be viewed as multiplying the covariate vector with a diagonal matrix $\mathbf{D}$, and hence

from the computation

$$\begin{aligned}\alpha(t) &= \alpha_0(t)\exp(\boldsymbol{\beta}^T\mathbf{x}) \\ &= \alpha_0(t)\exp(\boldsymbol{\beta}^T\mathbf{D}^{-1}\mathbf{D}\mathbf{x}) \\ &= \alpha_0(t)\exp((\mathbf{D}^{-1}\boldsymbol{\beta})^T\mathbf{D}\mathbf{x}),\end{aligned}$$

where the diagonal entries of $\mathbf{D}$ are the empirical standard deviations of the corresponding covariate column of the design matrix, we can deduce that the resulting coefficient estimates $\boldsymbol{\beta}^*$ are not on the same scale as the original data, but multiplied by the empirical standard deviation of each covariate. I.e., the estimated effect of the j-th covariate is $\beta_j^* = \hat{\sigma}_{x_j}\beta_j$, and hence to revert to the same scale as the data, we must divide each estimated coefficient with the same factor as the corresponding covariate column is divided by in the process of standardising the covariates.

When fitting the models, the tuning parameters in each of the algorithms are all chosen via cross-validation, using the same folds to ensure comparability. The penalty parameter for the likelihood-based boosting is set to $\lambda = 1116$, and the learning rate for the model-based boosting is set to $\nu = 0.1$. The reasons for these choices is merely that these are the default values set in the packages, where 1116 is 9 times the number of events in the dataset, which is the default value of λ for the CoxBoost package. The values of the tuning parameters that are selected, in this case the number of boosting steps, are $M_{CoxBoost} = 94$ for CoxBoost, and $M_{mboost} = 339$ for mboost. The resulting coefficients are all given in table 4.1. From the table we see that overall, ordinary Cox regression yield the largest estimates in absolute value, and that the CoxBoost estimates are similar to the Lasso estimates, but slightly larger. The lasso renders the sparsest model, followed by the CoxBoost, while mboost gives us the least sparse model. That is, of the techinques that incorporate some form of model selection.

4.3.2 Boosted Cox regression analysis of the Dutch breast cancer data

To exemplify the use of model-based boosting and likelihood-based boosting for Cox regression in situations where $p > n$, we will fit Cox regression models to the Dutch breast cancer data using the *CoxBoost* and *mboost* software. To ensure comparability, we divide the observation into 10 folds ourselves, instead of leaving it to the built-in validation of the software packages to decide how to split the data. For the likelihood-based version, we set $\lambda = 711$, for the same reason as in the previous example. For

Table 4.1: Estimates from the model and likelihood-based boosting fit to the primary biliary cirrhosis data, as well as the corresponding Lasso and Ridge estimates, which are included for comparison.

CoxBoost	Mboost	Lasso	Ridge	Cox
0	0	0	-0.01156	0.01367
0	0.06829	0	0.04028	0.12447
0	0.13272	0	0.12031	0.18257
0	0.27996	0	0.10474	0.18783
0.20501	0.37054	0.05247	0.18393	0.46761
0.02639	0.02685	0.02295	0.02221	0.02901
0.09453	0.09127	0.09136	0.0767	0.08479
-1.05905	-0.98862	-1.02312	-0.81383	-1.06789
0.00326	0.00285	0.00321	0.00303	0.00291
-0.00001	-0.00001	0	-0.00001	-0.00002
0.00263	0.00311	0.00219	0.00306	0.00368
0.28015	0.28012	0.26151	0.25858	0.31085

the model-based version, we let $\nu = 0.1$ and the maximum number of iterations be $M = 10000$. In figure 4.1, a plot of the cross validated partial log likelihood for the likelihood based approach, and a plot of the cross validated loss based on the Cox proportional hazards model are shown. The optimal number of iterations are chosen as $M_{CoxBoost} = 150$, and $M_{mboost} = 6017$, respectively. The final model for the likelihood-based boosting estimates non-zero coefficients for 18 covariates, while the model-based version estimates 205 of the regression coefficients to be non-zero.

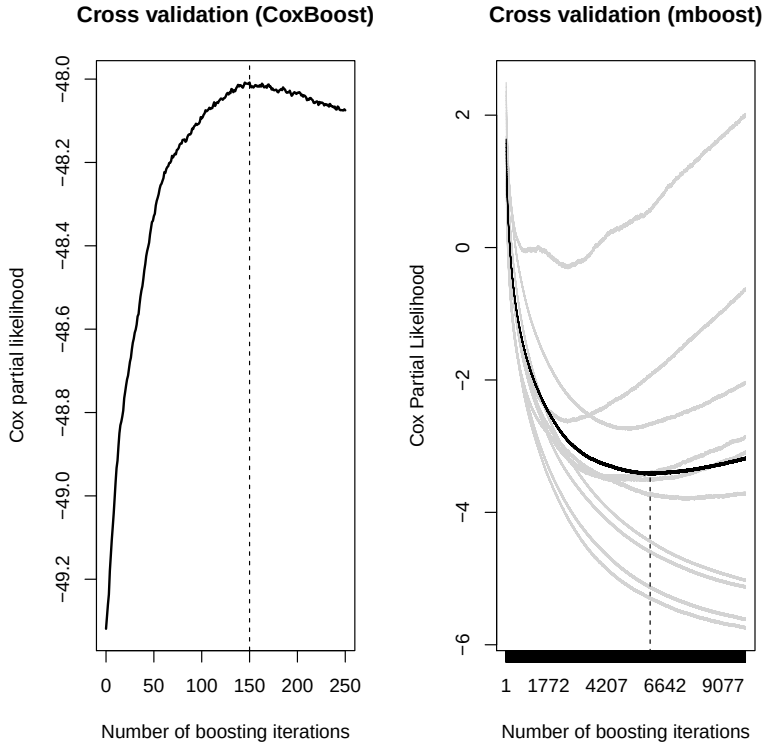

Figure 4.1: Cross validated partial likelihood for *CoxBoost*, and cross validated Cox partial likelihood based loss for *mboost*, computed for the Dutch breast cancer data.

Table 4.2: Estimates from the likelihood-based boosting fit to the Dutch breast cancer data, as well as the corresponding Lasso estimates, which are included for comparison.

Gene no	CoxBoost	Lasso
128	-0.093	-0.102
1925	0.057	0.003
2042	-0.094	-0.099
2242	-0.206	-0.204
2246	-0.198	-0.164
2309	-0.098	0
2363	-0.089	-0.102
2816	0.097	0.092
3154	0.221	0.281
3394	-0.21	-0.237
3822	-0.027	0
4175	-0.161	-0.197
4176	-0.099	-0.141
4197	0.045	0.004
4272	0.062	0.093
4309	0.499	0.545
4331	-0.083	-0.091
4616	0	-0.018
4630	-0.097	0

The estimates from the CoxBoost (likelihood based) model are given in table 4.2 along with estimates from a Lasso model using the same folds as for the CoxBoost fit. From the table, we see that at least for this particular split of the data, the estimates from the likelihood-based boosting are comparable to the Lasso estimates. In addition, both of the methods seem to select almost the same set of covariates. In this case, CoxBoost seems to estimate almost the same amount of coefficients to be non-zero. If we desired a sparser fit, we could have tuned the number of iterations with a smaller penalty parameter. Alternatively, if the penalty parameter was larger, the increments to the coefficients in each iteration would be smaller, and thus we might expect that more coefficients are estimated as non-zero. Most of the estimates that are non-zero for both models are similar in absolute value, although there is some variation. One could think – and hope – that the reason that the mboost procedure selects so many coefficients is that its estimates lie closer to the Ridge estimates on an imaginary continuum between the Lasso and the Ridge, but on inspection, its estimates are actually larger in absolute value than the CoxBoost estimates, and certainly larger than the Ridge estimates. In the interest of further illuminating the latter point of dicussion, it is interesting to investigate what estimates of prognostic index these coefficients correspond to, and to compare these across methods. To do this, we plot histograms of the estimated prognostic index for the CoxBoost, mboost, Lasso and Ridge estimates, which are shown in figure 4.2. We see from the histograms, that the distribution of the prognostic index is quite similar for the Ridge, Lasso and CoxBoost models, and that the mboost has quite a different distribution, which is in line with the fact that the coefficient estimates from the mboost-procedure are somewhat larger than those of the other methods, and that it selects a great number of covariates.

4.4 Extensions to landmarking

In the present section, we seek to extend the above likelihood-boosting scheme to fit sliding landmark models. We do this by replacing the penalised log likelihood by a penalised version of van Houwelingens integrated partial log likelihood (2.8) offset by the current estimate, and then we carry out an argument analogue to the Cox-case. Each covariate now has an associated S number of parameters which we wish to update at the same time. Hence, we must do a multivariate version of the Cox-version taking into account all the landmark coefficients of each variable simultaneously. Firstly, we

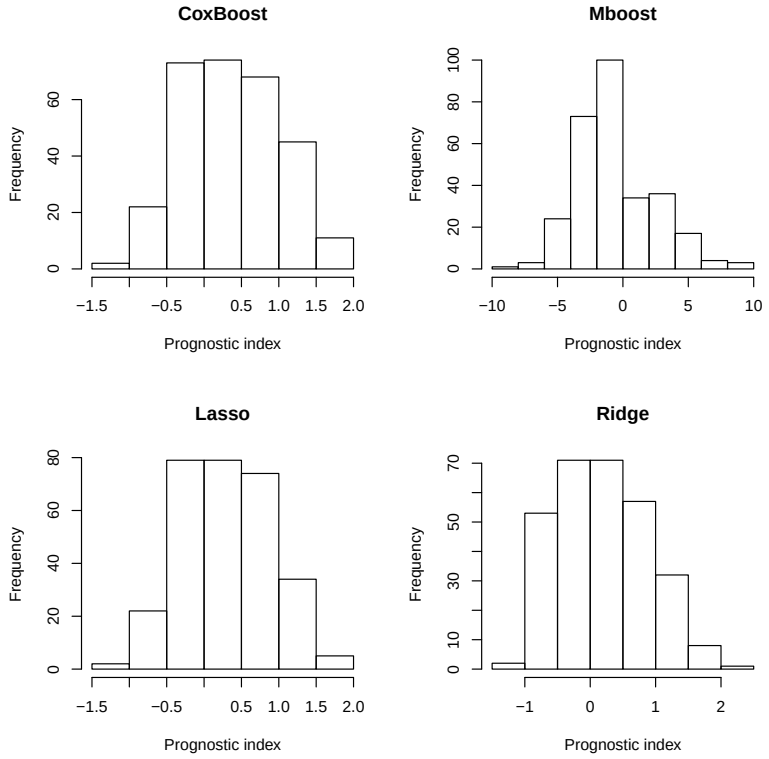

Figure 4.2: Histograms of the progostic indexes for the Dutch breast cancer data estimated by boosting using *CoxBoost* and *mboost*, and by Lasso and Ridge using *glmnet*.

observe that the first and second partial derivative of the penalised integrated partial log likelihood with respect to the g-th landmark coefficient of the j-th covariate evaluated at $\mathbf{0}$ are

$$\frac{\partial ipl\left(\mathbf{0}|\hat{\boldsymbol{\beta}}(\mathbf{LM})\right)}{\partial \beta_j(LM_g)} = \sum_{i \in A_g} d_i \left(x_{ij} - \frac{s_j^{(1)}\left(\boldsymbol{\beta}(LM_g), t_i\right)}{s^{(0)}\left(\boldsymbol{\beta}(LM_g), t_i\right)} \right),$$

and

$$\frac{\partial^2 ipl\left(\mathbf{0}|\hat{\boldsymbol{\beta}}(\mathbf{LM})\right)}{\partial \beta_j(LM_g)^2} = -\sum_{i \in A_g} d_i \frac{s_j^{(2)}\left(\boldsymbol{\beta}(LM_g), t_i\right) s^{(0)}\left(\boldsymbol{\beta}(LM_g), t_i\right) - \left(s_j^{(1)}\left(\boldsymbol{\beta}(LM_g), t_i\right)\right)^2}{\left(s^{(0)}\left(\boldsymbol{\beta}(LM_g), t_i\right)\right)^2}.$$

We will refer to the S-dimensional vector consisting of the first derivatives with respect to each landmark coefficient for the j-th covariate as $\mathbf{u}_j(\mathbf{0}|\hat{\boldsymbol{\beta}}(\mathbf{LM}))$ and the matrix consisting of second derivatives with respect to the landmark coefficients of the j-th

variable as $J_j(\mathbf{0}|\hat{\boldsymbol{\beta}}(\mathbf{LM}))$. One can easily observe that the latter is a diagonal matrix, because each of the components of the first derivative only depends on the coefficients associated with one of the landmarks. Using the multivariate Taylor approximation of the second order to the penalised integrated partial log likelihood

$$\begin{aligned} ipl^{pen}(\mathbf{b}(\mathbf{LM})|\boldsymbol{\beta}(\mathbf{LM})) \approx ipl(\mathbf{0}|\boldsymbol{\beta}(\mathbf{LM})) + \mathbf{b}_j(\mathbf{LM})^T u_j(\mathbf{0}|\boldsymbol{\beta}(\mathbf{LM})) \\ - \frac{1}{2}\mathbf{b}_j(\mathbf{LM})^T \left(J_j(\mathbf{0}|\hat{\boldsymbol{\beta}}(\mathbf{LM})) + \lambda I \right) \mathbf{b}_j(\mathbf{LM}), \end{aligned} \tag{4.2}$$

where $\mathbf{b}_j(\mathbf{LM})$ is the vector of landmark coefficient updates for the j-th covariate, we can derive the updates by differentiating with respect to $\mathbf{b}_j(\mathbf{LM})$ and solving the resulting equation. This predictably yields that the updates to the coefficients of the j-th variable are

$$\mathbf{b}_j(\mathbf{LM}) = \left(J_j(\mathbf{0}|\hat{\boldsymbol{\beta}}(\mathbf{LM})) + \lambda I \right)^{-1} u_j(\mathbf{0}|\boldsymbol{\beta}(\mathbf{LM})). \tag{4.3}$$

In situations where computing the value of the partial log likelihood p times for each boosting iteration is too costly, certainly the same will be true for van Houwelingens integrated partial log likelihood. Therefore, we will use the same approximation as the one we used to derive the coefficient updates to derive a scoring measure, as we did in the Cox setting. Inserting the updates (4.3) into the right hand side of (4.2) we get the expression

$$\begin{aligned} & ipl(\mathbf{0}|\boldsymbol{\beta}(\mathbf{LM})) + u_j(\mathbf{0}|\boldsymbol{\beta}(\mathbf{LM}))^T \left(J_j(\mathbf{0}|\hat{\boldsymbol{\beta}}(\mathbf{LM})) + \lambda I \right)^{-1} u_j(\mathbf{0}|\boldsymbol{\beta}(\mathbf{LM})) \\ & - \frac{1}{2} u_j(\mathbf{0}|\boldsymbol{\beta}(\mathbf{LM}))^T \left(J_j(\mathbf{0}|\hat{\boldsymbol{\beta}}(\mathbf{LM})) + \lambda I \right)^{-1} u_j(\mathbf{0}|\boldsymbol{\beta}(\mathbf{LM})) \\ = \; & ipl(\mathbf{0}|\boldsymbol{\beta}(\mathbf{LM})) + \frac{1}{2} u_j(\mathbf{0}|\boldsymbol{\beta}(\mathbf{LM}))^T \left(J_j(\mathbf{0}|\hat{\boldsymbol{\beta}}(\mathbf{LM})) + \lambda I \right)^{-1} u_j(\mathbf{0}|\boldsymbol{\beta}(\mathbf{LM})). \end{aligned}$$

Since the first term is the same for all covariates, and the constant $\frac{1}{2}$ does not matter when selecting the largest index, we arrive at the scoring measures

$$W^{(j)} = u_j(\mathbf{0}|\boldsymbol{\beta}(\mathbf{LM})^T \left(J_j(\mathbf{0}|\hat{\boldsymbol{\beta}}(\mathbf{LM})) + \lambda I \right)^{-1} u_j(\mathbf{0}|\boldsymbol{\beta}(\mathbf{LM})).$$

We then use these to select which covariate to update the landmark coefficients in each iteration. I.e., they are updated as

$$\boldsymbol{\beta}_{j^*}^{(m)}(\mathbf{LM}) = \boldsymbol{\beta}_{j^*}^{(m-1)}(\mathbf{LM}) + \mathbf{b}_{j^*}(\mathbf{LM})$$

and

$$\boldsymbol{\beta}_j^{(m)}(\mathbf{LM}) = \boldsymbol{\beta}_j^{(m-1)}(\mathbf{LM}), \quad j \neq j^*,$$

where $j^* = \underset{j}{\text{argmax}}\ W^{(j)}$. Van Houwelingens integrated partial log likelihood is sometimes referred to as *ipl* for short, and inspired by this we will sometimes refer to the algorithm above by the name *IplBoost*, for the sake of brevity. A prototype implementation that computes landmark estimates using likelihood boosting of van Houwelingens integrated partial likelihood is available through the package *IplBoost*, currently available for download at *https://github.com/simbrant/IplBoost*.

4.4.1 Boosted landmarking applied to the primary biliary cirrhosis data

As for the boosted Cox regression, we begin with a simpler low dimensional example, before we tackle a more high dimensional dataset, namely the primary biliary cirrhosis data previously discussed in section 3.2.2 and section 4.3.1. In addition to being low-dimensional, this dataset also has more heterogenous covariates than for instance the Dutch breast cancer dataset we have previously discussed, which requires us to standardise the covariates. Before we fit the landmark model, we must decide which landmarks to use, and how wide landmark intervals we wish to use. In addition, we have to decide upon the values of the penalty parameters in the algorithm. The survival times in this dataset are measured in days, the maximum follow-up time being $t = 4556$ days. For the purpose of this analysis, we convert these to years, so that the maximium follow up time is just short of 12.5 years. Therefore, we choose a grid of 81 equidistant points on the interval $[0, 8]$ as our landmarks, and an interval width of $w = 3$ years. To fit the landmark model we use the likelihood-boosted landmark algorithm outlined in section 4.4. In this algorithm, we can set individual values of the penalty parameter for each landmark point. We could set it to the same value for each landmark, but this might not be the best solution. It is reasonable that a better solution is to let the size of the penalty parameters depend upon the amount of data available to estimate the coefficients at the corresponding landmark, and can therefore be set to depend on the number of events in each landmark data set. For this reason, we will choose a strategy where we let the penalty parameter for each landmark point depend upon the number of events in the corresponding landmark interval in the same

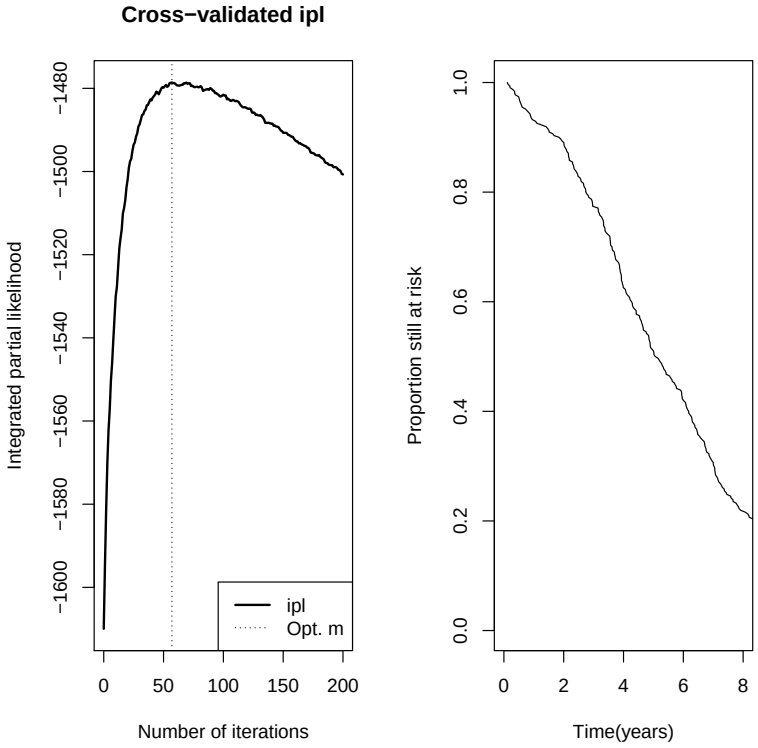

Figure 4.3: The values of the cross-validated integrated partial log likelihood for each iteration for the primaty biliary cirrhosis data, in addition to a plot of the proportion of individuals still at risk.

manner as for the likelihood-boosted Cox-regression analysis of the same dataset. That is, the penalty for the s-th landmark point is 9 times the number of events between LM_s and $LM_s + w$. Subsequently, we choose the number of iterations via 10-fold cross validation using the same folds as for the analysis in section 4.3.1, such that the estimates are comparable. In this case, for the penalty parameters described above, the optimal number of iterations is estimated to be $M_{IplBoost} = 57$. For reference, a plot of the cross-validated ipl is given in figure 4.3. The algorithm selects the same 8 effects that were also selected by *CoxBoost*. A rendition of these are given in figure 4.4, where they are drawn together with a dotted line denoting the value of the corresponding CoxBoost estimates for comparison. From these plots we see that the estimates from the IplBoost algorithm seem to correspond fairly well to the CoxBoost estimates in that they lie relatively close to them. In addition, most of the landmark coefficients

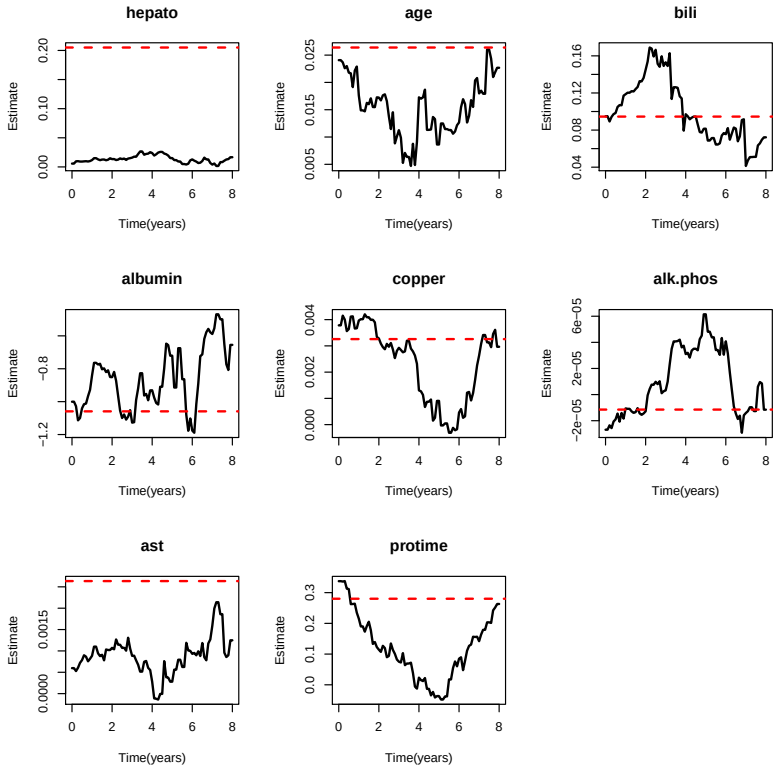

Figure 4.4: The estimated coefficient curves for the primary biliary cirrhosis data.The title of each plot of the coefficients denotes the covariate for which the estimated effects are drawn. The dotted lines correspond to the CoxBoost-estimates of the effects of the same covariate.

cross the corresponding effects estimated by CoxBoost. To see how well the IplBoost algorithm does compared to the CoxBoost and the lasso when it comes to making dynamic 3-year survival predictions for this dataset, we fit models on a subset of the available data and try to predict the 3 year survival probabilities at each landmark for each model for the remaining data. These predictions are then evaluated using the dynamic Brier score and the related dynamic R^2 that were presented in section 2.4.2. The plots of these scores for the IplBoost, CoxBoost and lasso models are shown in figure 4.5. Here we see that the predictions made using the CoxBoost and lasso models almost make an identical error. More imortantly, we see that for the most part of the first four years, the IplBoost seems to make the most accurate prognosis. For the next two years however, the CoxBoost and lasso estimates are more accurate, but the

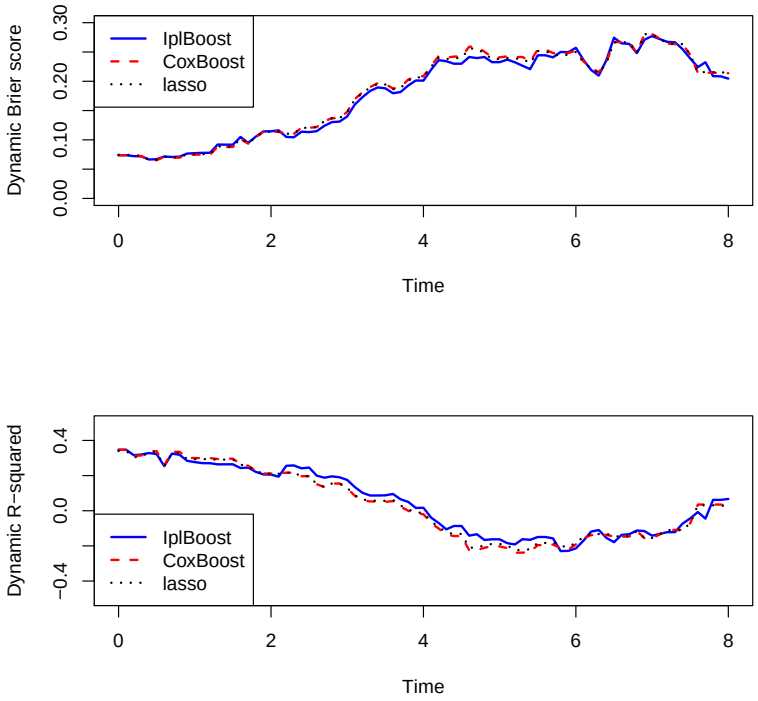

Figure 4.5: Dynamic Brier score, and dynamic R^2 computed for the primary biliary cirrhosis data using estimates computed by IplBoost, CoxBoost and lasso.

IplBoost are more accurate on the whole, in terms of average dynamic Brier score. One could also make the remark that there are less events later in the study, which therefore also makes the later landmark estimates, and also the evaluated performance of these, more unstable. The Brier scores will later in the study be, as it were, inherently more random that earlier in the study. Due to this increasing of the variance of the Brier scores as time passes, we should put more weight on the prognoses made at the earlier landmarks where more data is used both for estimation and validation. Thus it seems that for this particular dataset, for this particular division into training and test cases IplBoost seems to outperform CoxBoost when it comes to making dynamic survival predictions.

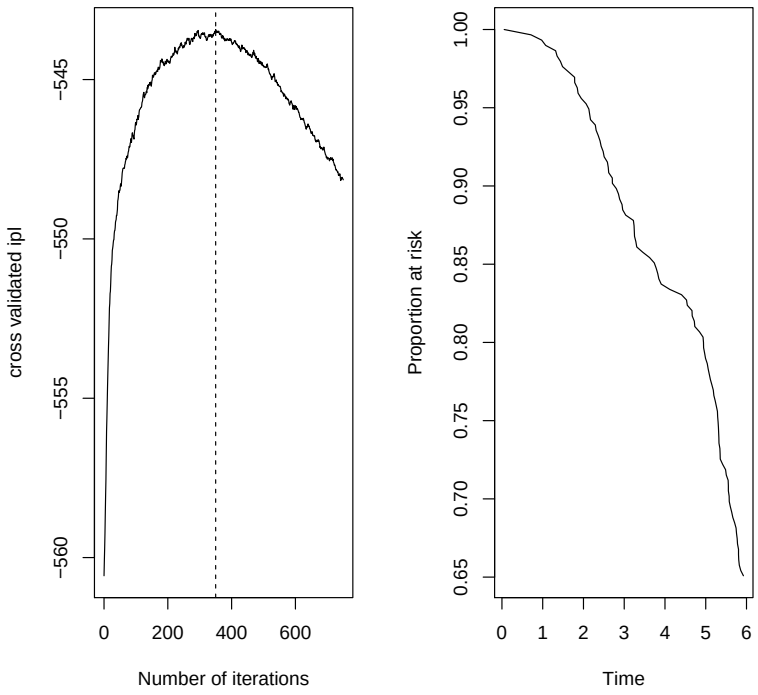

Figure 4.6: The values of the cross-validated integrated partial log likelihood for each iteration, and a plot of the proportion of the individuals that are still at risk for the Dutch breast cancer data.

4.4.2 Boosted landmarking applied to the Dutch breast cancer data

To further illustrate the IplBoost algorithm outlined in section 4.3.1, we will try compute landmark estimates for the Dutch breast cancer data. To define the model, we choose 31 equidistant landmarks on the interval $[0, 6]$, and an interval width of $w = 3$ years. We here choose to not standardise the covariate vectors as previously discussed in section 3.2.7, and employ the same strategy when selecting the penalty parameters as described in the previous example. The number of iterations is selected using 10-fold cross-validation using the same folds as in section 4.3.2, where a plot of the cross-validated ipl is given in figure 4.6. The selected number of iterations is in this case $M_{IplBoost} = 350$, and for this model 42 effects are selected. Of the 18 effects that CoxBoost selects, 14 are selected. These are all shown in figures 4.7 and 4.8 where the CoxBoost estimate is drawn as a dotted line for comparison. Plots of the remaining 28

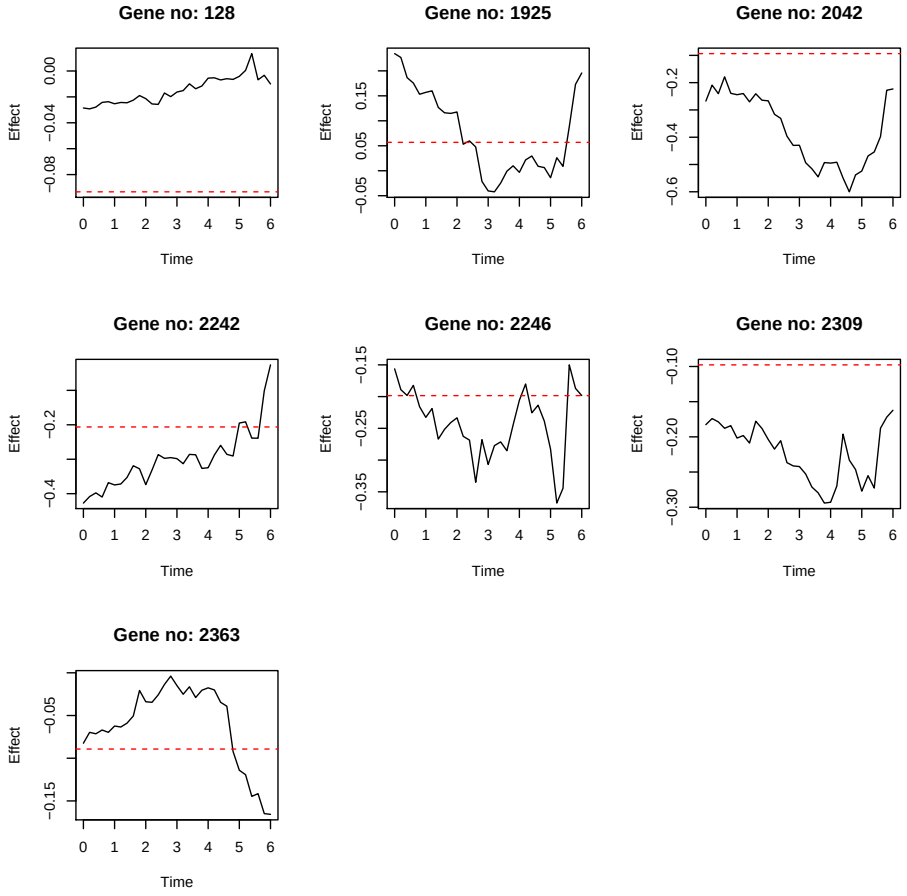

Figure 4.7: The estimated coefficient curves for 7 of the covariates that are selected by both IplBoost and by CoxBoost for the Dutch breast cancer data.

selected effects are given in figures (C.1), (C.2) and (C.3). The estimtates of the coefficients that are selected both by the CoxBoost algorithm and the IplBoost algorithm are somewhat diverse in nature. Some are larger or smaller in absolute value overall, and some cross the CoxBoost estimate. The ones that are only selected by IplBoost also display varying characteristics. Some are increasing in absolute value, some are decreasing, some of them cross 0, and some seem to be more stable than others. The resulting model from this analysis is less sparse than the CoxBoost model in that it selects 49 coefficients instead of the 18 covariates, but still relatively few compared to the 4919 avaiable covariates to choose from. The number of iterations is quite high, and if we desired a more sparse model that also took a shorter time to fit, we might set

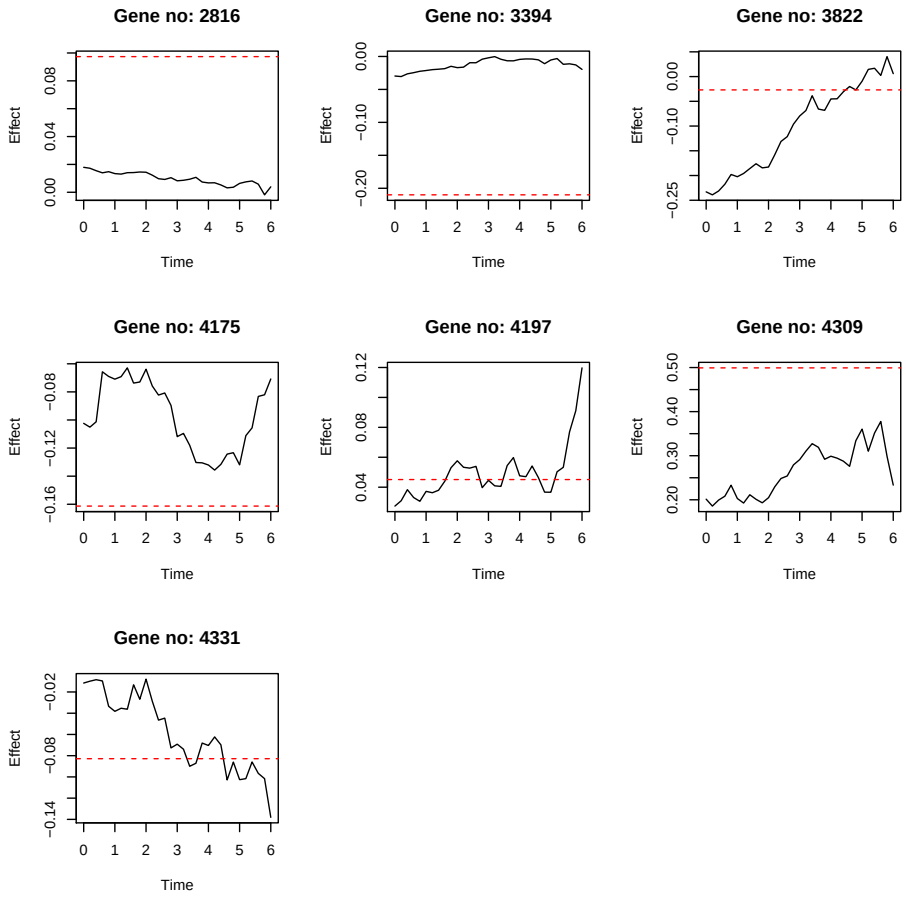

Figure 4.8: The estimated coefficient curves for 7 of the covariates that are selected both by IplBoost and by CoxBoost for the Dutch breast cancer data.

the penalties to be smaller. It could also be that putting more weight on the landmark datasets with fewer events by penalising them less is not the best idea. Perhaps we should have set the penalty to be the same for all landmark datasets, so that we shrink the landmark coefficients more where there is less available information, rather than the penalties being proportional to the number of events. To evaluate the quality of the predictions the algorithm makes, we again use the dynamic Brier score, and the dynamic R^2 measure based on it, as in the example in the previous section. We fit a model using the IplBoost algorithm to two thirds of the observations, using the same landmarks and penalty strucure as we did above. In addition, we fit models using both the lasso and CoxBoost algorithms to the same data for comparison. These two

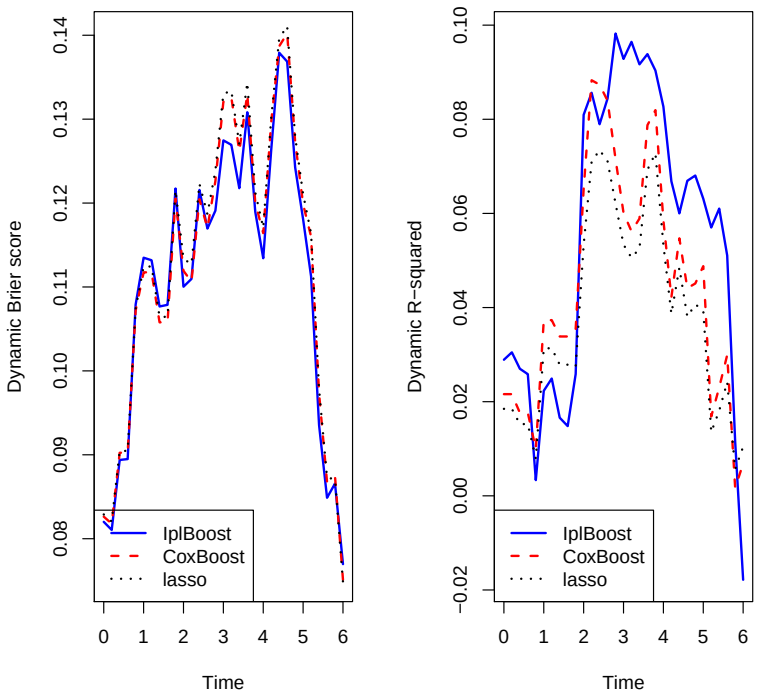

Figure 4.9: Dynamic Brier score, and dynamic R^2 computed for the Dutch breast cancer data using estimates computed by IplBoost, CoxBoost and lasso.

measures are plotted as curves, which are given in figure 4.9. From the plot of the Brier scores it is hard to discern a difference in terms of the error made by the three algorithms, they seem to be quite similar apart from some minor differences. These small differences are more pronounced when the Brier scores are translated into the dynamic R^2 measure, due to the scale of the R^2 scores, but all the three curves are very close together. The IplBoost estimates seem to be slighly more accurate in the two first years than the lasso and the Coxboost estimates, and somewhat less accurate from the fifth year and onwards. From the second to the fifth year, the IplBoost estimates seem to be more accurate than the lasso estimates, but less accurate than the CoxBoost estimates. Overall, the three methods are quite similar in term of predictive performance. The Coxboost is perhaps the best overall, while the IplBoost is better at the start of the study, and the lasso is slightly better at the end. As there are quite few events in the dataset, there is a high variance both in the estimates and the estimated test error, so it

is hard to say if any of these methods are superior to the other. To study how well the IplBoost method works, so to speak, we will instead study it in an artificial setting. By this we mean that we define a model that data could be generated from, simulate a set of observations from these, and then see how well the algorithm performs in terms of estimating regression coefficients and making predictions for unseen data.

Chapter 5

Simulations

In many settings related to survival analysis it can be difficult, indeed sometimes impossible, to determine by calculation if a given method of estimation works, and how well a given method works. By 'working' we here refer to theoretical properties such as consistency, i.e. that the model estimates will approach the 'true' values, under the assumption that the model is correctly specified, as we get more data. In these cases where theoretical computations are difficult or impossible without imposing unrealistic conditions on the problem, we can instead try to generate data where we know what the 'ground truth' is, and then apply our method to see how well it behaves.

5.1 Generating data

To generate survival data, one must first define the hazard from which the survival times will be drawn. Since we in this thesis are discussing the Cox proportional hazards regression model, and extensions of this model, the hazards we are interested in are of the form

$$\alpha(t|\mathbf{x}) = \alpha_0(t)\exp(g(t, \boldsymbol{\theta}, \mathbf{x})),$$

where g is a function that describes both the time-varying and time-constant effects. When we analyse real data, we assume that the baseline hazard $\alpha_0(t)$ is just some arbitrary function that is common to all of the individuals in the sample. Now, however, we are generating artificial data to test how well a given method of estimation works when we know the true generating mechanism, and thus we must explicitly define the

baseline hazard. Popular choices include $\alpha_0(t) = k$, $\alpha_0(t) = kt^{k-1}$, and $\alpha_0(t) = \exp(kt)$, for some real number k. In addition, it is convenient to define the hazard in such a way that we can find a closed form expression for the cumulative hazard $A(t|\mathbf{x}) = \int_0^t \alpha(s|\mathbf{x})ds$. Since there is a one-to-one correspondence between the cumulative hazard and the cumulative distribution function, we can then find an explicit expression for F^{-1}, the inverse of the cumulative distribution function. By the *inverse probability integral transform*, we have that if $U \sim \text{Uniform}([0, 1])$, then $T = F^{-1}(U) \sim F$, if F is the cumulative distribution function of T. Thus, we may simulate values of T by drawing values from $U \sim \text{Uniform}([0, 1])$, and transforming these via F^{-1}, which is known as the method of inversion [Devroye, 1986]. Since the methods we outline in this thesis are intended to extend the Cox model to accommodate for time-varying effects, our simulation models should reflect this. Thus, it is sensible for us to design models from which to draw samples that have both time-constant and time-varying effects. Therefore, we may describe the models we want to consider in our studies by a hazard function of the form

$$\alpha(t|\mathbf{x}, \mathbf{z}) = \alpha_0(t) \exp(\boldsymbol{\beta}^T \mathbf{x} + \boldsymbol{\gamma}(t)^T \mathbf{z})$$

Since, for computational reasons, we want to be able to find an expression for F^{-1}, and thus also analytically solve the integral of the hazard function from 0 up to t, the functions we use to represent time-varying effects must reflect this constraint. To generate a sample of survival data, we first draw uniformly distributed random variables and values of the covariates from some distribution to simulate the survival times. Then, we draw censoring times from some distribution (exponential, Weibull, etc.), or alternatively a censoring indicator from a Bernoulli distribution, and right-censor at some t_{end}.

5.1.1 Models with constant effects

Before we get to grips with finding a working model with time-varying effects, it is natural to first consider some simpler models with only time-constant effects. In this case, the hazard takes on the form

$$\alpha(t|\mathbf{x}) = \alpha_0(t) \exp(\boldsymbol{\beta}^T \mathbf{x}),$$

and thus all models of this have corresponding cumulative hazards that can be written as

$$A(t|\mathbf{x}) = \exp(\boldsymbol{\beta}^T\mathbf{x}) \int_0^t \alpha_0(s)ds = \exp(\boldsymbol{\beta}^T\mathbf{x})A_0(t).$$

Writing $F(t|\mathbf{x}) = 1 - \exp(-A(t|\mathbf{x}))$, and solving for t, we can see that, provided that we can find an expression for A_0^{-1}, survival times can be simulated from any model with time-constant effects via the formula

$$T = A_0^{-1}\left(-\exp(-\boldsymbol{\beta}^T\mathbf{x})\log(1-U)\right),$$

where U is drawn from a uniform distribution on the unit interval. If we consider the three examples mentioned above arising from the exponential, Weibull and Gompertz distributions, these have corresponding cumulative hazards $A_0(t) = kt$, $A_0(t) = t^k$, and $A_0(t) = \frac{1}{k}(\exp(kt) - 1)$, respectively. The inverse functions of these are $A_0^{-1}(y) = \frac{y}{k}$, $A_0^{-1}(y) = y^{\frac{1}{k}}$, and $A_0^{-1}(y) = \frac{1}{k}\log(ky - 1)$, and thus we the formulas from which we can generate survival times from models with these baseline hazards are

$$\begin{aligned}
T &= -\frac{1}{k}\exp(-\boldsymbol{\beta}^T\mathbf{x})\log(1-U), \\
T &= \left(-\exp(-\boldsymbol{\beta}^T\mathbf{x})\log(1-U)\right)^{\frac{1}{k}}, \text{ and} \\
T &= \frac{1}{k}\log\left(-k\exp(-\boldsymbol{\beta}^T\mathbf{x})\log(1-U)+1\right).
\end{aligned}$$

5.1.2 A class of models with time-varying effects

As we saw in the previous section, the time-constant effects do not make the problem of computing and inverting the cumulative distribution function any harder, as they only contribute to the hazard by a constant. For time-varying effects, this is not the case, and we have to choose the form of these effects in such a way that we are able to find an expression for the cumulative distribution function that we can invert. When time-varying effects occur in observed data, they are often decreasing with time, and thus we want to find a model that has this property. An example of a description of time-varying effects that has these properties is

$$\boldsymbol{\gamma}(t) = \boldsymbol{\gamma}_c(t_u - t),$$

where $\boldsymbol{\gamma}_c$ is a q-dimensional vector of constants, and t_u is some positive real number that is greater than the largest survival time, or larger than a timepoint where all survival times are right-censored. We will consider a class of models with time-varying effects on this form, in addition to a set of time-constant effects, with hazard functions

$$\alpha(t|\mathbf{x}, \mathbf{z}) = \alpha_0(t) \exp(\boldsymbol{\beta}^T \mathbf{x} + \boldsymbol{\gamma}_c^T \mathbf{z}(t_u - t)).$$

By moving all constants outside the integral, we can see that all such models have cumulative hazard functions that can be written in the form

$$A(t|\mathbf{x}, \mathbf{z}) = \exp(\boldsymbol{\beta}^T \mathbf{x} + \boldsymbol{\gamma}_c^T \mathbf{z} t_u) \int_0^t \alpha_0(s) \exp(-\boldsymbol{\gamma}_c^T \mathbf{z} s) ds,$$

and thus the integral we need to solve in each case is

$$\int_0^t \alpha_0(s) \exp(-cs) ds.$$

We will consider each of the shapes of the baseline hazard previously mentioned. For the case of $\alpha_0(t) = k$, the above integral is

$$\int_0^t k \exp(-cs) ds = \frac{k}{c} \left(1 - \exp(-ct)\right).$$

The case where $\alpha_0(t) = \exp(kt)$ is also straight-forward, and the solution of the integral is

$$\int_0^t \exp(ks) \exp(-cs) ds = \frac{1}{k - c} \left(\exp([k - c]\, t) - 1\right).$$

A more complicated situation arises when we let the baseline hazard assume the form $\alpha_0(t) = kt^{k-1}$, because the integral that corresponds to this cannot be expressed in closed form. However, by making a simple subsitution we see that we can express the integral as

$$\int_0^t k s^{k-1} \exp(-cs) ds = \frac{k}{c^k} \gamma\left(k, ct\right),$$

where γ is the lower incomplete gamma-function

$$\gamma\left(k, x\right) = \int_0^x u^{k-1} \exp(-u) du.$$

There does, however, exist software to evaluate this function and the corresponding inverse function, so we can still use this model to draw survival times. By writing $F(t|\mathbf{x}, \mathbf{z}) = 1 - \exp(-A(t|\mathbf{x}, \mathbf{z}))$, inserting the solution of the integrals above, and solving for t, we can find formulas that we can use to generate survival times for each of these cases. For $\alpha_0(t) = k$, we find that this formula is

$$T = -\frac{1}{\boldsymbol{\gamma}_c^T \mathbf{z}} \log\left(1 + \frac{\boldsymbol{\gamma}_c^T \mathbf{z}}{k \exp\left(\boldsymbol{\beta}^T \mathbf{x} + \boldsymbol{\gamma}_c^T \mathbf{z} t_u\right)} \log(1-U)\right).$$

An issue with this model is that since $\log(1-U) < 0$, we must have $\boldsymbol{\gamma}_c^T \mathbf{z} < 0$. For this reason we must reject those observations where this is the case. The formula for the model with $\alpha_0(t) = \exp(kt)$ is

$$T = \frac{1}{k - \boldsymbol{\gamma}_c^T \mathbf{z}} \log\left(1 - \frac{k - \boldsymbol{\gamma}_c \mathbf{z}}{\exp\left(\boldsymbol{\beta}^T \mathbf{x} + \boldsymbol{\gamma}_c^T \mathbf{z} t_u\right)} \log(1-U)\right),$$

where we must have $\boldsymbol{\gamma}_c \mathbf{z} < k$. For the last case, when $\alpha_0(t) = kt^{k-1}$, the corresponding formula from which we can generate survival times is

$$T = \frac{1}{\boldsymbol{\gamma}_c^T \mathbf{z}} \gamma^{-1}\left(k, -\frac{\left(\boldsymbol{\gamma}_c^T \mathbf{z}\right)^k}{k \exp\left(\boldsymbol{\beta}^T \mathbf{x} + \boldsymbol{\gamma}_c^T \mathbf{z} t_u\right)} \log(1-U)\right),$$

where $\gamma^{-1}(k, \cdot)$, the inverse of the lower incomplete gamma function, must be computed numerically. The latter is, however, not too difficult as it can be shown that $\gamma^{-1}(k, yt) = G^{-1}(\frac{y}{\Gamma(k)})$, where G is the cumulative distribution function of a gamma distributed variable with scale parameter $\beta = 1$, and shape parameter $\alpha = k$. Therefore it is easy to implement this using existing software in R.

5.2 A simulation study of likelihood-boosting in landmark models

In this thesis we are designing a new method of estimation, which aims to estimate and detect time-varying effects in high dimensional survival data. As previously discussed, this is done by combining van Houwelingen's concept of landmarking with that of likelihood boosting by Tutz and Binder [2006]. We are obviously interested in how this method performs, and in such a situation we can do one of two things to clarify

how well a given method works. We can either analyse the method from a theoretical point of view, or we can simulate data from a model and try to use our method in this setting where we know the true data-generating mechanism as discussed above. We will rely on the latter, as the former similarly requires an assumed truth, and the nature of the both the model and the method of estimation is quite complicated, which makes the analysis infeasible. As a starting point, we will first study likelihood boosting in Cox-regression for a model that has no time varying effects.

5.2.1 Likelihood boosted Cox regression

As mentioned above, we will initially study likelihood boosting in Cox regression via simulation. To this end, we simulate datasets and fit models to these using the CoxBoost software, with varying parameters. All the datasets are simulated using an exponential baseline hazard $\alpha_0(t) = k$, with parameter $k = 6$ and 10 time-constant effects that all are set to $\beta = \log(3)$. Censoring times are drawn from an exponential distribution with rate $\lambda = 0.2$. In addition, all observations are right-censored at $t = 1.5$. All covariates are drawn from a uniform distribution on $[-1, 1]$, where 0, 10, 50 or 200 additional variables are simulated, with $n = 400$ or $n = 1600$ total observations. This amounts to drawing 8 different datasets, to which we then fit Cox proportional hazards models using CoxBoost. These experiments are all repeated 100 times, using different seeds to the random number generator. The results of the simulations are summarised in figure 5.1, where we show how different measures depend on the number of covariates in the dataset for the two sample sizes. From the figures, we see that the accuracy of the estimates of the effects for those covariates that influence the survival times decreases when adding more covariates with zero effects, and that this error is smaller for the dataset with more observations. The average number of the variables with zero effects that are estimated to have nonzero effects increases with the number of such variables for the datasets with the least amount of observations. For the datasets with most observations, less incorrect covariates are selected for the datasets with 200 additional covariates than for the datasets with 50 additional covariates. In addition, the average mean squared error of the zero effects decreases as the number of zero effects increases. The latter is, however, averaged over all the zero covariates. If we instead sum the mean squared errors these are slightly increasing with the number of zero covariates. I.e., when more noise is added to the dataset in terms of extra variables

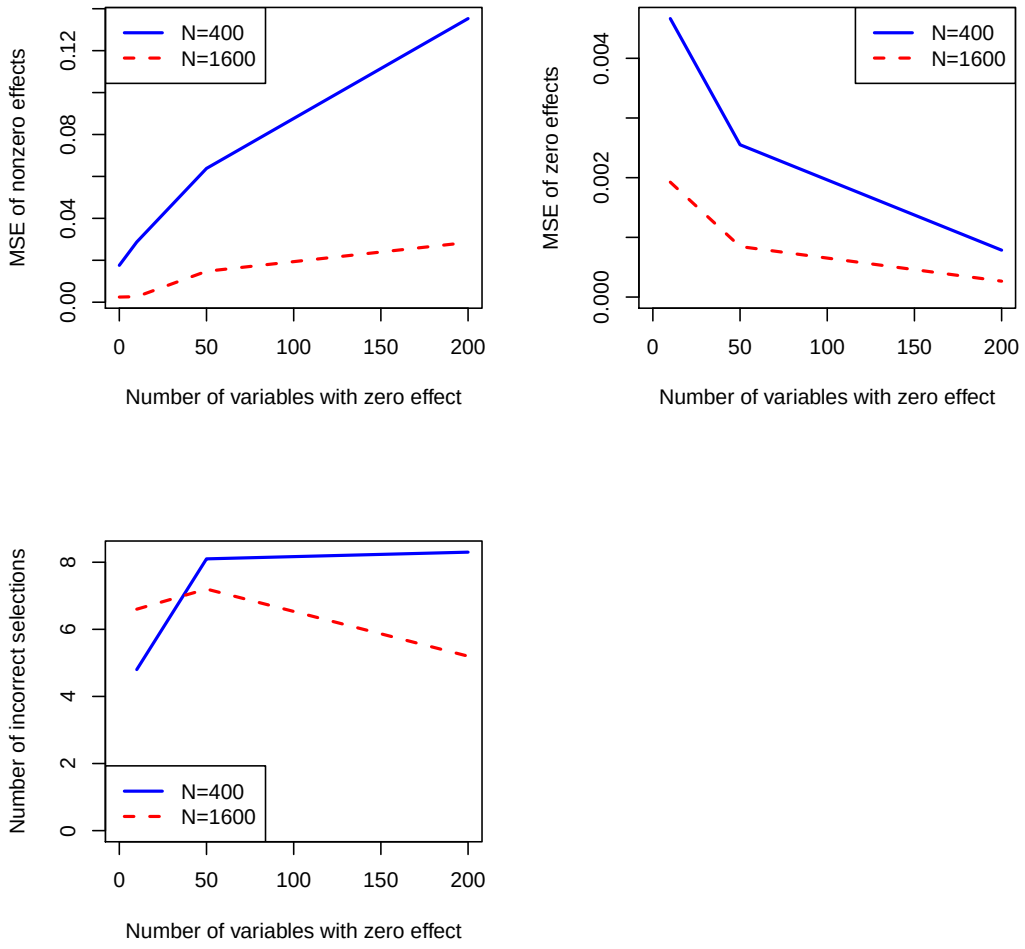

Figure 5.1: Plots illustrating how the CoxBoost procedure copes with the presence of covariates that do not have an effect on the survival. This is measured in terms of average mean squared error for both the covariates with non-zero and zero effects, and the number of covariates with zero effects that are estimated as non-zero.

that are uncorrelated with the survival times, a larger portion of the estimated effects consists of noise and not the true signal, as it were.

5.2.2 Likelihood boosted sliding landmark models

We continue with a simulation where we now introduce time-varying effects, and fit sliding landmark models using the likelihood boosting scheme introduced in section 4.4. In the simulation examples we consider, we generate the data using $p = 5$ time-constant effects and $q = 5$ time-varying effects from a model of the kind discussed in section 5.1.2 where we let the baseline be of the Weibull kind. I.e., we let $\alpha_0(t) = kt^{k-1}$, and set the baseline parameter to be $k = 4$. Similarly to the example above, we here also simulate datasets with 400 and 1600 observations, and 0, 10, 50, or 200 additional covariates that are independent of the survival times. All observations are censored at time $t = 1.5$. In addition to that, censoring times are drawn from an exponential distribution with rate $\lambda = \frac{1}{3}$. The parameters that define the time-varying effects are set to $t_u = 0.7$, and $\gamma(t)$ is chosen so that

$$\frac{\int_0^{1.5} \exp(\gamma(s))ds}{\int_0^{1.5} ds} = 1.5,$$

where $\gamma(t) = \gamma_c(t_u - t)$. The time-constant effects are all set to $\beta_j = \log(1.5)$. The landmark model is defined using 11 equidistant landmarks on the interval $[0, 1]$, with an interval width of $w = 0.25$. In an analysis of a dataset, we would of course use a denser grid of landmarks, but here it should suffice with a coarse grid to illustrate how the method works. These 8 different simulation examples are repeated 100 times each, using different seeds for the random number generator. In order to summarise the results, we compute the average of the estimated time-constant effects, time-varying effects and the estimated effects of the noise variables for each example. These are then drawn together with each of the estimated coefficient curves (as gray lines), and the true effect (as dotted red lines). For the time-varying effects however, the true effects are not what the estimates should converge towards. In fact, we know from section 2.3.1 that they should approximately be of the form (2.4), integrating over the s-th landmark interval instead of $[0, t]$. I.e., the g-th landmark estimate of a coefficient with

a time-varying effect should approximately equal

$$\frac{\int_{LM_g}^{LM_g+w} \alpha_0(s)\gamma(s)ds}{\int_{LM_g}^{LM_g+w} \alpha_0(s)\ ds},$$

which, in terms of the model in section 5.1.2 with baseline $\alpha_0(t) = kt^{k-1}$, is

$$\frac{\int_{LM_g}^{LM_g+w} ks^{k-1}\gamma_c(t_u - s)ds}{\int_{LM_g}^{LM_g+w} ks^{k-1}ds}.$$

If we compute the integrals and tidy up the resulting expression we can see that this becomes

$$\gamma_c \frac{\left[t_u - \frac{k(LM_g+w)}{k+1}\right](LM_g + w)^k - \left[t_u - \frac{kLM_g}{k+1}\right](LM_g)^k}{(LM_g + w)^k - (LM_g)^k}.$$

This is what we should compare the estimates of the 5 time varying effects to, instead of the true effects. For this reason, the true time-varying effects are not drawn for comparison in the plots of the estimates of the time-varying effects, but instead the curve defined by the expression above. The plots of the constant, varying and zero effects are given in the figures 5.2, 5.3, and 5.4, respectively. The image of the comparisons of the estimates in figure 5.3 to the approximations defined by (2.4) is quite striking in terms of the discrepancy between the estimates and what they should approximately converge to. However, if we look at the derivation of this approximation given in appendix B.2, we see that a premise for the approximation to be valid is that there are few events and few censorings in each landmark interval, compared to the number of individuals at risk. This condition cannot be said to hold true for our simulation example, and it may also be that the coefficients decrease too rapidly for this approximation to be true, as the coefficients cannot vary to much over the interval. The comparison of this approximation to the estimates is however a more relevant one than the true value.

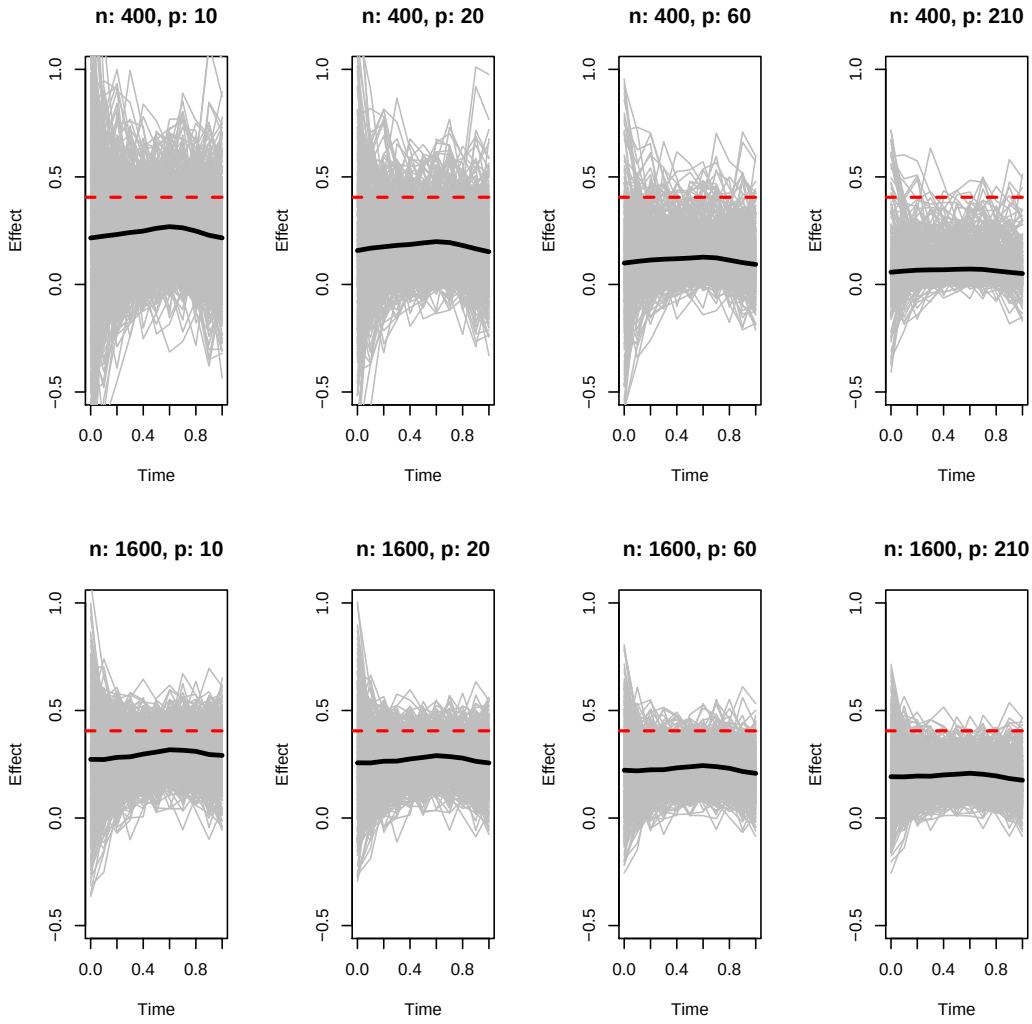

Figure 5.2: Plot of the estimated coefficient curves of the variables with constant effects. The average of all the estimates coefficient curves are drawn in a bold black line, while the dotted line denotes the true effect.

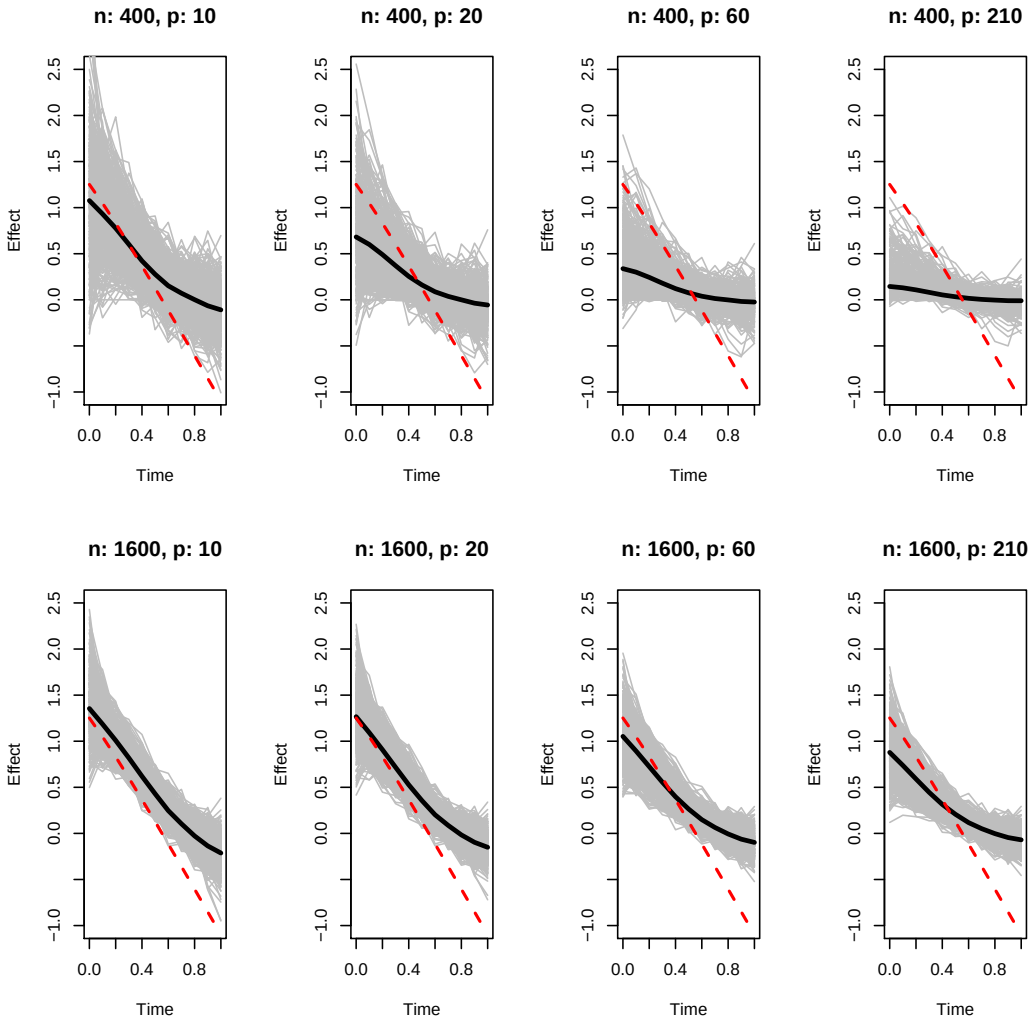

Figure 5.3: Plot of the estimated coefficient curves of the variables with time-varying effects. The average of all the estimates coefficient curves are drawn in a bold black line, while the dotted line is defined by the approximation in (2.4)

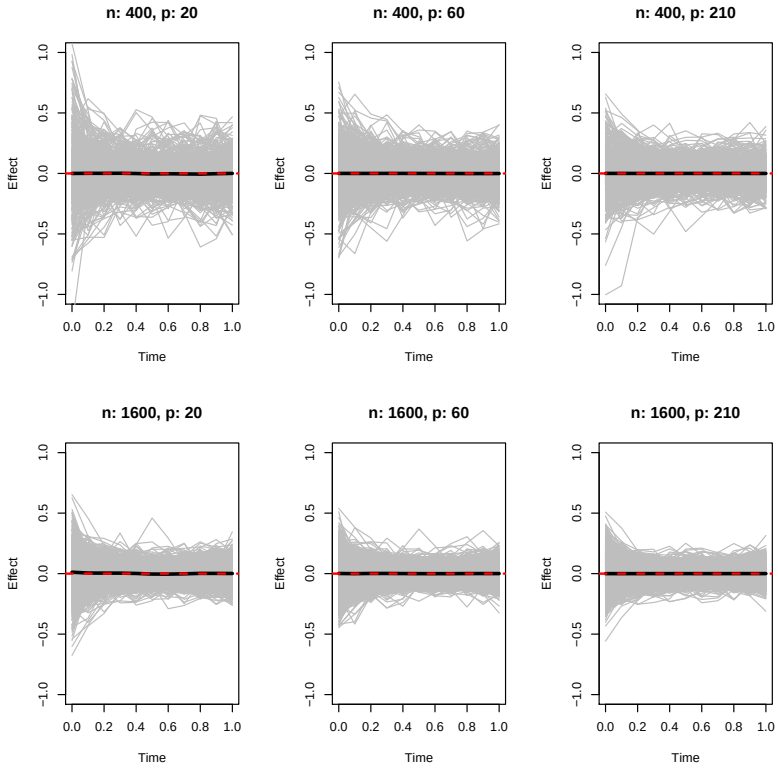

Figure 5.4: Plot of the estimated coefficient curves of the variables with no effect on the survival times. The average of all the estimates coefficient curves are drawn in a bold black line, while the dotted red line denoted the true effect $\beta_j = 0$.

Table 5.1: Table that displays the average mean squared error for the covariates with effects that are constant, time-varying and zero, the average number of coefficients that are correctly incorrectly estimated as non-zero for each of the models in the IplBoost simulation.

p	n	MSE Constant	MSE Varying	MSE zero	num correct	num wrong
10	400	0.0817	0.3000	0.0000	9.98	0.00
20	400	0.0860	0.3393	0.0140	9.58	7.44
60	400	0.1046	0.4289	0.0028	8.10	16.98
210	400	0.1265	0.5019	0.0006	5.75	20.25
10	1600	0.0303	0.2318	0.0000	10	0.00
20	1600	0.0339	0.2266	0.0086	10	8.38
60	1600	0.0439	0.2339	0.0025	10	17.57
210	1600	0.0544	0.2576	0.0007	10	21.74

It can be tricky to judge small differences across the different plots, therefore a table

(Table 5.1) where the average mean squared error for the covariates that have constant, time-varying and zero effects are shown for each model, as well as the average number of effects that are correctly and incorrectly estimated as non-zero. We see from the table that when the number of covariates increases the overall error of the constant and time-varying effects seems to increase, and the average error of the zero effects seems to decrease. In addition, if more covariates that are included that do not a have a non-zero effect on the survival, then more covariates are selected in total, where all the correct effects are selected for the datasets with 1600 observations, but not for the datasets with 400 observations. For these datasets, when the number of additional covariates increases, the number of correctly selected effects decreases, while the total number of selected effects increases.

5.3 Predictive accuracy

As a way to assess the predictive power of the different algorithms we have discussed in this thesis, we will simulate a selection of different datasets from different models. We will fit models to these datasets, predict survival probabilities, and validate these via the dynamic Brier score. The models will be fit using the CoxBoost algorithm, the IplBoost algorithm, the lasso algorithm, and sliding landmarking using the lasso algorithm to select covariates (SL-PS). We will look at a situation with only time-varying effects, one with only time-constant effects, and a situation with a mixture of time-constant and time-varying effects. For each of these situations, we will consider both effects that are relatively small and relatively large, and datasets that have 400 or 1600 observations. For all the different sampling models we define the landmark models using 11 equidistant landmarks on the interval $[0, 1]$, and a landmark interval width of $w = 0.25$. In addition, all observations are right-censored at $t = 1.5$, all covariates are drawn from a uniform distribution on $[-1, 1]$, and all datasets contain either 0, 10, 50 or 200 additional covariates that are indepentent of the simulated survival times.

First, we look at a setting where the data is generated using only time-constant effects. The data is generated with an exponential baseline hazard $\alpha_0(t) = k$, where k is chosen to be 2 for the datasets with 400 observations, and 6 for the datasets with 1600 observations. The censoring times are drawn from exponential distributions with rate $\lambda = 0.2$.

Table 5.2: Simulation with 10 time-constant effects. The effects are all equal to $\log(1.5)$.

N	p	CoxBoost	IplBoost	lasso	landmarking, lasso selection
400	10	0.1602	0.1660	**0.1602**	0.1605
400	20	0.1613	0.1678	**0.1613**	0.1618
400	60	0.1632	0.1700	**0.1631**	0.1638
400	210	0.1657	0.1723	**0.1656**	0.1662
1600	10	0.1588	0.1606	**0.1588**	0.1605
1600	20	**0.1591**	0.1615	0.1591	0.1622
1600	60	**0.1595**	0.1625	0.1595	0.1651
1600	210	0.1601	0.1636	**0.1600**	0.1699

Table 5.3: Simulation with 10 time-constant effects. The effects are all equal to $\log(3)$.

N	p	CoxBoost	IplBoost	lasso	landmarking, lasso selection
400	10	0.0914	0.0959	**0.0914**	0.0914
400	20	0.0923	0.0995	**0.0922**	0.0924
400	60	0.0942	0.1041	**0.0936**	0.0941
400	210	0.0964	0.1092	**0.0952**	0.0963
1600	10	**0.0904**	0.0914	0.0904	0.0914
1600	20	0.0907	0.0924	**0.0906**	0.0925
1600	60	0.0913	0.0942	**0.0910**	0.0952
1600	210	0.0917	0.0949	**0.0914**	0.1006

We consider datasets where all the effects are set to $\beta_j = \log(1.5)$, and where $\beta_j = \log(3)$. Tables summarising simulation results for both of these situations are given in tables 5.2 and 5.3. These tables, and the tables below contain the average dynamic Brier score for each simulation, averaged over all the simulations that are based on the same model. All experiments are here repeated 100 times.

For all the models with only time-constant effects under consideration, the Cox model estimated by the lasso or the CoxBoost algorithm is the most accurate, where the lasso has a tendency to be slightly more accurate. The IplBoost algorithm is unsurprisingly never the best at prediction dynamic survival. In addition, there seems to be a tendency that the IplBoost also performs increasingly worse when the number of covariates that do not influence the survival times increases, which is also the case for the other 3 methods. The IplBoost is quite bad compared to the CoxBoost and the lasso, which is not strange as the true effects are constant, and the IplBoost thus has

a lot more parameters to estimate that are not justified by the complexity of the data generating mechanism.

Next, we consider models that have 5 time constant effects of the same sizes as above, and 5 time varying effects. The effects are chosen either so that the average hazard ratio over the follow up range is equal to 1.5, or so that it is equal to 3 for every covariate that has a nonzero effect on the survival times. All the models have Weibull baseline hazards, where the parameter is chosen to be $k = 4$ for the models where the average hazard ratio is 1.5, and $k = 6$ where the average hazard ratio is 3. The censoring times are drawn from exponential distributions with rates equal to $\lambda = 0.33$ for the datasets with smaller effects, and $\lambda = 0.3$ for the datasets with larger effects. The tables 5.5 and 5.4 summarise the average Brier scores of these experiments.

Table 5.4: Simulation with 5 time-constant and 5 time-varying effects. The effects are chosen so that the average hazard ratio for each covariate is equal to 1.5.

N	p	CoxBoost	IplBoost	lasso	landmarking, lasso selection
400	10	0.1716	0.1729	0.1717	**0.1666**
400	20	0.1728	0.1750	0.1728	**0.1690**
400	60	0.1747	0.1766	0.1746	**0.1733**
400	210	**0.1759**	0.1777	0.1759	0.1761
1600	10	0.1699	0.1663	0.1700	**0.1663**
1600	20	0.1702	**0.1678**	0.1702	0.1684
1600	60	0.1705	**0.1693**	0.1703	0.1730
1600	210	0.1713	**0.1705**	0.1707	0.1812

Table 5.5: Simulation with 5 time-constant and 5 time-varying effects. The effects are chosen so that the average hazard ratio for each covariate is equal to 3.

N	p	CoxBoost	IplBoost	lasso	landmarking, lasso selection
400	10	0.1423	0.1406	0.1425	**0.1335**
400	20	0.1432	0.1449	0.1431	**0.1350**
400	60	0.1449	0.1496	0.1446	**0.1392**
400	210	0.1466	0.1528	0.1463	**0.1440**
1600	10	0.1408	0.1337	0.1409	**0.1335**
1600	20	0.1410	0.1351	0.1411	**0.1350**
1600	60	0.1417	**0.1372**	0.1413	0.1397
1600	210	0.1423	**0.1383**	0.1416	0.1472

For the datasets with 400 observations and average hazard ratios equal to 1.5, the performance of the lasso and CoxBoost is worse than SL-PS for the datasets with 0,

10 and 50 covariates with a zero effect, and marginally better for the datasets with 200 covariates without any effect. For all the datasets with 400 observations, the performance of the IplBoost is the worst of all the four methods. For the datasets with 1600 obsetvations and hazard ratios equal to 1.5, the situation is quite different. The IplBoost is only marginally less accurate than the SL-PS scheme for the dataset with 0 additional covariates, and the most accurate for the other datasets. It seems that while the SL-PS method is the most accurate for the datasets with no additional covariates, it cannot handle datasets with more covariates. The reason for the latter might be that the lasso selects more covariates due to the higher number of observations. Comparing the Brier scores for the IplBoost to that of the CoxBoost and lasso, it seems that while the IplBoost is better, it is decreasingly so when the number of covariates with no effect increases.

For the datasets with average hazard ratios equal to 3, and 400 observations SL-PS is significantly better than the other 3 methods. The IplBoost is better than the CoxBoost and the lasso for the datasets with 0 additional covariates, but not for the datasets with 10, 50 and 200 additional covariates. For the datasets with hazard ratios equal to 3 and 1600 observations, the IplBoost and the SL-PS are similar in performance for the datasets with 0 and 10 additional covariates, where the IplBoost is marginally less accurate. For the datasets with 50 and 200, the IplBoost is the most accurate algorithm. We also see also that the performance of SL-PS is quite bad for the datasets with 200 additional covariates. While the IplBoost is better than the CoxBoost and the lasso, we here also see that the gap in terms of performance between the lasso and CoxBoost and the IplBoost is decreasing when the number of additional covariates increases.

To conclude, we consider models that exclusively have time-varying effects. These are also chosen such that they cross 0 at $t_u = 0.7$, and such that the average hazard ratio over the follow up range is equal to either 3 or 1.5. All the datasets are simulated with a baseline hazard of Weibull form $\alpha_0(t) = kt^{k-1}$, where k is set to $k = 6$. The censoring times are drawn from exponential distributions with rates equal to $\lambda = 0.25$ for the datasets with smaller effects, and $\lambda = 0.33$ for the datasets with larger effects. As with the previous examples, the results are summed up in terms of average dynamic Brier scores, which are given in table 5.6 and table 5.7.

Table 5.6: Simulation with 10 time varying effects. The effects are chosen so that the average hazard ratio for each covariate is equal to 1.5.

N	p	CoxBoost	IplBoost	lasso	landmarking, lasso selection
400	10	0.1456	0.1451	0.1457	**0.1443**
400	20	0.1453	0.1455	0.1453	**0.1449**
400	60	0.1452	0.1453	**0.1451**	0.1451
400	210	0.1453	0.1452	**0.1451**	0.1454
1600	10	0.1465	**0.1396**	0.1467	0.1407
1600	20	0.1461	**0.1415**	0.1462	0.1433
1600	60	0.1451	**0.1433**	0.1450	0.1458
1600	210	0.1450	**0.1443**	0.1449	0.1461

Table 5.7: Simulation with 10 time varying effects. The effects are chosen so that the average hazard ratio for each covariate is equal to 3.

N	p	CoxBoost	IplBoost	lasso	landmarking, lasso selection
400	10	0.1723	0.1642	0.1725	**0.1555**
400	20	0.1731	0.1706	0.1729	**0.1576**
400	60	0.1767	0.1750	0.1745	**0.1621**
400	210	0.1793	0.1773	0.1777	**0.1706**
1600	10	0.1707	0.1557	0.1708	**0.1555**
1600	20	0.1708	0.1579	0.1707	**0.1577**
1600	60	0.1711	**0.1611**	0.1704	0.1637
1600	210	0.1707	**0.1616**	0.1702	0.1760

For the datasets with smaller effects and fewer observations the SL-PS method is best for the datasets with 0 or 10 additional covariates, while the three other methods are somewhat less accurate, but have similar performance. For the datasets with 50 or 200 additional covariates, all the methods have similar performance, but the lasso is marginally the more accurate one. For those datsets with smaller effects and more observations, the IplBoost is the most accurate. The SL-PS is more accurate than the CoxBoost and the lasso for the datasets with 0 or 10 additional covariates, but less for the other datsets. Looking at the datasets with larger effects, SL-PS is the most accurate for the datasets with 400 observations. For these datsets, the IplBoost is more accurate than the CoxBoost and the lasso, but comes quite close to these two for the datasets with 200 additional covariates. For the datasets with 1600 observations and large effects, SL-PS is more accurate for the datasets with either 0 or 10 additional covariates, while the IplBoost is the most accurate for the other datasets. The performance of the

IplBoost and SL-PS is, overall, better than the CoxBoost and the lasso, apart from the datasets with 200 additional covariates, where SL-PS performs poorly.

From these simulations, we observe that there are situations where the sliding landmark models estimated by the IplBoost algorithm, or maximum likelihood fitted models where we select covariates using the lasso (SL-PS), may be superior to Cox proportional hazards models fitted by lasso, or CoxBoost. For this to happen for the cases we have considered, it seems that quite a lot of data must be available. As we would expect, we have also seen that when the data are generated from a model with time-constant effects, the proportional hazards models yield better predictions than the sliding landmark models. The sliding landmark models do not however seem to require that the data come from a model with exclusively time-varying effects, as they appear to be better than the proportional hazards models for the situations we have considered with both constant and varying effects.

Chapter 6

Discussion

In this thesis we have looked at problems concerning survival data, where we both have a high-dimensional covariate space, and where we can have time-dependent effects. Since we are dealing with high dimensional covariate spaces, we have discussed ways of estimating models that are designed to work in this setting. In particular, we have considered L_1-penalised Cox proportional hazards regression [Tibshirani, 1994], and boosted Cox regression, with an emphasis on likelihood boosted Cox regression [Binder and Schumacher, 2008]. For the purpose of handling time dependent covariates, we have studied so called sliding landmark models [van Houwelingen and Putter, 2011], which serve the specific purpose of producing more accurate dynamic predictions when there are time dependent effects. As stated in the introduction, our main goal for this thesis was to try and combine the two concepts, and estimate sliding landmark models using penalised regression, or boosting.

The first possible solution we established is the somewhat ad-hoc method of fitting a proportional hazards model to the dataset using the lasso, and then to fit a sliding landmark model using the covariates that the lasso has selected. We then attempted to use the group lasso [Yuan and Lin, 2006] to estimate a sliding landmark model, by treating the landmark coefficients of a covariate as being grouped together like levels of a categorical covariate. As it turns out, this is not possible to do using the available software. For this to be possible, we would have to be able to estimate the model by stratifying on each landmark dataset. Instead, if we try to do this with the available software (i.e, without stratification), what we end up estimating is the maximum of the group penalised integrated partial likelihood, where the likelihood

contributions are given by (3.1) instead of the correct likelihood contributions (2.7).

Moving on from penalised regression, we considered a way of estimating the sliding landmark model using an algorithm based on likelihood boosting. This is done by treating the landmark coefficients of each covariate as one entity, so to speak, and updating all the landmark coefficients of one covariate for each iteration of the algorithm. This approach is attractive in its simplicity in that derivation of the algorithm is completely analogous to that of likelihood boosting, and therefore easy to understand. We cannot 'tweak' existing software so that it also works for sliding landmark models. Therefore we opted for writing taylored software for this problem, which we have collected in an R package. We have named this package *IplBoost*, as its purpose is to estimate a model corresponding to the integrated partial log-likelihood via boosting. As mentioned earlier in the thesis, the software is freely available for download at https://github.com/simbrant/IplBoost.

Landmark models can be seen as a direct answer to the problem of producing good dynamic survival predictions, given that there are time dependent effects. Therefore we judge the adequacy of our solution based on its ability to produce dynamic survival predictions, measured by the dynamic Brier score. The boosting algorithm seems to work well for the data examples we have studied compared to standard methodology, such as penalised Cox regression, but we cannot draw a conclusion merely on these grounds, as this could be by chance. Taking the simulations into account, we are lead to believe that there are situations where using the IplBoost can be useful. The simulation study does, however, indicate that the IplBoost algorithm might require a good number of observations to be more effective than proportional hazards models, and there must also be time dependent effects in the underlying mechanism that generates the data. For many datasets that are available this is not the case, but as current development is that datasets with survival data and genetic variables are getting larger, the IplBoost could be a relevant tool for analysis.

For further work, one of the things we would propose is the development an algorithm optimising the penalised integrated partial log likelihood. In addition, if dynamic predicitons are the ultimate goal, it might be that the way the IplBoost is designed might be slightly off the mark in terms of what it achieves. By this we mean that to minimise the dynamic prediction error, it may be more fruitful to take an aggregate measure of dynamic prediction error, say the sum of the dynamic Brier scores, as the loss function for a boosting algorithm. However, this will at least for the

dynamic Brier scores be more computationally intensive compared to the already quite computationally intensive IplBoost algorithm, and the derivatives of the dynamic Brier scores may not behave 'nicely enough' for the estimation to work properly. Another interesting problem, is to develop a way of penalising Aalen's additive model [Aalen et al., 2008], which as a problem is similar to that of group penalising estimation of the integrated partial likelihood. This is because the estimates of Aalen's additive model are computed incrementally at each event time by a least-square procedure. The difficulty is here then to develop a method that allows for shrinkage and selection, while ensuring that if a covariate is selected, it is selected at all the points of estimation. An interesting problem related to that we have discussed in this thesis, is to be able to determine whether a given covariate has a time-dependent effect or not. In particular one could try and develop a way of determining if there is something to be gained by a landmark effect of a covariate, compared to just a time-constant effect. Conceivably, one could possibly modify the IplBoost algorithm so that covariates can be specified as time-dependent, and the model be estimated by a hybrid of the IplBoost and likelihood boosted Cox regression.

Bibliography

O. Aalen, Ø. Borgan, and H. Gjessing. *Survival and Event History Analysis: A Process Point of View*. Statistics for Biology and Health. Springer New York, 2008. ISBN 9780387685601. URL https://books.google.no/books?id=wEi26X-VuCIC.

Per Kragh Andersen, Ørnulf Borgan, Richard D. Gill, and Niels Keiding. *Statistical Models Based on Counting Processes*. Springer, 1993.

Richard E. Bellman. *Adaptive Control Processes: A Guided Tour*. MIT Press, 1961. ISBN 9780691079011.

Harald Binder. *CoxBoost: Cox models by likelihood based boosting for a single survival endpoint or competing risks*, 2013. URL https://CRAN.R-project.org/package=CoxBoost. R package version 1.4.

Harald Binder and Martin Schumacher. Allowing for mandatory covariates in boosting estimation of sparse high-dimensional survival models. *BMC Bioinformatics*, 9(1): 14, Jan 2008. ISSN 1471-2105. doi: 10.1186/1471-2105-9-14. URL https://doi.org/10.1186/1471-2105-9-14.

Patrick Breheny and Jian Huang. Group descent algorithms for nonconvex penalized linear and logistic regression models with grouped predictors. *Statistics and Computing*, 25:173–187, 2015.

Leo Breiman. Heuristics of instability and stabilization in model selection. *Ann. Statist.*, 24(6):2350–2383, 12 1996. doi: 10.1214/aos/1032181158. URL https://doi.org/10.1214/aos/1032181158.

Glenn W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, 1950. doi: 10.1175/1520-0493(1950)078<0001:VOFEIT>2.0.CO;2. URL https://doi.org/10.1175/1520-0493(1950)078<0001:VOFEIT>2.0.CO;2.

Hege M. Bøvelstad and Ørnulf Borgan. Assessment of evaluation criteria for survival prediction from genomic data. *Biometrical Journal*, 53(2):202–216, 2011. doi: 10.1002/bimj.201000048. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/bimj.201000048.

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. doi: 10.1093/bioinformatics/btm305. URL +http://dx.doi.org/10.1093/bioinformatics/btm305.

Peter Bühlmann and Torsten Hothorn. Boosting algorithms: Regularization, prediction and model fitting. *Statist. Sci.*, 22(4):477–505, 11 2007. doi: 10.1214/07-STS242. URL https://doi.org/10.1214/07-STS242.

Gerda Claeskens and Nils Lid Hjort. *Model Selection and Model Averaging*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2008. doi: 10.1017/CBO9780511790485.

D. R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34(2):187–220, 1972. ISSN 00359246. URL http://www.jstor.org/stable/2985181.

Peter Craven and Grace Wahba. Smoothing noisy data with spline functions. *Numerische Mathematik*, 31(4):377–403, Dec 1978. ISSN 0945-3245. doi: 10.1007/BF01404567. URL https://doi.org/10.1007/BF01404567.

Riccardo De Bin. Boosting in cox regression: a comparison between the likelihood-based and the model-based approaches with focus on the r-packages coxboost and mboost. *Computational Statistics*, 31(2):513–531, Jun 2016. ISSN 1613-9658. doi: 10.1007/s00180-015-0642-2. URL https://doi.org/10.1007/s00180-015-0642-2.

L. Devroye. *Non-Uniform Random Variate Generation*. Springer New York, 1986. ISBN 9783540963059. URL https://books.google.no/books?id=mEw_AQAAIAAJ.

Dirk Eddelbuettel and Romain François. Rcpp: Seamless R and C++ integration. *Journal of Statistical Software*, 40(8):1–18, 2011. doi: 10.18637/jss.v040.i08. URL http://www.jstatsoft.org/v40/i08/.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, August 1997. ISSN 0022-0000. doi: 10.1006/jcss.1997.1504. URL http://dx.doi.org/10.1006/jcss.1997.1504.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *Ann. Statist.*, 28(2):337–407, 04 2000. doi: 10.1214/aos/1016218223. URL https://doi.org/10.1214/aos/1016218223.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33 (1):1–22, 2010. URL http://www.jstatsoft.org/v33/i01/.

Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001. ISSN 00905364. URL http://www.jstor.org/stable/2699986.

S. Geisser. The predictive sample reuse method with applications. *Journal of The American Statistical Association*, 70(350), 1975.

Erika Graf, Claudia Schmoor, Willi Sauerbrei, and Martin Schumacher. Assessment and comparison of prognostic classification schemes for survival data. *Statistics in Medicine*, 18(17-18):2529–2545. doi: 10.1002/(SICI)1097-0258(19990915/30)18:17/18<2529::AID-SIM274>3.0.CO;2-5. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-0258%2819990915/30%2918%3A17/18%3C2529%3A%3AAID-SIM274%3E3.0.CO%3B2-5.

Patricia M. Grambsch and Terry M. Therneau. Proportional hazards tests and diagnostics based on weighted residuals. *Biometrika*, 81(3):515–526, 1994. ISSN 00063444. URL http://www.jstor.org/stable/2337123.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1970.

Torsten Hothorn, Peter Buehlmann, Thomas Kneib, Matthias Schmid, and Benjamin Hofner. *mboost: Model-Based Boosting*, 2017. URL https://CRAN.R-project.org/package=mboost. R package version 2.8-1.

John Kalbfleisch and Ross Prentice. The statistical analysis of failure time data. 34:11 – 11, 05 1986.

Jochen Knaus. *snowfall: Easier cluster computing (based on snow).*, 2015. URL https://CRAN.R-project.org/package=snowfall. R package version 1.84-6.1.

Hein Putter. *dynpred: Companion Package to "Dynamic Prediction in Clinical Survival Analysis"*, 2015. URL https://CRAN.R-project.org/package=dynpred. R package version 0.1.2.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2017. URL https://www.R-project.org/.

Greg Ridgeway. The state of boosting. *Computing Science and Statistics*, 31, 12 2001.

Noah Simon, Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for cox's proportional hazards model via coordinate descent. *Journal of Statistical Software*, 39(5):1–13, 2011. URL http://www.jstatsoft.org/v39/i05/.

M. Stone. Cross-validatory choice and assessment of statistical predictions. *Roy. Stat. Soc.*, 36:111–147, 1974.

Terry M Therneau. *A Package for Survival Analysis in S*, 2015. URL https://CRAN.R-project.org/package=survival. version 2.38.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

Robert Tibshirani. The lasso method for variable selection in the cox model. *Statistics in Medicine*, 16(4):385–395, 1997. ISSN 1097-0258. doi: 10.1002/(SICI)1097-0258(19970228)16:4<385::AID-SIM380>3.0.CO;2-3. URL http://dx.doi.org/10.1002/(SICI)1097-0258(19970228)16:4<385::AID-SIM380>3.0.CO;2-3.

Gerhard Tutz and Harald Binder. Generalized additive modeling with implicit variable selection by likelihood-based boosting. *Biometrics*, 62(4):961–971, 2006. ISSN 1541-0420. doi: 10.1111/j.1541-0420.2006.00578.x. URL http://dx.doi.org/10.1111/j.1541-0420.2006.00578.x.

Hans van Houwelingen and Hein Putter. *Dynamic Prediction in Clinical Survival Analysis*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 2011. ISBN 1439835330, 9781439835333.

Hans C. van Houwelingen. Dynamic prediction by landmarking in event history analysis. *Scandinavian Journal of Statistics*, 34(1):70–85, 2007. ISSN 03036898, 14679469. URL http://www.jstor.org/stable/41548539.

Hans C. van Houwelingen, Tako Bruinsma, Augustinus A. M. Hart, Laura J. van't Veer, and Lodewyk F. A. Wessels. Cross-validated cox regression on microarray gene expression data. *Statistics in Medicine*, 25(18):3201–3216, 2006. ISSN 1097-0258. doi: 10.1002/sim.2353. URL http://dx.doi.org/10.1002/sim.2353.

Pierre J. M. Verweij and Hans C. Van Houwelingen. Cross-validation in survival analysis. *Statistics in Medicine*, 12(24):2305–2314, 1993. ISSN 1097-0258. doi: 10.1002/sim.4780122407. URL http://dx.doi.org/10.1002/sim.4780122407.

Pierre J. M. Verweij and Hans C. Van Houwelingen. Penalized likelihood in cox regression. *Statistics in Medicine*, 13(23-24):2427–2436, 1994. ISSN 1097-0258. doi: 10.1002/sim.4780132307. URL http://dx.doi.org/10.1002/sim.4780132307.

Ronghui Xu and J O'Quigley. Estimating average regression effect under non-proportional hazards. 1:423–39, 01 2001.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.

Appendix A

Software

In this thesis, we have used a variety of software. We will here give a summary of the software we have used, and of the software that has been developed specifically for this thesis.

A.1 R packages

We have used a few different packages that are not part of the R base distribution, some of which we have mentioned throughout this thesis. We will here give a short summary of most of the packages that are used, and what they have been used for. To fit ordinary Cox proportional hazards models and Cox regression models with smoothing splines, and to compute Schoenfeld residuals [Grambsch and Therneau, 1994], we have used the *survival* package authored by Therneau [2015]. To fit sliding landmark models, without any form of regularisation, we have used the *dynpred* package authored by Putter [2015], in conjunction with the *survival* package. For proportional hazards models with L_1 regularisation, or if you prefer, a lasso penalty, we have used the package *glmnet* [Friedman et al., 2010, Simon et al., 2011]. For proportional hazards models with a group lasso penalty, we have used the *grpreg* package by Breheny and Huang [2015]. This package is also used together with the *dynpred* package in our attempt to fit regularised sliding landmark models using a group lasso penalty. The package *mboost* [Hothorn et al., 2017] is used to fit gradient boosted Cox proportional hazards models, which De Bin [2016] refers to as *model-based* boosting. For the procedure De Bin [2016] refers to as *likelihood-based* boosting, the package *CoxBoost* created by

Binder [2013] is used.

A.2 Selection of software written for this thesis

As with surely any master thesis in statistics, quite a lot of the time spent writing this thesis has been consumed by various programming tasks. Not all of these are worth discussing of course, and we will here try to give an as detailed as possible account without, as it were, explaining the self-explanatory. Perhaps the most noteworthy piece of software written, is the software created to estimate landmark models via likelihood boosting, and so this seems like the most natural starting point.

A.2.1 IplBoost

As mentioned in section 4.4, we have written software that can be used to fit landmark models using likelihood boosting, and compiled this in a package that can be downloaded from *https://github.com/simbrant/IplBoost*. From the users perspective, this package has two functions. One to tune the number of iterations M, called *cv.IplBoost*, and one to fit the models up to a given number of iterations, called *IplBoost*. The software is designed to resemble the *CoxBoost* package, so users already familiar with this package should easily be able to use the software without much explanation. Documentation, and example code is however included in the package, and can be viewed if the package is installed. The function *IplBoost* that is called by the user, initialises the estimates as zero for each covariate, and each landmark as zero, which are organised in a $(S \times p)$ matrix, where S and p are the number of landmarks and covariates, respectively. One column of these, corresponding to the landmark effects are updated in each iteration of the algortihm, and the process of updating these are done by the function shown in figure A.1. This is of course done in the way explained in section 4.4. Computing the updates is quite computationally intensive, and to try and minimise computing time, the likelihood function and its first and second derivatives are computed in *C++*, where the *C++* code is integrated using the *Rcpp* package [Eddelbuettel and François, 2011].

Figure A.1: Function that computes the updates in the IplBoost algorithm.

```
1
2  .IplBoost.iter <- function(times, status, mat, betas, lms, w, lambda) {
3    ## Internal function that performs one iteration of the
4    ## IplBoost algorithm
5
6    # Matrix of risk functions for the coefficients for each landmark point
7    risk.s <- exp(mat %*% t(betas))
8
9    # Call Cpp function to compute S0 for each landmark
10   S0 <- .compute_S0(as.matrix(risk.s), times, length(times), length(lms))
11
12   # Call C++ functions sequentially to compute S1.j and S2.j for each
13   # landmark for each covariate j (loops over j)
14   S1 <- lapply(1:dim(mat)[2], .compute_S1_j, risk=as.matrix(risk.s),
15                times=times, mat=mat, n=length(times), S=length(lms))
16   S2 <- lapply(1:dim(mat)[2], .compute_S2_j, risk=as.matrix(risk.s),
17                times=times, mat=mat, n=length(times), S=length(lms))
18
19   # Call C++ functions to sequentially compute the first derivative and the
20   # negative of the second derivative for each landmark, for each covariate
         j
21   first.der <- lapply(1:dim(mat)[2],
22                       function(j){.compute_u_j(j, status, mat, times, S0,
23                                                S1[[j]], length(times),
24                                                length(lms), lms, w)})
25
26   neg.second.der <- lapply(1:dim(mat)[2],
27                            function(j){.compute_negI_j(j, status, times, S0
                                 ,
28                                                        S1[[j]], S2[[j]],
29                                                        length(times),
30                                                        length(lms), lms, w,
31                                                        lambda)})
32
33   # Compute scores (proportional to second order Taylor expansion of the
         ipl)
34   score.vars <- as.numeric(lapply(1:dim(mat)[2],
35                                   function(j){sum(first.der[[j]]**2/neg.
                                       second.der[[j]])}))
36
37   # Choose the variable that maximises the approximation
38   j.star <- which(score.vars == max(score.vars))
39
40   # Update coefficients
41   betas <- betas
42   betas[, j.star] <- (betas[, j.star] +
43                       first.der[[j.star]]/neg.second.der[[j.star]])
44
45   return(betas)
46 }
```

As mentioned, the function that the user calls again calls the function shown in figure A.1, in a loop. When the function terminates, it returns an object containing an $(M + 1)$ dimensional vector of values of the *integrated partial likelihood*, evaluated

at each iteration, as well as for the null model. In addition, the object also contains a list of matrices containing the coefficient estimates at each iteration, including the null model.

The function that tunes the number of iterations works in a similar fashion, where it divides the data in K partitions, and then calls the function *IplBoost*, excluding one of the partitions K times, yielding $K \times M_{max}$ models. These are then used to compute the cross validated integrated partial likelihood. Upon termination, the function returns an object containing an $M_{max} + 1$ dimensional vector of values for the cross-validated integrated partial likelihood for each iteration, as well as the value of M that maximises this. To be able to further minimise computing time, the package also supports parallellisation of the cross-validation procedure, via the *snowfall* package [Knaus, 2015]. The process of setting up the cross-validation so that it computes in parallel is similar to how the parallelised cross validation for the *CoxBoost* package is organised, but it is also explained in the documentation of *cv.IplBoost*. A remark that can be made about this piece of software, is that it can also be used to estimate likelihood boosted Cox models, of the kind the CoxBoost package does. While lacking a lot of the functionality that is built into CoxBoost, it does in our experience use somewhat less computing time to estimate models. To use the software for this purpose, the user has to specify a single landmark at time $t = 0$, and an interval width w that is larger than the greatest event time.

A.2.2 DynamicBrier

In addition to the package designed to estimate likelihood boosted sliding landmark models, we have also developed another package, *DynamicBrier*, which purpose is to compute dynamic Brier scores, defined in section 2.4.2. This package is not at the present time of writing available for download, but can be made avaiable upon request.

Figure A.2: Function that computes dynamic Brier scores.

```
DynamicBrier <- function(times, status, design, betas, landmarks, w){

  # Check for tied survival times, break if necessary.
  if (length(times) != length(unique(times))){
    times <- times + abs(rnorm(length(times), mean=0, sd=10^(-10)))
  }

  # Order observations ascendingly in time
  status <- status[order(times)]
  design <- design[order(times), ]
  times <- times[order(times)]


  # Compute e(\beta^T(LM_s)x_i) for each landmark
  if (is.null(dim(betas))){
    risk <- as.matrix(vapply(1:length(landmarks),
                             function(s) exp(design %*% betas),
                             times))
  } else{
    risk <- exp(design %*% t(as.matrix(betas)))
  }

  # Call C++ functions to compute S_0 and conditional survival
  # probabilities for each landmark
  s0 <- .compute_S0(risk, times, length(times), length(landmarks))
  pi <- .compute_pi(risk, status, times, s0, length(times),
                    length(landmarks), landmarks, w)

  # Call R functions that compute Y(LM_s), as well as the conditional
  # censoring function evaluated at each survival time in A_s,  and
  # the conditional censoring function evaluated at LM_s + w
  # for each landmark.
  Y_lm <- .comp.Y_lm(times, landmarks)
  c_hat <- .comp.c_hat(landmarks, times, status)
  c_hat_end <- .comp.chat.end(landmarks, w, times, c_hat)

  # Call C++ function that computes the dynamic Brier scores.
  .computeDynamicBrierScores(Y_lm, pi, times, status, c_hat, c_hat_end,
                             landmarks, w, length(times), length(landmarks))
}
```

The interface of this package consists of two functions, one that computes dynamic Brier scores, and another that computes a dynamic R^2 measure based on the dynamic Brier scores. As an illustration, the code for the function that computes the dynamic Brier scores is provided in figure A.2. In order to minimise computing time some of the heavier lifting is here also done by code written in *C++*, which is integrated using the *Rcpp* package.

Appendix B

Derivation of results from section 2.3.1

We will here provide some derviations of the results presented in section 2.3.1 of this thesis. In all of what follows, we assume that we have right censored survival data, where the hazard function of an individual with index i can be described as

$$\alpha_i(t) = \alpha_0(t)e^{\boldsymbol{\beta}^T(t)\mathbf{x}_i},$$

that the covariates are centered, and that the survival and censoring times are independent given the covariates.

B.1 Derivation of 2.3

For the following we impose the condition that

$$\int_0^t \alpha_0(s)\left((\boldsymbol{\beta}(s) - \overline{\boldsymbol{\beta}}(t))^T\mathbf{x}_i\right)^2 ds \text{ is small,}$$

where

$$\overline{\boldsymbol{\beta}}(t) = \frac{\int_0^t \alpha_0(s)\boldsymbol{\beta}(s)ds}{A_0(t)},$$

which requires that $\boldsymbol{\beta}(s)\mathbf{x}_i$ is small and does not to vary too much. Using a Taylor expansion of $e^{\boldsymbol{\beta}^T(s)\mathbf{x}_i}$ around the point $e^{\boldsymbol{\beta}^T(s)\mathbf{x}_i}$, we can write it as

$$e^{\boldsymbol{\beta}^T(s)\mathbf{x}_i} = e^{\overline{\boldsymbol{\beta}}^T(t)\mathbf{x}_i} + e^{\overline{\boldsymbol{\beta}}^T(t)\mathbf{x}_i}\left(\boldsymbol{\beta}(s) - \overline{\boldsymbol{\beta}}(t)\right)^T\mathbf{x}_i + \frac{e^c}{2}\left((\boldsymbol{\beta}(s) - \overline{\boldsymbol{\beta}}(t))^T\mathbf{x}_i\right)^2,$$

where c lies between $\overline{\boldsymbol{\beta}}(t)\mathbf{x}_i$ and $\boldsymbol{\beta}(s)\mathbf{x}_i$. Multiplying both sides of this expression with $\alpha_0(s)$ and integrating from 0 to t, we see that

$$\begin{aligned}\int_0^t \alpha_0(t)e^{\boldsymbol{\beta}^T(s)\mathbf{x}_i}ds = e^{\overline{\boldsymbol{\beta}}^T(t)\mathbf{x}_i}\int_0^t \alpha_0(s)ds \\ + e^{\overline{\boldsymbol{\beta}}^T(t)\mathbf{x}_i}\left(\int_0^t \alpha_0(s)\boldsymbol{\beta}^T(s)\mathbf{x}_i ds - \overline{\boldsymbol{\beta}}^T(t)\mathbf{x}_i\int_0^t \alpha_0(s)ds\right) \\ + \frac{e^c}{2}\int_0^t \alpha_0(s)\left((\boldsymbol{\beta}(s)-\overline{\boldsymbol{\beta}}(t))^T\mathbf{x}_i\right)^2 ds.\end{aligned}$$

Since

$$e^{\overline{\boldsymbol{\beta}}^T(t)\mathbf{x}_i}\left(\int_0^t \alpha_0(s)\boldsymbol{\beta}^T(s)\mathbf{x}_i ds - \overline{\boldsymbol{\beta}}^T(t)\mathbf{x}_i\int_0^t \alpha_0(s)ds\right) = 0,$$

and

$$\int_0^t \alpha_0(s)\left((\boldsymbol{\beta}(s)-\overline{\boldsymbol{\beta}}(t))^T\mathbf{x}_i\right)^2 ds$$

is small, we have that

$$A(t|\mathbf{x}_i) = \int_0^t \alpha_0(t)e^{\boldsymbol{\beta}^T(s)\mathbf{x}_i}ds \approx A_0(t)e^{\overline{\boldsymbol{\beta}}^T(t)\mathbf{x}_i}.$$

B.2 Derivation of 2.5

If we fit a Cox proportional hazards model with administrative censoring at some horizon t_{hor}, when the hazard can be described as

$$\alpha_i(t) = \alpha_0(t)e^{\boldsymbol{\beta}^T(t)\mathbf{x}_i},$$

then [van Houwelingen and Putter, 2011] the estimate converges to a limiting value approximately given by

$$\begin{aligned}\tilde{\boldsymbol{\beta}}_{Cox} \approx \left(\int_0^{t_{hor}} S(t)C(t)\mathbb{V}\left(\mathbf{X}|T=t,\boldsymbol{\beta}(t)\right)dt\right)^{-1} \\ \cdot \int_0^{t_{hor}} S(t)C(t)\mathbb{V}\left(\mathbf{X}|T=t,\boldsymbol{\beta}(t)\right)\boldsymbol{\beta}(t)dt,\end{aligned}$$

given that the true coefficients $\boldsymbol{\beta}(t)$ do not vary too much over time. Here $S(t)$, $C(t)$ and $\alpha(t)$ are the marginal survival, censoring and hazard functions, respecitvely. $\mathbb{V}\left(\mathbf{X}|T=t,\boldsymbol{\beta}(t)\right)$ is is defined as the limiting value of

$$\frac{\mathbf{S}^{(2)}(\boldsymbol{\beta}(t),t)}{S^{(0)}(\boldsymbol{\beta}(t),t)} - \left(\frac{\mathbf{S}^{(1)}(\boldsymbol{\beta}(t),t)}{S^{(0)}(\boldsymbol{\beta}(t),t)}\right)\left(\frac{\mathbf{S}^{(1)}(\boldsymbol{\beta}(t),t)}{S^{(0)}(\boldsymbol{\beta}(t),t)}\right)^T$$

where

$$S^{(0)}(\boldsymbol{\beta}(t),t) = \sum_{i=1}^{n} Y_i(t)\exp(\boldsymbol{\beta}(t)^T\mathbf{x}_i),$$

$$\mathbf{S}^{(1)}(\boldsymbol{\beta}(t),t) = \sum_{i=1}^{n} Y_i(t)\mathbf{x}_i\exp(\boldsymbol{\beta}(t)^T\mathbf{x}_i),$$

and

$$\mathbf{S}^{(2)}(\boldsymbol{\beta}(t),t) = \sum_{i=1}^{n} Y_i(t)\mathbf{x}_i\mathbf{x}_i^T\exp(\boldsymbol{\beta}(t)^T\mathbf{x}_i).$$

By limiting value, we here mean the value which the expression above, as it were, approaches when the number of observations increases. Under the conditions that t_{hor}, and the effects of the covariates, are small, $\mathbb{V}\left(\mathbf{X}|T=t,\boldsymbol{\beta}(t)\right)$ is approximately constant over the interval $[0,t_{hor}]$. Thus, under these conditions we have that

$$\tilde{\boldsymbol{\beta}}_{Cox} \approx \frac{\int_0^{t_{hor}} S(s)C(s)\alpha(s)\boldsymbol{\beta}(s)ds}{\int_0^{t_{hor}} S(s)C(s)\alpha(s)ds}.$$

Furthermore, if $C(t)\approx 1$, $S(t)\approx 1$ and $\alpha(t)\propto\alpha_0(t)$, then by (2.4) we have that

$$\tilde{\boldsymbol{\beta}}_{Cox} \approx \overline{\boldsymbol{\beta}}(t_{hor}).$$

B.3 Derivation of 2.6

We will now argue that under some conditions, $A_{Cox}(t_{hor}|\mathbf{x}) \approx A(t_{hor}|\mathbf{x})$. The most important of these conditions is that $\boldsymbol{\beta}(t)^T\mathbf{x}_i$ does not vary too much. First we observe that for the Breslow estimator of the baseline hazard, we have

$$\frac{d\hat{A}_0(\boldsymbol{\beta}(t),t)}{d\hat{A}_0(\boldsymbol{\beta},t)} = \frac{S^{(0)}(\boldsymbol{\beta},t)}{S^{(0)}(\boldsymbol{\beta}(t),t)},$$

for arbitrary $\boldsymbol{\beta}$, where

$$\hat{A}_0(\boldsymbol{\beta}, t) = \sum_{t_i \leq t} \frac{d_i}{\sum_{\ell \in \mathcal{R}_i} \exp(\boldsymbol{\beta}^T \mathbf{x}_\ell)} = \sum_{t_i \leq t} \frac{d_i}{S^{(0)}(\boldsymbol{\beta}, t_i)}.$$

By defining

$$\pi_i(\boldsymbol{\beta}, t) = \frac{Y_i(t) \exp(\boldsymbol{\beta}^T \mathbf{x}_i)}{\sum_{j=1}^n Y_j(t) \exp(\boldsymbol{\beta}^T \mathbf{x}_j)},$$

and writing

$$\frac{S^{(0)}(\boldsymbol{\beta}, t)}{S^{(0)}(\boldsymbol{\beta}(t), t)} = \sum_{i=1}^n \exp((\boldsymbol{\beta} - \boldsymbol{\beta}(t))^T \mathbf{x}_i) \pi_i(\boldsymbol{\beta}, t),$$

we see that by Theorem 1 of Xu and O'Quigley [2001], this converges in probability to

$$\mathbb{E}(\exp((\boldsymbol{\beta} - \boldsymbol{\beta}(t))^T \mathbf{X}) | T = t),$$

given that we have random censoring. By making a Taylor expansion of

$$\exp((\boldsymbol{\beta} - \boldsymbol{\beta}(t))^T \mathbf{X})$$

around $(\boldsymbol{\beta} - \boldsymbol{\beta}(t))^T \mathbb{E}(\mathbf{X}|T = t)$, and then taking the expectation conditioned on that $T = t$, on both sides, one can see that

$$\mathbb{E}(\exp((\boldsymbol{\beta} - \boldsymbol{\beta}(t))^T \mathbf{X}) | T = t) \approx \exp\left\{(\boldsymbol{\beta} - \boldsymbol{\beta}(t))^T \mathbb{E}(\mathbf{X}|T = t)\right\},$$

provided that $(\boldsymbol{\beta} - \boldsymbol{\beta}(t))^T \mathbb{V}(\mathbf{X}|T = t)(\boldsymbol{\beta} - \boldsymbol{\beta}(t))$ is small. Thus we get that

$$\frac{d\hat{A}_0(\tilde{\boldsymbol{\beta}}_{Cox}, t)}{d\hat{A}_0(\boldsymbol{\beta}(t), t)} \approx \exp\left\{(\boldsymbol{\beta}(t) - \tilde{\boldsymbol{\beta}}_{Cox})^T \mathbb{E}(\mathbf{X}|T = t)\right\},$$

and therefore

$$\alpha_{0,Cox}(t) \approx \alpha_0(t) \exp\left((\boldsymbol{\beta}(t) - \tilde{\boldsymbol{\beta}}_{Cox})^T \mathbb{E}(\mathbf{X}|T = t)\right).$$

From this it can be argued that

$$\begin{aligned}
A_{Cox}(t_{hor}|\mathbf{x}) &= \exp(\tilde{\boldsymbol{\beta}}^T\mathbf{x}) \int_0^{t_{hor}} \alpha_{Cox,0}(t)dt \\
&\approx \exp(\tilde{\boldsymbol{\beta}}^T\mathbf{x}) \int_0^{t_{hor}} \alpha_0(t) \exp\left\{(\boldsymbol{\beta}(t) - \tilde{\boldsymbol{\beta}}_{Cox})^T \mathbb{E}(\mathbf{X}|T=t)\right\} dt \\
&= \int_0^{t_{hor}} \exp\left\{\boldsymbol{\beta}(t)\mathbf{x} + (\tilde{\boldsymbol{\beta}}_{Cox} - \boldsymbol{\beta}(t))^T(\mathbf{x} - \mathbb{E}(\mathbf{X}|T=t))\right\} dt \\
&\approx \int_0^{t_{hor}} \alpha_0(t) \exp\left\{\boldsymbol{\beta}(t)^T\mathbf{x}\right\} \\
&= A(t_{hor}|\mathbf{x}),
\end{aligned}$$

given that $\boldsymbol{\beta}(t)$ does not vary too much.

Appendix C

Figures

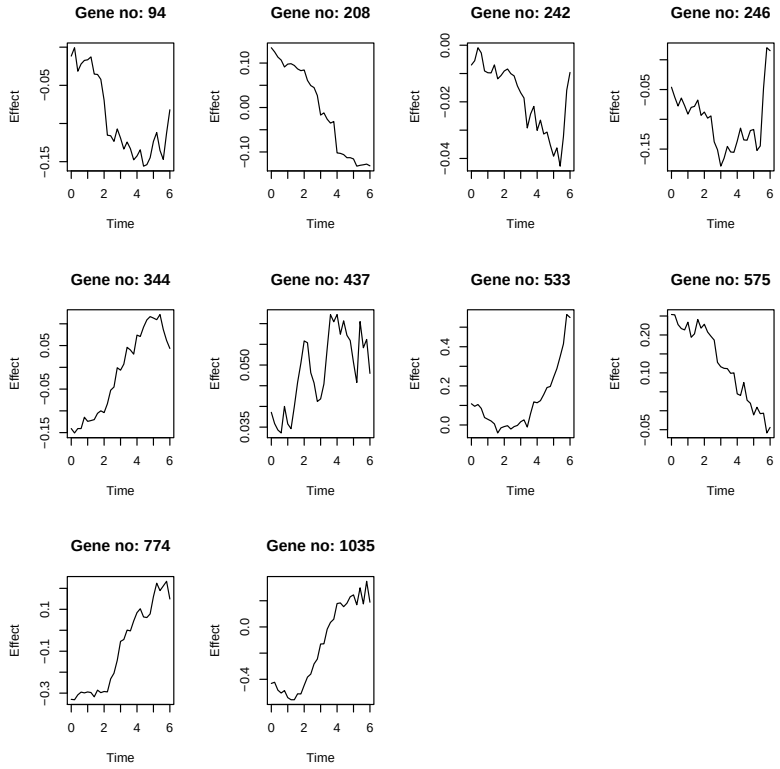

Figure C.1: Some of the estimated coefficient curves for the covariates that are selected only by IplBoost and not by CoxBoost for the Dutch breast cancer data, in section 4.4.2.

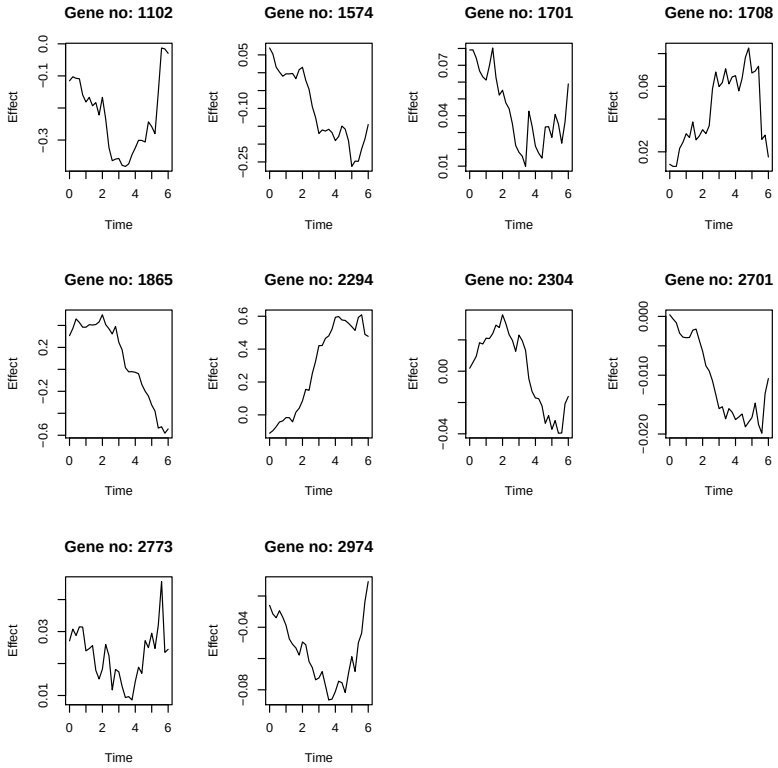

Figure C.2: Some of the estimated coefficient curves for the covariates that are selected only by IplBoost and not by CoxBoost for the Dutch breast cancer data, in section 4.4.2.

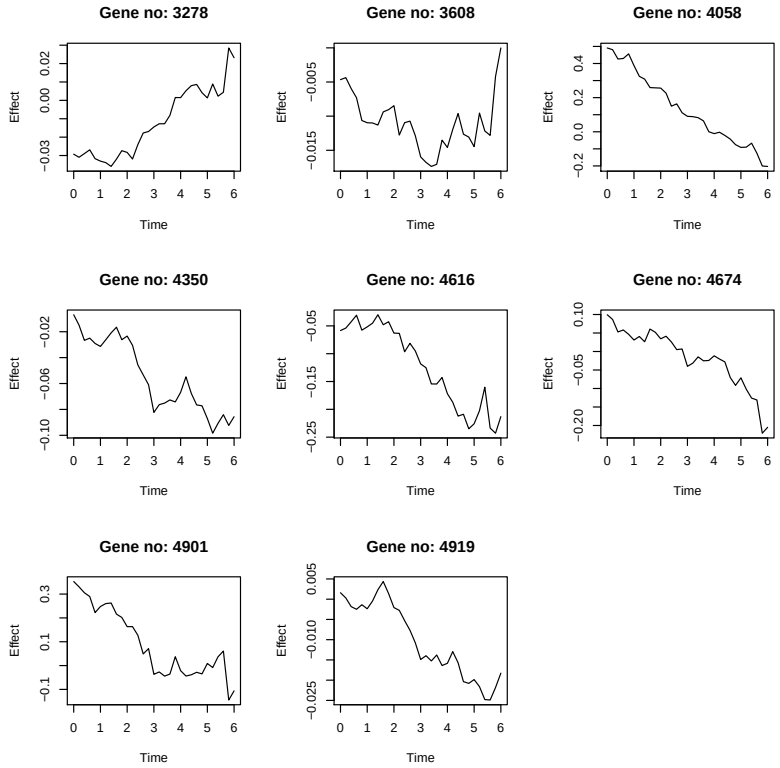

Figure C.3: Some of the estimated coefficient curves for the covariates that are selected only by IplBoost and not by CoxBoost for the Dutch breast cancer data, in section 4.4.2.