Measures of discrimination and predictive accuracy for interval censored survival data

Tsouprou, S.

MATHEMATICAL INSTITUTE

MASTER THESIS

STATISTICAL SCIENCE FOR THE LIFE AND BEHAVIOURAL SCIENCES

Measures of discrimination and predictive accuracy for interval censored survival data

Author:
Sofia Tsouprou

First Supervisor:
Prof. dr. Hein Putter
LUMC

Second Supervisor:
Dr. Marta Fiocco
LUMC & LU

December 2015

Abstract

Medical researchers frequently make statements that one model predicts survival better than another, and are frequently challenged to provide rigorous statistical justification for these statements. In general, it is important to quantify how well the model is able to distinguish between high risk and low risk subjects (discrimination), and how well the model predicts the probability of having experienced the event of interest prior to a specified time t (predictive accuracy). For ordinary – right censored – survival data, the two most popular methods for discrimination and predictive accuracy are the concordance index, or c-index (Harrell et al. 1986) and the prediction error based on the Brier score (Graf et al. 1999).
In the absence of censoring, it is straightforward to define and estimate these measures. Adaptations of these simple estimates for right censored survival data have been proposed and are now in common use. The novel part of this thesis is to develop methods for calculating/estimating the concordance index and the Brier score prediction error in the context of interval censored survival data. The starting point is that we have interval censored data of the form $(L_i, R_i]$ for subjects $i = 1, ..., n$, with $L_i < R_i$(L_i may be 0, R_i may be infinity to accommodate right censored data), and a given prediction model yielding a single (estimated) baseline hazard $h_0(t)$, one vector of (estimated) regression coefficients beta. From this prediction model, prognostic scores $\beta^T x_i$, and predicted survival probabilities $S(t|x_i) = \exp(-H_0(t)\beta^T x_i)$, may be calculated for each subject i. Methods to estimate the concordance index and the Brier score prediction error for exponential and Weibull baseline hazards are proposed and evaluated in a simulation study. An application to real data is also provided.

Acknowledgement

I wish to express special thanks to Prof. dr. H. Putter, my supervisor, who was more than generous with his expertise and precious time and most of all patience throughout the entire process. I would also like to thank the members of the 'Survival lunch' and in particular dr. Marta Fiocco, for their continued support.

I would like to acknowledge and thank my teachers from the master track *Statistical Science for the Life and Behavioural Sciences* for sharing their expertise and enthusiasm about statistics.

Special thanks goes to my family, for supporting me spiritually and whose steadfast support of all these years missing home was greatly needed and deeply appreciated. I thank my fellow classmates and especially Irene Zanellato, in for the stimulating discussions, for the days we were working together before deadlines, and for all the fun we have had in the last two years.

Table of Contents

1 Introduction

1.1 Survival data analysis

Survival analysis is generally defined as a set of methods for analyzing data where the outcome variable is the time until the occurrence of an **event of interest**. The event can be death, occurrence of a disease, etc. The time to event or survival time can be measured in days, weeks, years, etc. Survival analysis involves the modelling of time to event data; in this context, death or failure is considered an ”event”. In survival analysis, subjects are usually followed over a specified time period and the focus is on the time at which the event of interest occurs.

1.2 Censoring

Censoring is an important issue in survival analysis, representing a particular type of missing data. Censoring that is random and non informative is usually required in order to avoid bias. Censoring occurs when observations have some information available for a variable but the information is not complete. This lack of information arises when a variable can be measured precisely only within a certain range. Outside of this range, the only information available is that it is greater or smaller than a specific value or that it lies between two values. Analyzing a censored variable requires procedures designed to account for the censoring. Censoring is a form of missing data problem which is common in survival analysis. Ideally, both the birth and death dates of a subject are known, in which case the lifetime is known. If it is known only that the date of death is after some date, this is called right censoring. Right censoring will occur for those subjects whose birth date is known but who are still alive when they are lost to follow-up or when the study ends.

There are three main types of censoring: right, left, and interval. The present thesis focuses on **interval censoring**. Interval censoring – a data point is somewhere on an interval between two values. Interval censored data arises when a failure time T can not be observed, but can only be determined to lie in an interval obtained from a sequence of examination times. Interval censoring can occur when observing a value requires follow-ups or inspections. Left and right censoring are special cases of interval censoring, with the beginning of the interval at zero or the end at infinity, respectively.

1.3 Interval censored data

Let T denote a nonnegative random variable representing survival time of a subject. T is a continuous variable with a known survival function $S(t)$. An observation on T is **interval censored** when the exact value of T is unknown and only an interval $(L, R]$ with $L \leq R$ is observed for which $T \in (L, R]$.
In the case of interval censored data, the event of interest is only known to be within an interval (in Figure 1.1 , a_i and b_i). That means that we know that the event happened in the interval but not the exact time point.

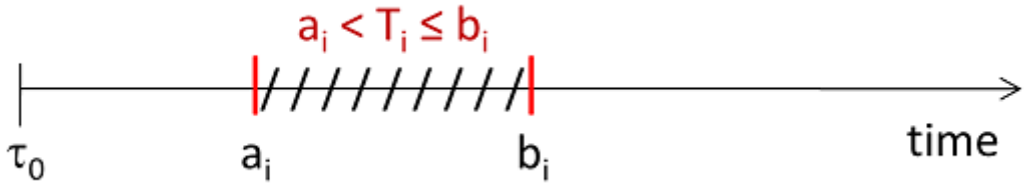


Figure 1.1: The range of an interval and the event of interest

The interval censoring scheme can be described as follows. Suppose n identical items are put on a life test and let $T_1, \ldots, T_n$ be the lifetime of these items. For the i-th item, there is a random censoring interval $(L_i, R_i]$, which follows some unknown bivariate distribution. Here L_i and R_i denote the left and right random end point, respectively, of the censoring interval. Every subject is observed in an interval, where $0 \leq L_i < R_i \leq \infty$. If $R_i = \infty$, the actual value of T_i is not observed in the interval $(L_i, R_i]$ and the i-th item is considered to be right censored at L_i.

Throughout this thesis, it is assumed that the censoring mechanism is independent and non-informative. To satisfy this assumption, the design of the underlying study must ensure that the mechanisms giving rise to censoring of individual subjects are not related to the probability of an event occurring.

1.4 C-index

Multivariable regression models are widely used in biomedical research with the aim of predicting the outcome of individual patients. In this context, the assessment of the model performance has to focus on the accuracy of the predictions, rather than merely on the covariate effects and their statistical signicance. Similarly, when new covariates are available, their possible contribution to the model may be evaluated by the gain in predictive accuracy. Two concepts play an important role in assessing the performance of prediction models in survival analysis, discrimination and calibration.

Prediction is of fundamental importance in all the sciences. The accuracy of a measurement is the degree of closeness of measurements of a quantity to that quantity's true value. Over many predictions, accuracy can be measured as a product of both calibration and discrimination:

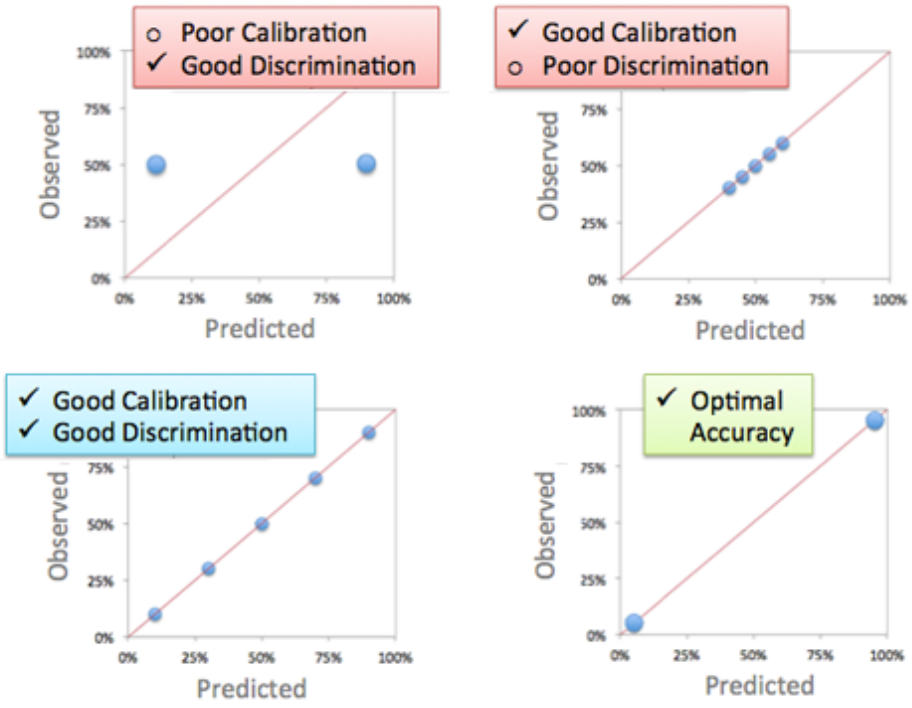


Figure 1.2: Calibration and Discrimination

Considering the difference between calibration and discrimination (see Figure 1.2), discrimination measures the ability to correctly separate outcome classes and is typically assessed by c-index, while calibration shows how closely the predicted probabilities agree with the actual outcome. Perfect calibration and perfect discrimination are usually inconsistent for predictive models.
Discrimination is important in a diagnostic setting where the classification of individuals into different groups is the main interest. However, in a prognostic setting, where individuals probabilities of future events are the main goal, calibration is of paramount importance and should not be ignored.

Definition 1.4.1

Models that distinguish well between patients who die quickly and those who die later on are said to have good **discrimination**. A commonly used measure of discrimination is the c-index. It is the probability for a randomly selected pair that the order of dying is correctly predicted by the model. The c-index ranges from 0 to 1, with higher values indicating better discrimination. A value of 0.5 corresponds to no discrimination.

Definition 1.4.2

Calibration refers to the ability of a model to match predicted and observed death rates across the entire range of the data. A model in which the numbers of observed deaths align well with the numbers of deaths predicted by the model demonstrates good calibration. Good calibration is essential for reliable risk adjustment.

Most often the event one wants to predict is in the future, but predictive modelling can be applied to any type of unknown event, regardless of when it occurred. This thesis will study prediction models that are defined in the context of proportional hazards, where the hazard $h(t|X)$ of a subject with covariates X equals:

$$h(t|X) = h_0(t)\exp(\beta^T X)$$

The *baseline hazard* $h_0(t)$ may either be defined by a parametric model or may be completely unspecified. The model defines a so-called prognostic score $\beta^T x$ that orders subjects (through their covariate values) in terms of their risk; high values of $\beta^T x$ imply a high risk, while low values of $\beta^T x$ correspond to a low risk.

The **concordance index** (c-index) is the probability that the order of dying for a random pair of subjects is correctly predicted by the model. It measures predictive information derived from a set of predictor variables in a model. If the predicted survival time is larger for the subject who lived longer, the predictions for that pair are said to be concordant with the outcomes. A null model, or a model which has no discrimination (when predicted survivals are identical for a patient pair), will have a concordance index of 0.5, while a perfect model will have a concordance of 1. In the absence of censoring, this concordance index may be estimated by considering all pairs and calculating the proportion of these pairs for which the model has correctly identified the order. In a proportional hazards model, the predicted order for two subjects is completely

determined by the prognostic scores $\beta^T x_1$ and $\beta^T x_2$; the subject with the highest prognostic score is expected to experience the event of interest first. Thus, a natural estimate of the concordance index for survival data without censoring is the mean of 0/1 variables over all usable pairs, indicating for each pair whether the order has been predicted correctly.

Definition 1.4.3

C-index in the case of no censoring The formula used to estimate the c-index is:

1. Create all pairs of observations.
2. Test whether the corresponding predictions $z_i = \hat{\beta}^T x_i$ are concordant, i.e, $z_i > z_j$ and $t_i < t_j$, or $z_i < z_j$ and $t_i > t_j$. If so add 1 to the running sum s. If $z_i = z_j$, add 0.5 to the sum. Count the number $\binom{n}{2}$ of response pairs.
3. Divide the total sum by the number of response pairs.

In the ideal case of **no censoring**, the c-index is calculated by the formula:

$$c = \binom{n}{2}^{-1} \sum_{i<j} \sum \left[\mathbb{1}\{z_i > z_j \text{and } t_i < t_j\} + \mathbb{1}\{z_i < z_j \text{and } t_i > t_j\} \right]$$

Despite the fact that in the absence of censoring it is straightforward to define and estimate these measures, in the presence of right censoring – which is usually the case for survival data, complications arise for the calculation of the concordance index. The reason is that the order of dying cannot always be established with certainty. In the context of right censoring, it has been proposed that it could be dealt with by so-called inverse probability of censoring weighting. The idea is to use only the pairs for which the order of dying can be established with certainty and to reweight these "usable" pairs or subjects.

1.5 Brier score

The Brier Score is probably the most commonly used verification measure for assessing the predictive accuracy of a prognostic model. There are various ways to assess the performance of a statistical prediction model. The Brier score is

a proper score function that measures the accuracy of probabilistic predictions. The set of possible outcomes can be either binary or categorical in nature, and the probabilities assigned to this set of outcomes must sum to one (where each individual probability is in the range of 0 to 1).
The Brier score prediction error is the squared distance between the indicator function of the event having taken place before time t and the predicted event probability, or, equivalently, between the indicator function for survival and the predicted survival probability. Note that, while the baseline hazard doesn't play a role in the concordance index, it does play a role in the prediction error. The Brier score prediction error at a given time t can easily be estimated, in the absence of censoring, by the mean over all subjects of the squared distances between the indicator of survival beyond time t for the subject and the predicted survival probability given the model for that subject. For right censored survival data, inverse probability of censoring weighting techniques can be used. In Chapter 4 an estimate of the Brier score is derived in the case of interval censored data.

1.6 Data description

A data set of heart transplant monitoring data from the msm package (`data(cav)`) is used to illustrate our methods. It consists of a series of approximately yearly angiographic examinations of heart transplant recipients. The state at each time is a grade of cardiac allograft vasculopathy (CAV), a deterioration of the arterial walls, eventually leading to death. The data have been (considerably) simplified, by only extracting information about death. So the endpoint is death, the column 'la' corresponds to the left of the interval (last known to be alive), the column 'fd' to the right of the interval (first known to be dead). If 'fd' is infinity, then the subject is never seen to be dead, so this is right censoring. There are 614 patients (by removing 8 patients with missing values of the covariates), the rows are grouped by patient number and ordered by years after transplant, with each row representing an examination and containing additional covariates. An example of 20 individuals is ordered by the left interval of each individual (L_i) presented in Figure 1.3, as a perception of the data used. The intervals of each individual have a different start and end point, while the chronological time of each range also varies. To this extent, the intervals may overlap.

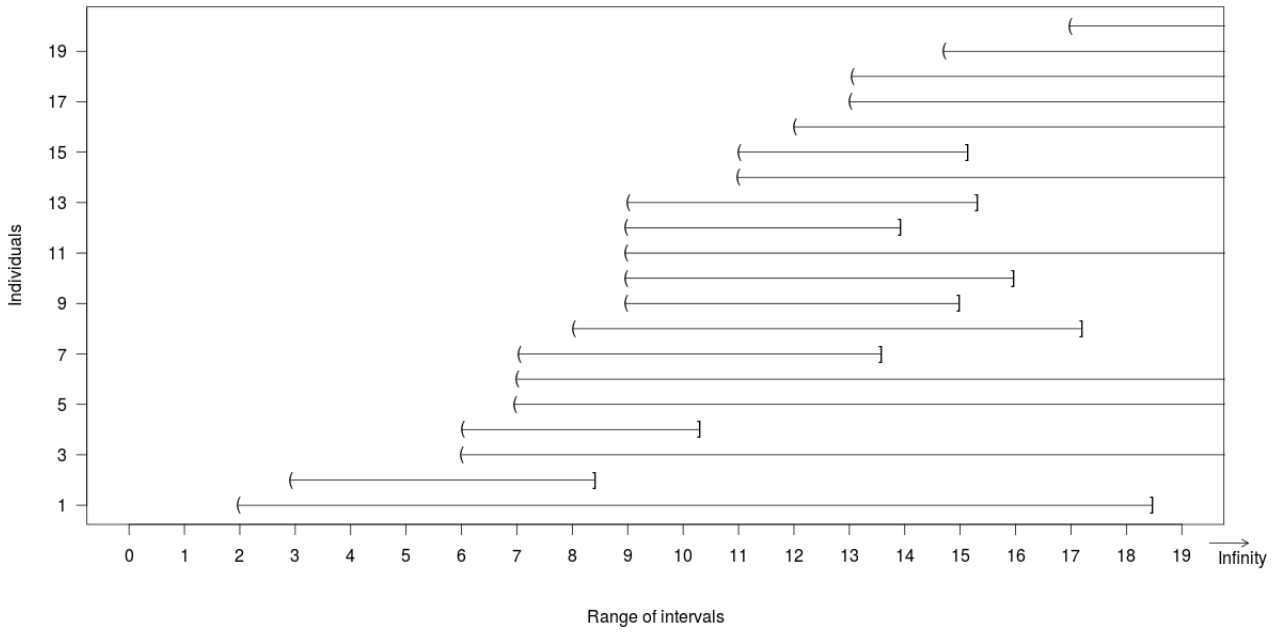


Figure 1.3: Visualization of interval censored data

The covariates in the data are:

- PTNUM (numeric) Patient identification number
- dage (numeric) Age of heart donor (years)
- sex (numeric) sex (0=male, 1=female)
- pdiag (factor) Primary diagnosis (reason for transplant) IHD=ischaemic heart disease, IDC=idiopathic dilated cardiomyopathy , Other=other disease

Patients' age is distributed between 0 and 61 years with mean at 30 years and standard deviation at 12 (see Figure 1.4). The majority (85%) of patients in the dataset is male. In addition, the number of deaths assigned per sex is presented. Most patients did not experience the event and are censored. Approximately 41% of males and 32% of females patients died. The primary diagnosis (reason for transplant) is affected by both IHD and IDC in high percentages 50% and 43% respectively (see Table 1).

		n (%)	**Deaths observed (%)**
Gender	**Males**	527 (85.8%)	221 (41.9%)
	Females	87 (14.2%)	28 (32.2%)
Diagnosis	**IDC**	270 (43.9%)	98 (36.3%)
	IHD	313 (50.9%)	139 (44.4%)
	Other	31 (5.2%)	12 (38.7%)
	Total	614	249

Table 1: Death rates by sex and diagnosis

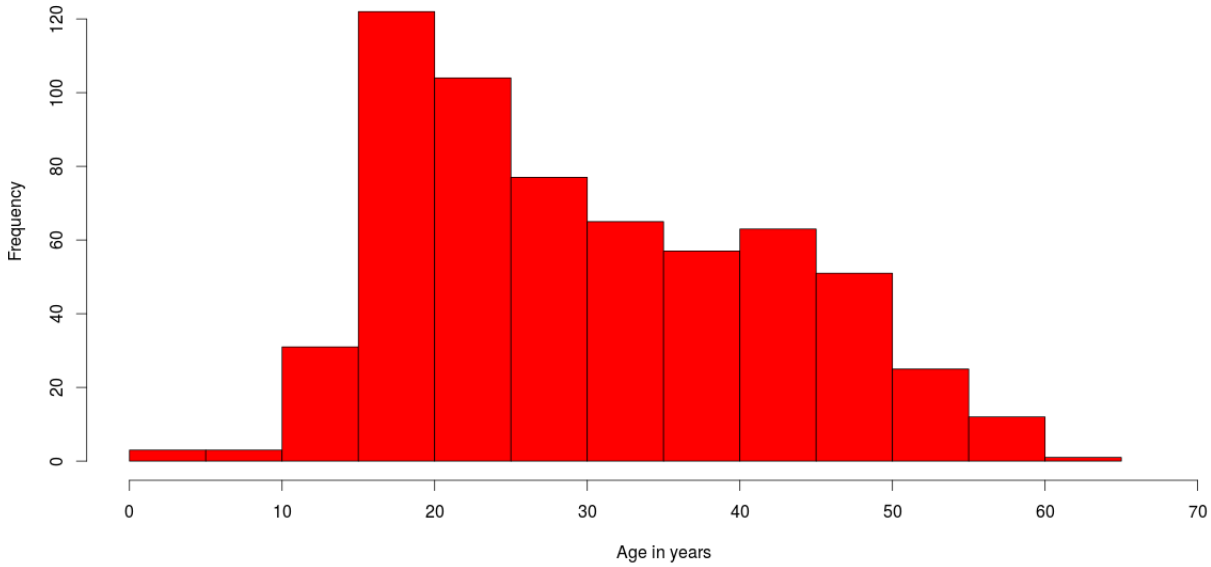


Figure 1.4: Patient age distribution

1.7 Aims of this thesis

Multivariate regression models are powerful tools that are used frequently in studies of clinical outcomes. These models can use a mixture of categorical and continuous variables and can handle partially observed (censored) responses. However, uncritical application of modelling techniques can result in models that poorly fit the dataset at hand, or, even more likely, inaccurately predict outcomes on new subjects.
After having developed a prediction model for survival data, which (assuming that the proportional hazards model holds) boils down to estimating the regression coefficients beta and the baseline hazard, it is important to quantify how well the model is able to distinguish between high risk and low risk subjects (discrimination), and how well the model predicts the probability of having experienced the event of interest prior to a specified time t (predictive accuracy). For ordinary – right censored – survival data, the two most popular methods for discrimination and predictive accuracy are the **concordance index, or c-index** (Harrell et al. 1986) and the **prediction error based on the Brier score** (Graf et al. 1999).
The concept of developing methods for calculating/estimating the concordance index and the Brier score prediction error in *interval censored data* is the challenge of this thesis.

1.8 Structure of the thesis

This thesis is organized as follows. In Chapter 2, it will be shown how parametric survival models based on accelerated failure time (AFT) models can be used to fit proportional hazards models with exponential or Weibull baseline hazards. These models will be the models for which the c-index and the Brier scores are to be calculated. The *survreg* function in R is reviewed and issues of parametrization will be discussed. In Chapter 3, a detailed explanation of the basic formula of the c-index is given and the way it will be derived in the case of interval censored survival data is also provided. Furthermore, the c-index is described separately for the case that time follows an exponential and a Weibull distribution respectively. Technical details and results are provided. Chapter 4 starts by introducing the basic concepts of prediction, predictive accuracy, prediction error and Brier score. Also here extensions to the interval censored case are detailed. To assess the performance of the new methods proposed, a simulation study is performed in Chapter 5. In Chapter 6, cross-validation is used to measure c-index and Brier score. To overcome a possible over-optimism by using the same data to validate the model, leave-one-out cross-validation is performed. Chapter 7 revisits the `data(cav)` introduced in Chapter 1 to illustrate and compare the methods described. Discussion follows about further research of these methods. The statistical analysis is performed in the R-software environment. All R code can be found in the appendix.

2 Models for interval censoring

Semi-parametric proportional hazards regression models for interval censored data are notoriously difficult to fit. Methods to fit these models do exist and are actually implemented in a number of packages in R, but unfortunately they are far from stable and can give unreliable results. A common approach in case of interval censored data is to resort to parametric models, i.e. to assume a particular family of distributions for the baseline hazard in the proportional hazards regression model. In this thesis, we will follow this practice and concentrate on the exponential and Weibull models.

2.1 Exponential and Weibull parametrization

As mentioned, there are different ways to parametrize a model. In our analysis, the parametrization is derived from Klein & Moeschberger (2003).

Definition 2.1.1

Exponential distribution Assume that T denotes the actual survival time and follows an exponential distribution with scale parameter λ. The exponential distribution has constant hazard $\lambda(t) = \lambda$. Thus, the survivor function is $S(t) = \exp(-\lambda t)$ and the density is $f(t) = \lambda \exp(-\lambda t)$, for $\lambda > 0, t \geq 0$. It can be shown that $\mathrm{E}(T) = 1/\lambda$ and $\mathrm{var}(T) = 1/\lambda^2$.

Many probability distributions are not a single distribution, but are in fact a family of distributions. This is due to the distributions having scale and/or shape parameters. Shape parameters allow a distribution to take on a variety of shapes, depending on the value of the shape parameter. These distributions are particularly useful in modeling applications since they are flexible enough to model a variety of data sets. The Weibull distribution is an example of a distribution that has a shape parameter. In the Weibull distribution, it is important to mention that in comparison to the exponential distribution, the baseline hazard function may change over time. Two parameters (shape and scale) must be estimated to describe the underlying hazard function over time. The Weibull distribution reduces to the exponential distribution when the shape parameter γ equals 1. When $\gamma > 1$, the hazard function is increasing; when $\gamma < 1$ it is decreasing.

Definition 2.1.2

Weibull distribution Assume that T denotes the actual survival time and follows a Weibull distribution with scale parameter λ and shape parameter γ, denoted $T \sim W(\lambda, \gamma)$. The survival time T is assumed to be conditionally independent given a (time-invariant) vector of baseline covariates, i.e., the censoring is supposed to be non-informative. The survivor function is $S(t) = \exp(-\lambda t)^{\gamma}$, the density $f(t) = \lambda\gamma t^{\gamma-1}\exp(-\lambda t^{\gamma})$ and the hazard is $\lambda(t) = \lambda\gamma t^{\gamma-1}$.

Another parametrizations are also in use; for instance the functions *survreg* and *rweibull* in the survival package use two different parametrizations. This will be discussed in more detail in section 2.4.

2.2 Proportional hazards (PH) models

It is generally of interest in survival studies to describe the relationship of a factor of interest (e.g.treatment) to the time to event, in the presence of several covariates, such as age, gender, etc. A number of models are available to analyze the relationship of a set of predictor variables with the survival time. Parametric methods assume that the underlying distribution of the survival times follows certain known probability distributions. Popular ones include the exponential and Weibull distributions.
A popular regression model for the analysis of survival data is the **Cox proportional hazards regression model**. It allows testing for differences in survival times of two or more groups of interest, while allow adjusting for covariates of interest. The Cox regression model is usually used as a semi-parametric model, making fewer assumptions than typical parametric methods. In particular, and in contrast with parametric models, it usually makes no assumptions about the shape of the so-called baseline hazard function. The Cox regression model provides useful and easy to interpret information regarding the relationship of the hazard function to predictors. While a non-linear relationship between the hazard function and the predictors is assumed, the hazard ratio comparing any two observations is in fact constant over time in the setting where the predictor variables do not vary over time. This assumption is called the *proportional hazards assumption* and checking if this assumption is met is an important part of a Cox regression analysis.
Let X denote a set of p covariates. The hazard of failure $h(t|X_1, ..., X_p)$ is related to the covariates by:

$$h(t|X_1, ..., X_p) = h_0(t) \exp(\beta_1 X_1 + ... + \beta_p X_p)$$

where $h_0(t)$ is usually an unspecified baseline hazard function for the reference subject with all covariates equal to 0. But it is also possible to specify a parametric hazard like the exponential or Weibull. The linear predictor $\beta^T X = \beta_1 X_1 + ... + \beta_p X_p$ defines a prognostic score that orders subjects (through their covariate values) in terms of their risk.

2.3 Accelerated Failure Time (AFT) models

In this thesis, parametric survival regression models are fitted, concerning the two most common survival distributions, exponential and Weibull. This is a common approach for interval censored survival data, where semi-parametric models are notoriously hard to fit. These models are location-scale models for an arbitrary transform of the time variable, the most common cases use a log-transformation leading to **accelerated failure time models**. In the statistical area of survival analysis, an accelerated failure time model (AFT model) is a parametric model that provides an alternative to the commonly used proportional hazards models. Whereas a proportional hazards model assumes that the effect of a covariate is to multiply the hazard by some constant, an AFT model assumes that the effect of a covariate is to accelerate or decelerate the life course of a disease by some constant. In full generality, the accelerated failure time model can be specified as: $h(t|\theta) = \theta h_0(\theta t)$, where θ denotes the joint effect of covariates, typically $\theta = \exp(-[\alpha_1 X_1 + \cdots + \alpha_p X_p])$. (Specifying the regression coefficients with a negative sign implies that high values of the covariates increase the survival time, but this is merely a sign convention; without a negative sign, they increase the hazard.)

The Weibull distribution (including the exponential distribution as a special case) can be parametrized as either a proportional hazards model or an AFT model, and is the only family of distributions to have this property. The results of fitting a Weibull model can therefore be interpreted in either framework. In this thesis, AFT models will be fitted using the *survreg* function in R and then the results will be transformed to fit a proportional hazards model.
Under AFT models we measure the direct effect of the explanatory variables on the survival time instead of hazard, as we do in the PH model. This characteristic allows for an easier interpretation of the results because the parameters measure the effect of the correspondent covariate on the mean survival time. Similar to the PH model, the AFT model describes the relationship between survival probabilities and a set of covariates.

Definition 2.3.1

For a group of patients with covariate $(X_1, X_2, ..., X_p)$, the model is written mathematically as $S(t|\mathbf{X}) = S_0(t/\eta(\mathbf{X}))$, where $S_0(t)$ is the baseline survival function and $\eta(X) = \alpha_1 X_1 + ... + \alpha_p X_p$ is an "acceleration factor" that is a ratio of survival times corresponding to any fixed value of $S(t)$.

Under an accelerated failure time model, the covariate effects are assumed to be constant and multiplicative on the time scale, that is, the covariate impacts on survival by a constant factor (acceleration factor). According to the relationship of survival function and hazard function, the hazard function for an individual with covariate $X_1, X_2, ..., X_p$ is given by:

$$h(t|\mathbf{X}) = [1/\eta(\mathbf{X})]h_0[t/\eta(\mathbf{X})]$$

The corresponding log-linear form of the AFT model with respect to time is given by:

$$\log T_i = \mu + \alpha_1 X_{1i} + \alpha_2 X_{2i} + ... + \alpha_p X_{pi} + \sigma\epsilon_i \tag{1}$$

where μ is intercept, σ is scale parameter and ϵ_i is a random variable, assumed to have a particular distribution, in our case a Weibull (or exponential) distribution. This form of the model is adopted by most software for the AFT model.

2.4 Survreg parametrization

To set up a Weibull regression (the same procedure follows for the exponential case) using the parametrization of **Definition 2.1.2**, we then assume that the hazard function, for a given covariate vector $\mathbf{X}$ and a corresponding vector β of regression parameters, can be written as:

$$h(t|\mathbf{X}) = \exp(\beta^T\mathbf{X})h_0(t) = \exp(\beta^T\mathbf{X})\lambda_0\gamma t^{\gamma-1} = \lambda^\star\gamma t^{\gamma-1}$$

The last equality shows that we model the rate parameter $\lambda^\star = \lambda_0 \exp(\beta^T\mathbf{X})$ using a baseline rate λ and the effect of the covariates. The coefficients $\exp(\beta_j)$ feature the proportional hazards property, i.e., can be interpreted as hazard ratios. On the other hand, *survreg* embeds Weibull regression in the structure of a more general accelerated failure time model and its output provides estimates for $\log(\sigma)$ [denoted Log(scale)], $-\mu/\sigma$ (Intercept), and the regression parameters α from (1). The reparametrization from **Definition 2.1.2** is:

$$\gamma = \sigma^{-1}, \lambda_0 = \exp(-\mu/\sigma), \beta = -\alpha/\sigma$$

After applying the *survreg* function, we use the output of the *survreg* as input and transform the parameter estimates to the parametrization in **Definition 2.1.2**, thereby allowing to easily switch from the accelerated failure time to the proportional hazard interpretation.

3 Concordance index

The concordance index (Harrell et al., 1982) is a well recognized measure of discrimination for models that predict a time to event. It has been reported more often than any other prediction model metric in the survival setting. In survival analysis, as mentioned in Chapter 1, a pair of subjects is called concordant if the risk of the event predicted by a model is lower for the subject who experiences the event at a later timepoint. The concordance probability (c-index) is the proportion of concordant pairs among all pairs of subjects. Concordant pairs are assigned a score of 1, discordant pairs are assigned a score of 0, and pairs in which there is equality among either variable are assigned a score of 0.5. It can be used to measure and compare the discriminative power of a risk prediction models. Harrell's c-index has been widely used as a measure of separation of two survival distributions.

3.1 C-index and censoring

In the absence of censored data, the c-index is estimated by the proportion of concordant pairs, which is given by the formula:

$$c = \binom{n}{2}^{-1} \sum_{i<j} \sum [\mathbb{1}\{z_i > z_j \text{ and } t_i < t_j\} + \mathbb{1}\{z_i < z_j \text{ and } t_i > t_j\}] \quad (2)$$

Now we consider the situation of possibly censored time to event data. For each patient we observe $T_i = \min(T_i, C_i)$ and $\delta_i = \mathbb{1}(T_i \leq C_i)$, where T_i represents the time to the event of interest and C_i the (hypothetical) time under observation $(i = 1, 2, ..., n)$.
In the presence of randomly right censored data, the c-index is no longer estimated by (2). In the White and Rapsomaniki (2015), there are several options of estimating the c-index while censoring:

1. Harrell et al. (1996): Harrell proposed estimating the c-index as the mean of C_{ij} ($c = \mathrm{E}[C_{ij}]$) over informative pairs, where pair (i, j) is informative if $t_i^\star < t_j^\star$ and $d_i = 1$ or $t_i^\star > t_j^\star$ and $d_j = 1$: that is, if the first event in the pair is observed.

   where $r(x_i)$ is a linear predictor from a correctly specified proportional

hazards model and

$$c_{ij} = \begin{cases} \mathbb{1}\{t_i^\star < t_j^\star\}, & \text{if } r(x_i) > r(x_j) \\ 0.5, & \text{if } r(x_i) = r(x_j) \\ \mathbb{1}\{t_i^\star > t_j^\star\}, & \text{if } r(x_i) < r(x_j) \end{cases}$$

2. Liu et al. (2012): A version of the c-index corrected for censoring can be obtained by Inverse Probability of Censoring Weighting (IPCW). When the censoring time C is random, the estimator of the c-index converges to a quantity depending on the distribution of censoring time. Assuming that the random censoring time C is independent of predictors, a consistent estimator of the c-index has the form:

$$\boxed{\hat{c} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} \mathbb{1}\{X_i < X_j\}\mathbb{1}\{Y_i < Y_j\}w_i}{\sum_{i=1}^{n}\sum_{j=1}^{n} \mathbb{1}\{Y_i < Y_j\}w_i}}$$

where $w_i = \frac{d_i}{G^2(Y_i)}, d_i = \mathbb{1}\{T_i < c_i\}, G(t) = P(t < c)$ for $t > 0$.

3. Gonen and Heller (2005): Gonen and Heller proposed an alternative estimator to avoid bias due to censoring. Suppose $r^\star(x_i)$ is a linear predictor from a correctly specified proportional hazards model. Then $r^\star(x_i) - r^\star(x_j)$ represents the log hazard ratio between individuals i and j, and the probability that individuals i and j are concordant is: $\text{expit}(r^\star(x_i) - r^\star(x_j))$ if $r(x_i) > r(x_j)$ where $\text{expit}(\eta) = \frac{1}{1+\exp(-\eta)}$. Similarly, it is : $\text{expit}(r^\star(x_j) - r^\star(x_i))$ if $r(x_i) < r(x_j)$ and 0.5 if $r(x_i) = r(x_j)$. Then the estimator $\hat{c}$ is the average of this concordance probability.

4. Heagerty and Zheng (2005): Let $\tau = \max_i t_i$ be the longest follow-up time observed. The study only gives information about discrimination at time $t \leq \tau$ and c can only be estimated by extrapolating to times $t > \tau$. For example, $\hat{c}$ assumes that the proportional hazards model continues to hold at times beyond τ. This is called the restricted c-index.

3.2 C-index and interval censoring

In this novel part of the thesis, the derivation of the formulas for calculating the c-index are presented, taking into account that we use interval censored data.

The two main distributions in survival analysis are studied (exponential and Weibull).
The data are of the form (L_i, R_i) for n independent subjects and $z_i = \beta^T X_i$ defines the risk score of each subject. The actual event times are unknown, but it is known that $T_i \in (L_i, R_i]$.
Firstly, we take all pairs of subjects (i, j) and without loss of generality, call the subject with lowest risk z, number 1 and the one with highest risk number 2. The ordering is correct if $T_2 < T_1$. But in interval censored data we do not observe T_1 and T_2. So, we replace $\mathbb{1}\{T_2 < T_1\}$ by an estimate of $P(T_2 < T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2])$.

3.2.1 Exponential case

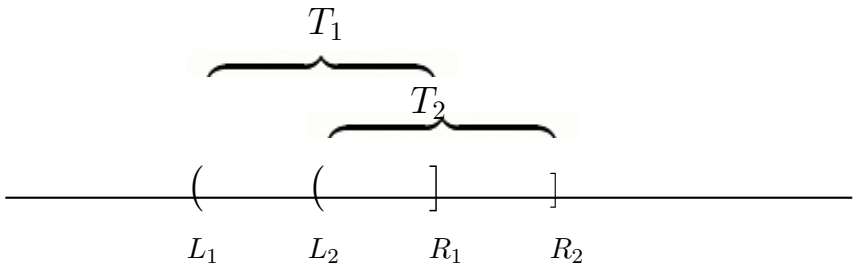
In the exponential case, each ordering of the left and right intervals L_1, L_2, R_1, R_2 could in principle give a different probability of $P(T_2 < T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2])$. There are 4! = 24 of such orderings. But:

- Some orderings are invalid, since we must have $L_1 \leq R_1$ and $L_2 \leq R_2$.
- Some orderings will have probability 0 or 1.
- $P(T_2 > T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2]) = 1 - P(T_1 > T_2 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2])$.

Out of the 24 orderings, for 6 of them the probability of $P(T_2 < T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2])$ needs to be calculated. The predicted probability that "patient1" of each pair will experience the event of interest after "patient 2" will be calculated. i.e. $P(T_2 < T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2])$. The 6 cases are:

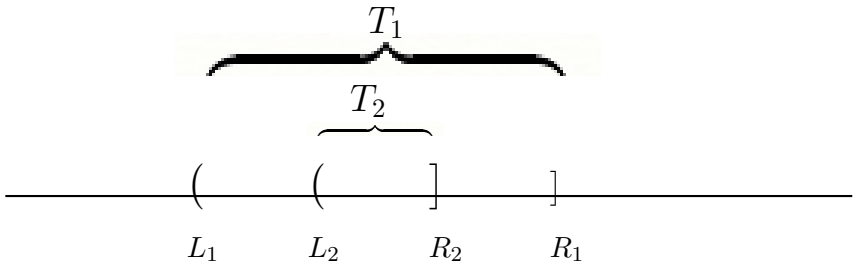
1. $L_1 \leq L_2 \leq R_1 \leq R_2$
2. $L_1 \leq L_2 \leq R_2 < R_1$
3. $L_1 < R_1 \leq L_2 < R_2$
4. $L_2 < L_1 \leq R_1 \leq R_2$
5. $L_2 < L_1 \leq R_2 < R_1$
6. $L_2 < R_2 \leq L_1 < R_1$

1. If $L_1 \leq L_2 \leq R_1 \leq R_2$



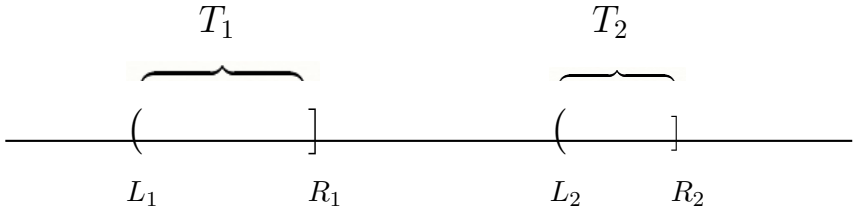
$T_1 < T_2$ or $T_1 > T_2$

2. If $L_1 \leq L_2 \leq R_2 < R_1$



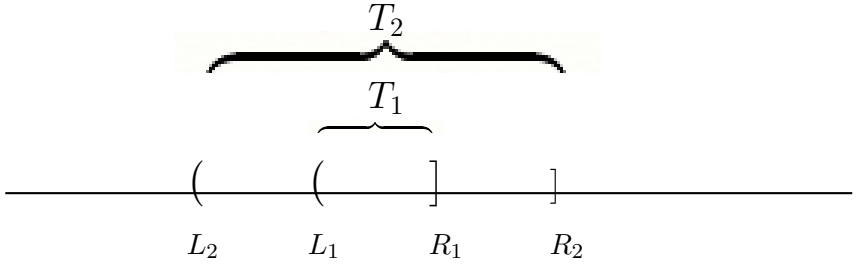
$T_1 < T_2$ or $T_1 > T_2$

3. $L_1 < R_1 \leq L_2 < R_2$



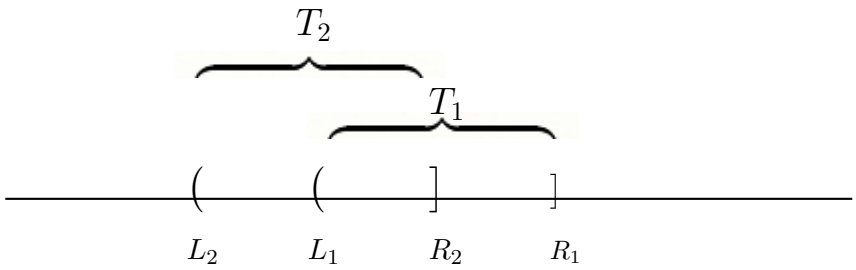
$T_1 < T_2$

4. If $L_2 < L_1 \leq R_1 \leq R_2$



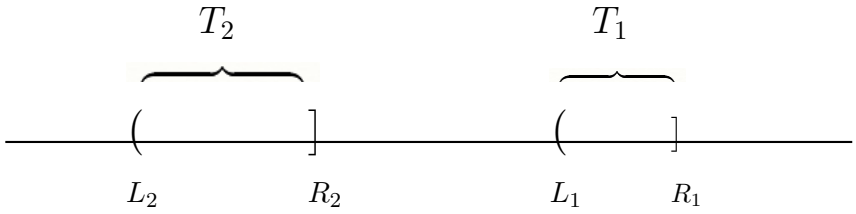
$T_1 < T_2$ or $T_1 > T_2$

5. If $L_2 < L_1 \leq R_2 < R_1$



$T_1 < T_2$ or $T_1 > T_2$

6. If $L_2 < R_2 \leq L_1 < R_1$



$T_1 > T_2$

Given that $T \sim$ exponential distribution and a p-dimensional vector of covariates X, it will be assumed that the hazard function of T will be given by:

$$h(t|X) = h_0(t)\exp(\beta_1 X_1 + ... + \beta_p X_p).$$

Here $h_0(t) \equiv \lambda_0$; $\lambda_1 = \lambda_0 \exp(\beta_1 X_{11} + ... + \beta_p X_{1p}) = \lambda_0 \exp(\beta^T X_1)$; $\lambda_2 = \lambda_0 \exp(\beta^T X_2)$. We observe $T_1 \in (L_1, R_1]$ and $T_2 \in (L_2, R_2]$, and X_1, X_2. So, we can calculate $\lambda_1 = \lambda_0 \exp(\beta^T X_1)$ and $\lambda_2 = \lambda_0 \exp(\beta^T X_2)$, where $T_1 \sim \exp(\lambda_1)$ and $T_2 \sim \exp(\lambda_2)$.

Preliminaries: Conditionally given $T_1 \in (L_1, R_1]$ and $T_2 \in (L_2, R_2]$, T_1 and T_2 are still independent and for $i = 1, 2$:

$$P(T_i \leq t_i | T_i \in (L_i, R_i]) = \frac{P(L_i < T_i \leq t_i)}{P(L_i < T_i \leq R_i)} = \frac{S_i(L_i) - S_i(t_i)}{S_i(L_i) - S_i(R_i)}$$

The corresponding density for an individual i is given by:

$$f_{T_i|T_i \in (L_i, R_i]}(t_i) = \frac{d}{dt_i} P(T_i \leq t_i | T_i \in (L_i, R_i]) = \frac{f_{T_i}(t_i)}{S_i(L_i) - S_i(R_i)} \quad (3)$$

for $t_i \in (L_i, R_i]$.

The probability of $P(T_1 > T_2 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2])$ is the integral over $D = \{(t_1, t_2) \in \mathbf{R}^+ : t_1 > t_2\}$ of the joint density of (T_1, T_2) given $T_1 \in (L_1, R_1]$ and $T_2 \in (L_2, R_2]$. Since T_1 and T_2 are conditionally independent given $T_1 \in (L_1, R_1]$ and $T_2 \in (L_2, R_2]$, this **joint density equals the product of**:

$$f_{T_1|T_1\in(L_1,R_1]}(t_1) * f_{T_2|T_2\in(L_2,R_2]}(t_2) = \frac{f_{T_1}(t_1) f_{T_2}(t_2)}{(S_1(L_1) - S_1(R_1))(S_2(L_2) - S_2(R_2))} \tag{4}$$

For the special case when $T_1 \sim \exp(\lambda_1)$ and $T_2 \sim \exp(\lambda_2)$, we can write $f_{T_i}(t) = \lambda_i$ and $S_i(t) = \mathrm{P}(T_i > t) = e^{-\lambda_i t})$ in (3) and obtain for T_1:

$$\frac{\lambda_1 e^{-\lambda_1 t_1}}{e^{-\lambda_1 L_1} - e^{-\lambda_1 R_1}}$$

and for T_2:

$$\frac{\lambda_2 e^{-\lambda_2 t_2}}{e^{-\lambda_2 L_2} - e^{-\lambda_2 R_2}}$$

while the joint density in (4) equals the product of:

$$\frac{\lambda_1 e^{-\lambda_1 t_1}}{e^{-\lambda_1 L_1} - e^{-\lambda_1 R_1}} * \frac{\lambda_2 e^{-\lambda_2 t_2}}{e^{-\lambda_2 L_2} - e^{-\lambda_2 R_2}}$$

Now, considering the 6 cases, we calculate the $P(T_2 < T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2])$ for each case:

Case1 If $L_1 \leq L_2 \leq R_1 \leq R_2$, the integral over D of the joint density of (T_1, T_2) given $T_1 \in (L_1, R_1]$ and $T_2 \in (L_2, R_2]$:

$$\int_{L_2}^{R_1} \int_{L_2}^{t_1} \frac{\lambda_1 e^{-\lambda_1 t_1}}{e^{-\lambda_1 L_1} - e^{-\lambda_1 R_1}} \frac{\lambda_2 e^{-\lambda_2 t_2}}{e^{-\lambda_2 L_2} - e^{-\lambda_2 R_2}} \, dt_2 \, dt_1.$$

The numerator equals:

$$\int_{L_2}^{R_1} \lambda_1 e^{-\lambda_1 t_1} \left[-e^{-\lambda_2 t_2}\right]_{L_2}^{t_1} dt_1 = \int_{L_2}^{R_1} \lambda_1 e^{-\lambda_1 t_1} \left(e^{-\lambda_2 L_2} - e^{-\lambda_2 t_1}\right) dt_1 =$$

$$e^{-\lambda_2 L_2} \int_{L_2}^{R_1} \lambda_1 e^{-\lambda_1 t_1} dt_1 - \frac{\lambda_1}{\lambda_1+\lambda_2} \int_{L_2}^{R_1} (\lambda_1+\lambda_2) e^{-(\lambda_1+\lambda_2)t_1} dt_1 =$$

$$e^{-\lambda_2 L_2}(e^{-\lambda_1 L_2} - e^{-\lambda_1 R_1}) - \frac{\lambda_1}{\lambda_1+\lambda_2} \int_{L_2}^{R_1} (\lambda_1+\lambda_2) e^{-(\lambda_1+\lambda_2)t_1} dt_1 =$$

$$e^{-\lambda_2 L_2}(e^{-\lambda_1 L_2} - e^{-\lambda_1 R_1}) - \frac{\lambda_1}{\lambda_1+\lambda_2}(e^{-(\lambda_1+\lambda_2)L_2} - e^{-(\lambda_1+\lambda_2)R_1}) =$$

$$\frac{-\lambda_1}{\lambda_1+\lambda_2} e^{-\lambda_1 R_1}(e^{-\lambda_2 L_2} - e^{-\lambda_2 R_1}) + \frac{\lambda_2}{\lambda_1+\lambda_2} e^{-\lambda_2 L_2}(e^{-\lambda_1 L_2} - e^{-\lambda_1 R_1}).$$

So, the probability becomes:

$$P(T_2 < T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2]) = \frac{\frac{-\lambda_1}{\lambda_1+\lambda_2} e^{-\lambda_1 R_1}(e^{-\lambda_2 L_2} - e^{-\lambda_2 R_1}) + \frac{\lambda_2}{\lambda_1+\lambda_2} e^{-\lambda_2 L_2}(e^{-\lambda_1 L_2} - e^{-\lambda_1 R_1})}{(e^{-\lambda_1 L_1} - e^{-\lambda_1 R_1})(e^{-\lambda_2 L_2} - e^{-\lambda_2 R_2})} \quad (5)$$

Case 4 If $L_2 < L_1 \leq R_1 \leq R_2$, the integral over D of the joint density of (T_1, T_2) given $T_1 \in (L_1, R_1]$ and $T_2 \in (L_2, R_2]$:

$$\int_{L_1}^{R_1} \lambda_1 e^{-\lambda_1 t_1} \int_{L_2}^{t_1} \lambda_2 e^{-\lambda_2 t_2} dt_2\, dt_1 = e^{-\lambda_2 L_2}(e^{-\lambda_1 L_1} - e^{-\lambda_1 R_1}) - \frac{\lambda_1}{\lambda_1+\lambda_2}(e^{-(\lambda_1+\lambda_2)L_1} - e^{-(\lambda_1+\lambda_2)R_1}).$$

So, the probability becomes:

$$P(T_2 < T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2]) = \frac{e^{-\lambda_2 L_2}(e^{-\lambda_1 L_1} - e^{-\lambda_1 R_1}) - \frac{\lambda_1}{\lambda_1 + \lambda_2}(e^{-(\lambda_1+\lambda_2)L_1} - e^{-(\lambda_1+\lambda_2)R_1})}{(e^{-\lambda_1 L_1} - e^{-\lambda_1 R_1})(e^{-\lambda_2 L_2} - e^{-\lambda_2 R_2})} \tag{6}$$

Case 2 If $L_1 \leq L_2 \leq R_2 < R_1$:

For case 2, consider $T_3 \sim \exp(\lambda_3)$ and $T_4 \sim \exp(\lambda_4)$ with $T_3 \in (L_3, R_3], T_4 \in (L_4, R_4]$ and suppose that $L_3 \leq L_4 \leq R_4 < R_3$. Then:

$$P(T_3 < T_4 | T_3 \in (L_3, R_3], T_4 \in (L_4, R_4]) = 1 - P(T_4 < T_3 | T_3 \in (L_3, R_3], T_4 \in (L_4, R_4]).$$

Now take $\lambda_3 = \lambda_2, \lambda_4 = \lambda_1, L_3 = L_2, L_4 = L_1, R_3 = R_2, R_4 = R_1$. It can be seen that $P(T_4 < T_3 | T_3 \in (L_3, R_3], T_4 \in (L_4, R_4])$ is given by one minus the formula (6) where we now interchange all the subscripts 1 and 2.

So, the probability becomes:

$$P(T_2 < T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2]) = 1 - \frac{e^{-\lambda_1 L_1}(e^{-\lambda_2 L_2} - e^{-\lambda_2 R_2}) - \frac{\lambda_2}{\lambda_2 + \lambda_1}(e^{-(\lambda_1+\lambda_2)L_2} - e^{-(\lambda_1+\lambda_2)R_2})}{(e^{-\lambda_1 L_1} - e^{-\lambda_1 R_1})(e^{-\lambda_2 L_2} - e^{-\lambda_2 R_2})} \tag{7}$$

Case 5 If $L_2 < L_1 \leq R_2 < R_1$:

Using the same idea with case 2, in case 5 the $P(T_2 < T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2])$ is given by one minus formula (5) where we again interchange the subscripts 1 and 2.

So, the probability becomes:

$$P(T_2 < T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2]) = \tag{8}$$
$$1 - \frac{\frac{-\lambda_2}{\lambda_1+\lambda_2} e^{-\lambda_2 R_2}(e^{-\lambda_1 L_1} - e^{-\lambda_1 R_2}) + \frac{\lambda_1}{\lambda_1+\lambda_2} e^{-\lambda_1 L_1}(e^{-\lambda_2 L_1} - e^{-\lambda_2 R_2})}{(e^{-\lambda_1 L_1} - e^{-\lambda_1 R_1})(e^{-\lambda_2 L_2} - e^{-\lambda_2 R_2})}$$

Case 3 If $L_1 < R_1 \leq L_2 < R_2$:

In case 3, we have:

$$P(T_2 < T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2]) = 0 \tag{9}$$

as the intervals do not overlap and $R_1 \leq L_2$.

Case 6 If $L_2 < R_2 \leq L_1 < R_1$:

In case 6 (similarly to case 3), we have:

$$P(T_2 < T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2]) = 1 \tag{10}$$

as the intervals do not overlap and $R_2 \leq L_1$.

The corresponding R code can be found in the appendix.

3.2.2 General case

Ideas in Section 3.2.1, apply to general parametric models, if in Equation (3) and Equation (4), the appropriate density $f_{T_i}(t_i)$ and the survival probabilities $S_{T_i}(L_i)$ and $S_{T_i}(R_i)$ are used. It is very hard to give formulas like (5)-(8) for

general parametric distributions. It is, however, possible to obtain a Monte Carlo approximation to these probabilities. The steps are:

1. Given the hazard and the intervals for each individual, generate a large number (M) of time points $t_i^{(1)}, ..., t_i^{(M)}$ according to $f_{T_i|T_i \in (L_i, R_i]}$. Having specified the distribution of time (exponential, Weibull), we generate a large number of random draws from this distribution, assumed to be within the interval of time.

2. Define pairs of subjects (i, j) and for each pair, if the intervals are not overlapping:

   - If $R_1 \leq L_2$, the probability that the first individual will experience the event of interest earlier is 1
   - If $R_2 \leq L_1$, the probability that the first individual will experience the event of interest earlier is 0

3. For those individuals that their intervals overlap, the conditional probabilities will be used. To measure the probability that the first individual experiences the event of interest earlier, we compare the random draws of each pair. If the event happens earlier for the first individual, value 1 is given, otherwise 0. The mean of those values of the total number of the individuals that their intervals overlap gives the desired probability.

4. Finally, for those pairs that the hazard of the first individual is higher than the one of the second (the risk to experience the event is higher), the probabilities that the first individual will experience the event earlier are summed and divided by the total number of individuals that overlap.

The R code that describes the function for calculating the c-index in the general case is shown in the appendix.

4 Predictive accuracy

4.1 Brier score

The Brier score can be thought of as either a measure of the "calibration" of a set of probabilistic predictions, or as a "cost function". In order to compare the predictive abilities of the different risk scores, we calculate the Brier score. More precisely, across all items $i \in 1...n$ in a set of n predictions, the Brier score measures the mean squared difference between:

- The predicted probability assigned to the possible outcomes for item i
- The actual outcome O_i

The lower this deviation, the better the respective risk prediction model. The Brier score accounts for both discrimination (i.e. correct classification in different outcome groups) and calibration (agreement of the predictions with the true risk). This is a clear advantage over other common methods for prediction assessment, especially the receiver operating characteristic (ROC) curve which focuses on discrimination. Moreover, the Brier score is easy to interpret and is not dependent on an arbitrary definition of thresholds for the classification of individual risk scores to different risk groups.
Therefore, the lower the Brier score is for a set of predictions, the better the predictions are calibrated. Note that the Brier score, in its most common formulation, takes on a value between zero and one, since this is the largest possible difference between a predicted probability (which must be between zero and one) the actual outcome (which can take on values of only 0 and 1).

In the context of survival analysis, T_i is the time of the event of interest of subject i, t_0 is the exact follow-up time, the outcome is $O_i = \mathbb{1}\{T_i > t_0\}$, $\hat{S}(t_0|X_i)$ estimates the probability of observing a subject at risk at t_0, given the covariates X. The Brier score is defined as:

$$\text{BS}(t_0) = \text{E}(\mathbb{1}\{T_i > t_0\} - \hat{S}(t_0|X))^2$$

This formula can be written as:

$$\begin{gathered}\text{BS}(t_0) = \text{E}(\mathbb{1}\{T > t_0\} - \hat{S}(t_0|X))^2 = \\ \text{E}(\mathbb{1}\{T > t_0\} - S(t_0|X) - (\hat{S}(t_0|X) - S(t_0|X)))^2 = \\ \text{E}(\mathbb{1}\{T > t_0\} - S(t_0|X))^2 + \text{E}(\hat{S}(t_0|X) - S(t_0|X))^2\end{gathered}$$

In the case of **no censoring** the formula for Brier score is:

$$\widehat{\text{BS}}(t_0) = \frac{1}{n}\sum_{i=1}^{n}(\mathbb{1}\{T_i > t_0\} - \hat{S}(t_0|X_i))^2$$

In the presence of **right censored data**, we observe (t_i, δ_i), the indicator $\mathbb{1}\{T_i > t_0\}$ cannot be always computed and is thus unknown. The indicator $\mathbb{1}\{T_i > t_0\}$ equals 1 when subject i is known to have survived at least until t_0, equals 0 if the person died before t_0, otherwise the indicator $\mathbb{1}\{T_i > t_0\}$ is unknown if the person was censored before t_0, i.e. if $T_i < t_0$ and $d_i = 0$. To overcome this issue, Inverse Probability of Censoring Weighting (IPCW) may be used, in the Liu et al. (2012) version of the c-index mentioned in Section 3.1:

$$\widehat{\text{BS}}(t_0) = \frac{1}{n}\sum_{i=1}^{n}(\mathbb{1}\{T_i > t_0\} - \hat{S}(t_0|X_i))^2 w_i$$

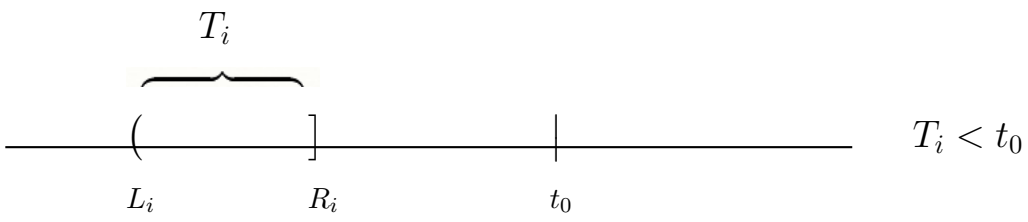
where

$$w_i = \begin{cases} 0, & \text{if } T_i < t_0 \text{ and } d_i = 0 \\ \dfrac{1}{\widehat{\text{G}}(t_0)}, & \text{if } T_i > t_0 \\ \dfrac{1}{\widehat{\text{G}}(T_i)}, & \text{if } T_i < t_0 \text{ and } d_i = 1 \end{cases}$$

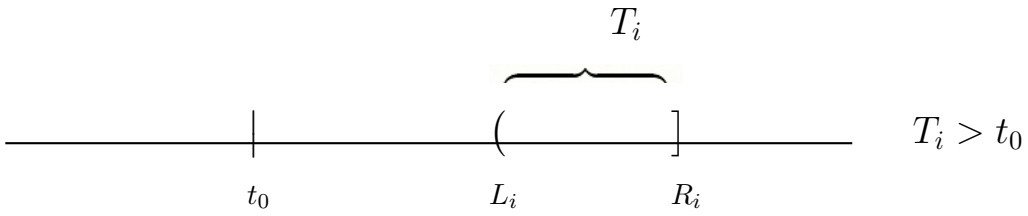
4.2 Brier score and interval censoring

In the case of **interval censored data**, where the observed outcome of subject i is that $T_i \in (L_i, R_i]$, the contribution of subject i in Brier score at time t_0 is one of the three possibilities:

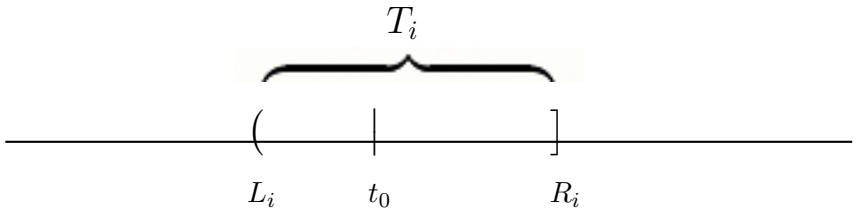
1. $R_i < t_0 \Rightarrow T_i \leq t_0$ with certainty $\Rightarrow \mathbb{1}\{T_i > t_0\} = 0$



2. $L_i \geq t_0 \Rightarrow T_i > t_0$ with certainty $\Rightarrow \mathbb{1}\{T_i > t_0\} = 1$



3. $L_i < t_0 \leq R_i$, so it is uncertain whether $T_i > t_0$, and $\mathbb{1}\{T_i > t_0\}$ cannot be determined with certainty



If we define

$$\widehat{\text{BS}}(t_0) = \frac{1}{n}\sum_{i=1}^{n} \widehat{\text{BS}}(t_0|X_i),$$

with $\widehat{\text{BS}}(t_0|X_i) = (\mathbb{1}\{T_i > t_0\} - \hat{S}(t_0|X_i))^2$,

this implies the following for these three possibilities:

1. $R_i < t_0 \Rightarrow \mathbb{1}\{T_i > t_0\} = 0 \Rightarrow \widehat{\text{BS}}(t_0|X_i) = (0 - \hat{S}(t_0|X_i))^2$;
2. $L_i \geq t_0 \Rightarrow \mathbb{1}\{T_i > t_0\} = 1 \Rightarrow \widehat{\text{BS}}(t_0|X_i) = (1 - \hat{S}(t_0|X_i))^2$;
3. $L_i < t_0 \leq R_i$, the indicator $\mathbb{1}\{T_i > t_0\}$ is unknown.

In the latter case, to compute the Brier score, we will use the following identity for the Brier score:

$$\begin{aligned}
\text{BS}(t_0) &= E(\mathbb{1}\{T > t_0\} - \hat{S}(t_0|X))^2 \\
&= E(\mathbb{1}\{T > t_0\})^2 - 2E\mathbb{1}\{T > t_0\}\hat{S}(t_0|X) + E\hat{S}^2(t_0|X) \\
&= E\mathbb{1}\{T > t_0\} - 2E\mathbb{1}\{T > t_0\}\hat{S}(t_0|X) + E\hat{S}^2(t_0|X),
\end{aligned}$$

and in $\widehat{\text{BS}}(t_0|X_i) = \mathbb{1}\{T_i > t_0\} - 2\mathbb{1}\{T_i > t_0\}\hat{S}(t_0|X_i) + \hat{S}^2(t_0|X_i)$ we will

replace $\mathbb{1}\{T_i > t_0\}$ by

$$E(\mathbb{1}\{T_i > t_0\}|T_i \in (L_i, R_i], X_i) = \mathrm{P}(T_i > t_0|T_i \in (L_i, R_i], X_i),$$

leading to $\widehat{\mathrm{BS}}(t_0|X_i) = \widehat{\mathrm{P}}(T_i > t_0|T_i \in (L_i, R_i], X_i)$. To calculate this probability, note that

$$\widehat{\mathrm{P}}(T_i > t_0|T_i \in (L_i, R_i], X_i) = \frac{\widehat{\mathrm{P}}(t_0 < T_i \leq R_i|X_i)}{\widehat{\mathrm{P}}(L_i < T_i \leq R_i|X_i)} = \frac{\hat{S}(t_0|X_i) - \hat{S}(R_i|X_i)}{\hat{S}(L_i|X_i) - \hat{S}(R_i|X_i)}$$

where $\hat{S}(t|X_i)$ is calculated assuming that T_i follows an exponential or a Weibull distribution (taking into account the different shape), with rate parameter $\hat{\lambda}_i$ depending on X_i.

5 Simulation study

In this chapter, the performance of the proposed methods (c-index and Brier score) for interval censored data is evaluated by means of a simulation study.

Definition

Simulation is a way to model random events, such that simulated outcomes closely match real-world outcomes. By observing simulated outcomes, researchers gain insight on the real world.

The simulation study will be loosely mimicked after the data set introduced in Chapter 1 that will also be used as illustration of the methods on real data in Chapter 7. A Monte Carlo approximation will be explored under different scenarios (shape of the Weibull distribution). The true value of the concordance index and the true value of Brier score in different time points will be calculated, and this creates the opportunity to study bias and mean squared error of the proposed measures, and its dependence on the underlying model and the observation scheme (length of intervals).

5.1 Data simulation

The following steps are used to simulate the data:

Step 1 Generate the covariates $X_i = (X_{i1}, ..., X_{ip})$ where $i = 1, 2, ..., n$:

$$\begin{cases} X_{i1} \sim N(0, 10), & \text{where } X_{i1} \text{ is the continuous covariate of age} \\ X_{i2}, & \text{generate a random sample of two values (0 or 1) with the same probability to form the covariate of sex} \\ X_{i3}, X_{i4} & \text{generate a random sample of three values (1 to 3) with probabilities 0.40, 0.35 and 0.25, respectively and form the dummies of the covariate pdiag} \end{cases}$$

It is important to mention that all covariates are centered to have mean 0, so from X_{i2}, X_{i3} and X_{i4}, the values 0.5, 0.35 and 0.25, respectively, are subtracted.

Step 2 Different values of the shape parameter γ were chosen, as γ can have marked effects on the behavior of the distribution. Weibull distributions with $\gamma < 1$ have a failure rate that decreases with time. Weibull distributions with γ close to or equal to 1 have a fairly constant failure rate and Weibull distributions with $\gamma > 1$ have a failure rate that increases with time. The three different values that are chosen for shape are: $\gamma = 1$, $\gamma = 0.5 < 1$ and $\gamma = 2 > 1$.
For given shape of γ, choose baseline λ_0 so that the mean survival of the Weibull at the mean of the covariates equals 10. The formula to calculate the mean $\mathrm{E}(X)$ in the Weibull distribution is:

$$\frac{\Gamma(1+1/\gamma)}{\lambda_0^{1/\gamma}}$$

in which it is needed to be 10 and $\lambda_0 = \exp(\beta_0)$. So:

$$\frac{\Gamma(1+1/\gamma)}{\exp(\beta_0/\gamma)} = 10 \Rightarrow \beta_0 = \log(\frac{\Gamma(1+1/\gamma)}{10})\gamma$$

Step 3 For $i = 1, ..., n$, generate t_i from $W(\lambda_i, \gamma)$, with $\lambda_i = \lambda_0 \exp(\beta_1 x_{i1} + ... + \beta_p x_{ip}) = \exp(\beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip})$. These are the exact times of the event of interest for each individual.

Step 4 Apply interval censoring, so that not t_i is observed, but only the interval in which it lies. For simplicity, only equally spaced observation intervals will be used, with length (*by* =) 1, 3 or 5 time units, starting at 0 and ending at 30. Event times $t_i > 30$ are observed to be lying in $(30, \infty]$, so are right censored at 30.

Step 5 This will generate an interval censored data set of the form $(L_i, R_i]$ and X_i, for $i = 1, ..., n$, on which each of the following methods of calculating c-index and Brier scores will be applied.

Step 6 Application of methods. For all of the methods, first the parameters β, λ_0 and γ will be estimated using AFT models through *survreg* function in R (Weibull regression). After reparametrization to PH model (see Section 2.4), this leads to estimates $\widehat{\lambda}_0, \widehat{\beta}$ and $\widehat{\gamma}$, and to estimates $\widehat{\lambda}_i = \widehat{\lambda}_0 \exp(\widehat{\beta}_1 X_{i1} + ... + \widehat{\beta}_p X_{ip})$.

- C-index: Calculate the c-index using both the exact (Section 3.2.1) and the general (Section 3.2.2) methods.

- Brier score: Discrete time points ($t = 1.5, 4.5, 7.5, ..., 28.5$) lying in the observation intervals are selected. The Brier score is calculated for the interval censored data as described in Section 4.2.

Steps 1 – 6 are repeated for $m = 1, \ldots, M = 1000$ replications. All estimates of c-index and Brier score are stored.

5.2 Bias and RMSE

The goal of the simulation study is to test whether the ideas discussed in the thesis are applicable to our data. So, the bias and the root-mean-square-error are considered in the simulation results.

A statistic is biased if, in the long run, it consistently over or underestimates the parameter it is estimating. More technically it is biased if its expected value is not equal to the parameter. A statistic is positively biased if it tends to overestimate the parameter; a statistic is negatively biased if it tends to underestimate the parameter. An unbiased statistic is not necessarily an accurate statistic. If a statistic is sometimes much too high and sometimes much too low, it can still be unbiased. It would be very imprecise, however. A slightly biased statistic that systematically results in very small overestimates of a parameter could be quite efficient.

The mean squared error (MSE) is the second moment (about the true value) of the error, and thus incorporates both the variance of the estimator and its bias.

$$\text{MSE}(\hat{\theta}) = E(\hat{\theta} - \theta)^2 = \text{var}(\hat{\theta}) + (\text{Bias}(\hat{\theta}, \theta))^2$$

The root mean squared error (RMSE) of an estimator $\hat{\theta}$ with respect to an estimated parameter θ is defined as the square root of the mean square error:

$$\text{RMSE}(\hat{\theta}) = \sqrt{\text{MSE}(\hat{\theta})} = \sqrt{E(\hat{\theta} - \theta)^2}$$

For an unbiased estimator, the RMSE is the square root of the variance, known as the standard error.
To calculate the bias and the RMSE, the parameter θ, which indicates the 'true value' of each measure is needed. For the c-index and Brier score under the simulation set up of Section 5.1, this is very hard, if not impossible, to calculate mathematically. For this reason, a Monte Carlo approximation was used. Generate a single huge data set of 1000000 subjects under the simulation

set up of Section 5.1, retaining the uncensored event times t_i, and for:

c-index assuming that observe all of them, calculate the proportion of concordant pairs among all pairs of subjects.

Brier score calculate the mean squared difference between $\mathbb{1}\{T_i > t_0\}$ and $\hat{S}(t_0|X_i)$ at different discrete time points ($t_0 = 1.5, 4.5, 7.5, ..., 28.5$).

The results of the Monte Carlo approximation (referred to from now on as "true" values) are shown in Table 2.

	Shape=1	**Shape=0.5**	**Shape=2**
c-index	0.5487	0.5493	0.5479
Brier score			
$time$			
1.5	0.1203	0.2411	0.0174
4.5	0.2293	0.2332	0.1261
7.5	0.2455	0.2041	0.2278
10.5	0.2236	0.1773	0.2397
13.5	0.1891	0.1545	0.1795
16.5	0.1538	0.1355	0.1049
19.5	0.1223	0.1196	0.0510
22.5	0.0958	0.1063	0.0214
25.5	0.0743	0.0951	0.0077
28.5	0.0573	0.0854	0.0025

Table 2: Monte Carlo approximation for c-index and Brier score

It is clear that the c-index is almost independent of the shape of the Weibull distribution, while the Brier score depends a lot on the shape of the Weibull distribution.

5.3 Simulation results

The simulation results for the c-index are shown in Table 3, while for Brier score in Tables 4-6, where "exact" stands for the c-index in the exponential case described in Section 3.2.1 and "sim" for the general case introduced in Section 3.2.2.
Although flexible and often insightful, Monte Carlo studies are limited by their

finite nature, and as such are subject to sampling variability. When a simulation is run more than once, different results are obtained. This is called between-simulation variability Monte Carlo error (MCE) and should be taken into account when interpreting the simulation results.

c-index results In general, both methods, exact and sim, provide almost unbiased estimates. The root mean squared error is mostly higher for sim method, except for shape=0.5. In addition, when n increases, both bias and RMSE are smaller (especially the RMSE). The shape and the observational intervals (by=1,3,5) seem not to have an important effect on the results of c-index, contrary to expectation.

Brier score results Principally, the accuracy of the measurements is unbiased with low variation. The estimates are more unbiased and have smaller RMSE at the beginning and at the end of the observation time as the survival probability is closer to more subjects experience the event of interest in the meanwhile. The shape appears to influence the ”true values” of the Brier score given in Table 2.

	Shape=1			**Shape=0.5**			**Shape=2**		
	$n = 250$	$n = 500$	$n = 1000$	$n = 250$	$n = 500$	$n = 1000$	$n = 250$	$n = 500$	$n = 1000$
$by = 1$									
Bias exact	0.0052	0.0015	-0.0008	0.0118	0.0066	0.0040	0.0014	-0.0016	-0.0035
RMSE exact	0.0194	0.0134	0.0099	0.0226	0.0152	0.0110	0.0177	0.0128	0.0102
Bias sim	0.0093	0.0054	0.0029	0.0099	0.0048	0.0023	0.0092	0.0055	0.0033
RMSE sim	0.0218	0.0151	0.0108	0.0212	0.0142	0.0102	0.0215	0.0151	0.0111
$by = 3$									
Bias exact	0.0066	0.0014	-0.0016	0.0123	0.0062	0.0038	0.0028	-0.0015	-0.0043
RMSE exact	0.0203	0.0135	0.0098	0.0235	0.0159	0.0109	0.0185	0.0132	0.0104
Bias sim	0.0108	0.0054	0.0020	0.0104	0.0044	0.0021	0.0106	0.0056	0.0024
RMSE sim	0.0228	0.0151	0.0104	0.0222	0.0150	0.0102	0.0227	0.0155	0.0108
$by = 5$									
Bias exact	0.0055	0.0006	-0.0011	0.0122	0.0064	0.0038	0.0030	-0.0007	-0.0034
RMSE exact	0.0194	0.0133	0.0095	0.0230	0.0156	0.0108	0.0179	0.0127	0.0100
Bias sim	0.0098	0.0046	0.0025	0.0102	0.0046	0.0021	0.0108	0.0065	0.0035
RMSE sim	0.0219	0.0147	0.0103	0.0215	0.0147	0.0102	0.0223	0.0153	0.0110

Table 3: Simulation results for c-index

Shape=1	**n=250**		**n=500**		**n=1000**	
	Bias	RMSE	Bias	RMSE	Bias	RMSE
$by = 1$						
$time$						
1.5	-0.0011	0.0142	0.0001	0.0103	0.0001	0.0072
4.5	-0.0021	0.0082	-0.0010	0.0058	-0.0004	0.0039
7.5	-0.0027	0.0054	-0.0015	0.0034	-0.0008	0.0023
10.5	-0.0026	0.0091	-0.0016	0.0069	-0.0009	0.0047
13.5	-0.0022	0.0125	-0.0012	0.0092	-0.0006	0.0064
16.5	-0.0018	0.0142	-0.0008	0.0100	-0.0006	0.0072
19.5	-0.0015	0.0146	-0.0008	0.0104	-0.0006	0.0074
22.5	-0.0010	0.0141	-0.0008	0.0102	-0.0003	0.0072
25.5	-0.0008	0.0132	-0.0005	0.0094	-0.0001	0.0068
28.5	-0.0004	0.0125	-0.0003	0.0085	-0.0000	0.0062
$by = 3$						
$time$						
1.5	0.0000	0.0144	-0.0005	0.0106	-0.0000	0.0074
4.5	-0.0019	0.0082	-0.0012	0.0058	-0.0004	0.0041
7.5	-0.0032	0.0058	-0.0014	0.0034	-0.0006	0.0021
10.5	-0.0034	0.0099	-0.0013	0.0070	-0.0006	0.0048
13.5	-0.0029	0.0127	-0.0008	0.0094	-0.0005	0.0066
16.5	-0.0022	0.0139	-0.0004	0.0102	-0.0004	0.0072
19.5	-0.0014	0.0142	-0.0004	0.0101	-0.0004	0.0073
22.5	-0.0009	0.0138	-0.0005	0.0096	-0.0004	0.0070
25.5	-0.0004	0.0130	-0.0004	0.0093	-0.0003	0.0066
28.5	-0.0001	0.0120	-0.0003	0.0087	-0.0002	0.0060
$by = 5$						
$time$						
1.5	-0.0009	0.0145	-0.0000	0.0104	0.0021	0.0073
4.5	-0.0022	0.0086	-0.0008	0.0056	-0.0046	0.0039
7.5	-0.0030	0.0055	-0.0012	0.0032	-0.0075	0.0021
10.5	-0.0033	0.0100	-0.0013	0.0063	-0.0079	0.0046
13.5	-0.0028	0.0127	-0.0010	0.0087	-0.0045	0.0063
16.5	-0.0021	0.0138	-0.0009	0.0100	-0.0029	0.0072
19.5	-0.0017	0.0142	-0.0007	0.0104	-0.0019	0.0074
22.5	-0.0017	0.0138	-0.0002	0.0103	-0.0014	0.0072
25.5	-0.0014	0.0129	-0.0000	0.0096	0.0000	0.0066
28.5	-0.0011	0.0120	0.0001	0.0088	0.0040	0.0061

Table 4: Simulation results for Brier score, shape=1

Shape=0.5	**n=250**		**n=500**		**n=1000**	
	Bias	RMSE	Bias	RMSE	Bias	RMSE
$by = 1$						
$time$						
1.5	-0.0028	0.0066	-0.0013	0.0042	-0.0005	0.0030
4.5	-0.0031	0.0082	-0.0015	0.0051	-0.0009	0.0038
7.5	-0.0029	0.0116	-0.0015	0.0079	-0.0011	0.0058
10.5	-0.0026	0.0133	-0.0013	0.0095	-0.0010	0.0068
13.5	-0.0020	0.0144	-0.0010	0.0101	-0.0008	0.0072
16.5	-0.0017	0.0151	-0.0008	0.0105	-0.0007	0.0074
19.5	-0.0013	0.0152	-0.0005	0.0105	-0.0006	0.0075
22.5	-0.0010	0.0149	-0.0005	0.0105	-0.0005	0.0073
25.5	-0.0009	0.0147	-0.0005	0.0103	-0.0005	0.0072
28.5	-0.0010	0.0143	-0.0004	0.0099	-0.0005	0.0071
$by = 3$						
$time$						
1.5	-0.0032	0.0070	-0.0012	0.0043	-0.0006	0.0030
4.5	-0.0030	0.0083	-0.0018	0.0056	-0.0008	0.0036
7.5	-0.0027	0.0121	-0.0017	0.0084	-0.0009	0.0057
10.5	-0.0024	0.0140	-0.0015	0.0098	-0.0007	0.0067
13.5	-0.0017	0.0150	-0.0012	0.0104	-0.0004	0.0072
16.5	-0.0013	0.0154	-0.0011	0.0107	-0.0002	0.0074
19.5	-0.0008	0.0152	-0.0008	0.0106	-0.0001	0.0074
22.5	-0.0006	0.0149	-0.0007	0.0106	-0.0002	0.0075
25.5	-0.0008	0.0146	-0.0008	0.0104	-0.0002	0.0075
28.5	-0.0009	0.0143	-0.0010	0.0102	-0.0002	0.0073
$by = 5$						
$time$						
1.5	-0.0029	0.0068	-0.0015	0.0046	-0.0006	0.0030
4.5	-0.0034	0.0087	-0.0015	0.0054	-0.0006	0.0036
7.5	-0.0031	0.0123	-0.0015	0.0082	-0.0003	0.0056
10.5	-0.0028	0.0140	-0.0011	0.0096	-0.0001	0.0067
13.5	-0.0023	0.0146	-0.0008	0.0102	-0.0000	0.0071
16.5	-0.0019	0.0148	-0.0006	0.0106	-0.0000	0.0073
19.5	-0.0016	0.0148	-0.0003	0.0108	-0.0000	0.0074
22.5	-0.0015	0.0147	-0.0002	0.0107	-0.0002	0.0074
25.5	-0.0014	0.0145	-0.0003	0.0105	-0.0003	0.0072
28.5	-0.0013	0.0142	-0.0003	0.0102	-0.0003	0.0070

Table 5: Simulation results for Brier score, shape=0.5

Shape=2	**n=250**		**n=500**		**n=1000**	
	Bias	RMSE	Bias	RMSE	Bias	RMSE
$by = 1$						
$time$						
1.5	-0.0001	0.0050	-0.0001	0.0034	-0.0000	0.0024
4.5	-0.0011	0.0138	-0.0006	0.0094	-0.0002	0.0070
7.5	-0.0020	0.0085	-0.0009	0.0056	-0.0005	0.0041
10.5	-0.0027	0.0067	-0.0014	0.0043	-0.0007	0.0029
13.5	-0.0026	0.0137	-0.0013	0.0087	-0.0005	0.0064
16.5	-0.0014	0.0144	-0.0005	0.0096	-0.0002	0.0070
19.5	-0.0004	0.0115	-0.0003	0.0078	-0.0002	0.0054
22.5	-0.0003	0.0077	-0.0002	0.0055	-0.0002	0.0037
25.5	0.0000	0.0047	0.0000	0.0034	0.0001	0.0023
28.5	0.0000	0.0026	0.0000	0.0019	0.0000	0.0013
$by = 3$						
$time$						
1.5	-0.0001	0.0049	-0.0002	0.0034	-0.0000	0.0024
4.5	-0.0007	0.0137	-0.0010	0.0095	-0.0002	0.0068
7.5	-0.0019	0.0086	-0.0012	0.0057	-0.0004	0.0040
10.5	-0.0032	0.0070	-0.0013	0.0042	-0.0006	0.0029
13.5	-0.0032	0.0133	-0.0012	0.0088	-0.0008	0.0063
16.5	-0.0017	0.0139	-0.0005	0.0098	-0.0003	0.0067
19.5	-0.0011	0.0110	-0.0003	0.0079	-0.0001	0.0056
22.5	-0.0005	0.0074	-0.0002	0.0054	-0.0000	0.0038
25.5	0.0000	0.0045	0.0001	0.0033	0.0001	0.0024
28.5	0.0000	0.0026	0.0000	0.0019	0.0000	0.0014
$by = 5$						
$time$						
1.5	-0.0000	0.0048	-0.0000	0.0034	-0.0000	0.0024
4.5	-0.0008	0.0138	-0.0004	0.0093	-0.0003	0.0067
7.5	-0.0022	0.0086	-0.0011	0.0057	-0.0005	0.0040
10.5	-0.0028	0.0067	-0.0015	0.0044	-0.0007	0.0029
13.5	-0.0022	0.0126	-0.0013	0.0090	-0.0007	0.0061
16.5	-0.0008	0.0137	-0.0003	0.0097	-0.0001	0.0066
19.5	-0.0003	0.0110	-0.0001	0.0077	-0.0001	0.0053
22.5	-0.0000	0.0076	0.0000	0.0052	-0.0001	0.0037
25.5	0.0003	0.0048	0.0002	0.0032	0.0001	0.0023
28.5	0.0002	0.0028	0.0001	0.0018	0.0000	0.0013

Table 6: Simulation results for Brier score, shape=2

6 Cross-validation

The basic idea of the discrimination and prediction error measures studied in this thesis is to quantify how well the prediction model will predict survival of future patients. Typically the prediction model is developed on a "training" data set. When the same data are used to validate the model, the danger is real that the results of the discrimination and prediction error measures give an over-optimistic view. Especially in the case of over-fitting the difference between predictive performance on the same data and on external data can be substantial.
To overcome this problem, one possibility is to split the data in two parts and use one part of the data as a training set to build the model and reserve another part as validation set to validate the model. Especially when the original data are not really large, this is rather wasteful. We will therefore use leave-one-out cross-validation to correct for over-optimism in our predictive accuracy measures. Leave-one-out cross-validation is essentially an estimate of the generalisation performance of a model trained on $n-1$ samples of data.
The procedure is as follows:

1. For $i = 1, \ldots, n$, use the data with subject i removed to fit the exponential or Weibull regression model, leading to estimates $\hat{\lambda}_0^{(-i)}$, $\hat{\beta}_1^{(-i)}, \ldots, \hat{\beta}_p^{(-i)}$, and $\hat{\gamma}^{(-i)}$ of baseline rate λ_0, regression coefficients $\beta_1, \ldots, \beta_p$ and shape parameter γ. For the exponential model, γ is not estimated but set to 1.

2. Define the cross-validated rates $\hat{\lambda}^{(-i)} = \hat{\lambda}_0^{(-i)} \exp(\hat{\beta}_1^{(-i)} x_{i1} + \ldots + \hat{\beta}_p^{(-i)} x_{ip})$ and shape $\hat{\gamma}^{(-i)}$ for subject i.

3. **c-index** All $f_{T_i|T_i \in (L_i, R_i]}(t_i)$'s are now based on Weibull distributions with rate $\hat{\lambda}^{(-i)}$ and shape $\hat{\gamma}^{(-i)}$. The prognostic scores, used for predicting the order of dying, are based on the leave-one-out values defined by $\hat{z}^{(-i)} = \hat{\beta}_1^{(-i)} x_{i1} + \ldots + \hat{\beta}_p^{(-i)} x_{ip} = \hat{\beta}^{(-i)^T} x_i$. Then the c-index is calculated according to the formulas described in Section 3.2.

   Brier score Here $P(T_i > t_0 | T_i \in (L_i, R_i])$ is based on Weibull distributions with rate $\hat{\lambda}^{(-i)}$ and shape $\hat{\gamma}^{(-i)}$. Given the rate and the shape, leaving out subject i each time, the Brier score is calculated as described in Section 4.2.

The simulation results for the c-index with cross-validation and without are shown in Table 7 and similarly for Brier score in Tables 8-10. These simulation results are based on the simulation set-up of Chapter 5. Reported are the mean

values of $M = 1000$ replications.

CV results: The cross-validated c-index values are always lower than those without using cross-validation. The cross-validated Brier scores are generally higher than those without using cross-validation. This indicates that the cross-validated results give a more modest, and probably a more realistic, idea of the discriminative ability and of the predictive accuracy of the model to new data. The difference between cross-validated and non-cross-validated c-index values and Brier scores becomes smaller with larger sample sizes. This makes sense, since the degree of overfitting is usually smaller for larger sample sizes.

In Figures 6.1-6.3, estimates of Brier score are shown with/without cross-validation at a set of time points, for n=250, by=5, in the three different shapes used. The dashed line, representing the cross-validated Brier score, is always slightly higher as expected.

	Shape=1			**Shape=0.5**			**Shape=2**		
	$n = 250$	$n = 500$	$n = 1000$	$n = 250$	$n = 500$	$n = 1000$	$n = 250$	$n = 500$	$n = 1000$
$by = 1$									
c-exact	0.5540	0.5503	0.5479	0.5611	0.5559	0.5533	0.5494	0.5463	0.5444
c-sim	0.5581	0.5542	0.5516	0.5592	0.5541	0.5516	0.5571	0.5535	0.5513
c-exact CV	0.5137	0.5292	0.5379	0.5229	0.5368	0.5441	0.5042	0.5231	0.5334
c-sim CV	0.5149	0.5316	0.5409	0.5202	0.5346	0.5422	0.5079	0.5281	0.5392
$by = 3$									
c-exact	0.5554	0.5502	0.5471	0.5617	0.5555	0.5531	0.5508	0.5464	0.5436
c-sim	0.5596	0.5542	0.5508	0.5597	0.5537	0.5515	0.5586	0.5536	0.5504
c-exact CV	0.5156	0.5287	0.5369	0.5241	0.5354	0.5439	0.5058	0.5232	0.5325
c-sim CV	0.5170	0.5312	0.5399	0.5214	0.5332	0.5420	0.5097	0.5282	0.5382
$by = 5$									
c-exact	0.5543	0.5494	0.5476	0.5615	0.5557	0.5531	0.5510	0.5472	0.5445
c-sim	0.5586	0.5534	0.5513	0.5595	0.5540	0.5514	0.5588	0.5545	0.5515
c-exact	0.5134	0.5283	0.5376	0.5234	0.5361	0.5437	0.5073	0.5248	0.5336
c-sim CV	0.5148	0.5307	0.5407	0.5205	0.5339	0.5419	0.5112	0.5299	0.5394

Table 7: Simulation results for c-index (the mean value) by applying cross-validation versus without

Shape=1	**n=250**		**n=500**		**n=1000**	
	Brier	BrierCV	Brier	BrierCV	Brier	BrierCV
$by = 1$						
$time$						
1.5	0.1192	0.1198	0.1205	0.1208	0.1204	0.1206
4.5	0.2271	0.2314	0.2282	0.2304	0.2288	0.2299
7.5	0.2427	0.2484	0.2439	0.2467	0.2446	0.2460
10.5	0.2209	0.2266	0.2220	0.2248	0.2226	0.2241
13.5	0.1868	0.1919	0.1878	0.1903	0.1884	0.1896
16.5	0.1519	0.1561	0.1529	0.1550	0.1531	0.1542
19.5	0.1208	0.1240	0.1214	0.1230	0.1216	0.1224
22.5	0.0947	0.0972	0.0949	0.0961	0.0954	0.0960
25.5	0.0735	0.0753	0.0737	0.0746	0.0742	0.0746
28.5	0.0568	0.0582	0.0569	0.0576	0.0572	0.0575
$by = 3$						
$time$						
1.5	0.1204	0.1210	0.1198	0.1201	0.1203	0.1204
4.5	0.2273	0.2317	0.2280	0.2301	0.2288	0.2298
7.5	0.2422	0.2479	0.2440	0.2468	0.2448	0.2462
10.5	0.2201	0.2258	0.2223	0.2251	0.2229	0.2243
13.5	0.1862	0.1912	0.1883	0.1908	0.1886	0.1898
16.5	0.1516	0.1557	0.1534	0.1555	0.1534	0.1544
19.5	0.1208	0.1240	0.1218	0.1234	0.1219	0.1227
22.5	0.0949	0.0973	0.0952	0.0965	0.0954	0.0960
25.5	0.0739	0.0757	0.0739	0.0748	0.0739	0.0744
28.5	0.0571	0.0584	0.0569	0.0575	0.0570	0.0574
$by = 5$						
$time$						
1.5	0.1193	0.1200	0.1203	0.1206	0.1205	0.1207
4.5	0.2270	0.2313	0.2284	0.2306	0.2288	0.2298
7.5	0.2425	0.2481	0.2442	0.2470	0.2447	0.2461
10.5	0.2202	0.2259	0.2222	0.2251	0.2228	0.2242
13.5	0.1862	0.1912	0.1881	0.1906	0.1886	0.1899
16.5	0.1516	0.1558	0.1529	0.1549	0.1535	0.1545
19.5	0.1205	0.1237	0.1216	0.1232	0.1221	0.1229
22.5	0.0940	0.0965	0.0955	0.0967	0.0956	0.0962
25.5	0.0728	0.0747	0.0742	0.0751	0.0744	0.0748
28.5	0.0561	0.0574	0.0574	0.0580	0.0577	0.0580

Table 8: Simulation results for Brier score by applying cross-validation for shape=1

Shape=0.5	**n=250**		**n=500**		**n=1000**	
	Brier	BrierCV	Brier	BrierCV	Brier	BrierCV
$by = 1$						
$time$						
1.5	0.2382	0.2430	0.2398	0.2421	0.2405	0.2417
4.5	0.2301	0.2365	0.2317	0.2348	0.2322	0.2338
7.5	0.2012	0.2070	0.2026	0.2054	0.2030	0.2045
10.5	0.1746	0.1798	0.1759	0.1784	0.1762	0.1775
13.5	0.1524	0.1569	0.1534	0.1556	0.1536	0.1547
16.5	0.1338	0.1377	0.1347	0.1366	0.1348	0.1357
19.5	0.1182	0.1217	0.1190	0.1207	0.1190	0.1198
22.5	0.1052	0.1083	0.1057	0.1073	0.1058	0.1065
25.5	0.0941	0.0968	0.0946	0.0959	0.0945	0.0952
28.5	0.0844	0.0868	0.0850	0.0862	0.0848	0.0854
$by = 3$						
$time$						
1.5	0.2378	0.2426	0.2398	0.2422	0.2405	0.2416
4.5	0.2302	0.2366	0.2314	0.2346	0.2324	0.2339
7.5	0.2014	0.2072	0.2024	0.2053	0.2032	0.2047
10.5	0.1748	0.1799	0.1758	0.1783	0.1766	0.1778
13.5	0.1527	0.1572	0.1532	0.1554	0.1541	0.1552
16.5	0.1342	0.1382	0.1344	0.1363	0.1353	0.1362
19.5	0.1187	0.1222	0.1188	0.1205	0.1194	0.1203
22.5	0.1056	0.1087	0.1056	0.1071	0.1060	0.1068
25.5	0.0942	0.0969	0.0942	0.0955	0.0948	0.0955
28.5	0.0845	0.0869	0.0844	0.0856	0.0851	0.0857
$by = 5$						
$time$						
1.5	0.2381	0.2429	0.2396	0.2419	0.2404	0.2416
4.5	0.2298	0.2362	0.2317	0.2348	0.2326	0.2342
7.5	0.2010	0.2068	0.2026	0.2055	0.2038	0.2052
10.5	0.1744	0.1795	0.1761	0.1787	0.1771	0.1783
13.5	0.1521	0.1566	0.1536	0.1558	0.1544	0.1555
16.5	0.1335	0.1374	0.1348	0.1368	0.1354	0.1364
19.5	0.1179	0.1213	0.1192	0.1210	0.1195	0.1204
22.5	0.1048	0.1078	0.1060	0.1076	0.1061	0.1068
25.5	0.0936	0.0963	0.0948	0.0961	0.0947	0.0954
28.5	0.0841	0.0865	0.0850	0.0862	0.0851	0.0857

Table 9: Simulation results for Brier score by applying cross-validation for shape=0.5

Shape=2	**n=250**		**n=500**		**n=1000**	
	Brier	BrierCV	Brier	BrierCV	Brier	BrierCV
$by = 1$						
$time$						
1.5	0.0172	0.0171	0.0172	0.0172	0.0174	0.0173
4.5	0.1249	0.1260	0.1254	0.1260	0.1258	0.1261
7.5	0.2257	0.2297	0.2269	0.2288	0.2273	0.2282
10.5	0.2370	0.2425	0.2383	0.2410	0.2390	0.2403
13.5	0.1769	0.1813	0.1782	0.1804	0.1789	0.1801
16.5	0.1034	0.1058	0.1043	0.1055	0.1046	0.1052
19.5	0.0505	0.0513	0.0506	0.0510	0.0507	0.0509
22.5	0.0211	0.0212	0.0211	0.0212	0.0211	0.0212
25.5	0.0078	0.0078	0.0078	0.0077	0.0078	0.0078
28.5	0.0025	0.0025	0.0025	0.0025	0.0026	0.0026
$by = 3$						
$time$						
1.5	0.0173	0.0171	0.0171	0.0171	0.0173	0.0173
4.5	0.1253	0.1264	0.1250	0.1256	0.1258	0.1261
7.5	0.2259	0.2299	0.2266	0.2285	0.2274	0.2283
10.5	0.2365	0.2420	0.2384	0.2411	0.2391	0.2404
13.5	0.1763	0.1807	0.1783	0.1805	0.1787	0.1798
16.5	0.1031	0.1055	0.1043	0.1055	0.1045	0.1051
19.5	0.0498	0.0506	0.0506	0.0511	0.0508	0.0511
22.5	0.0208	0.0210	0.0211	0.0212	0.0213	0.0213
25.5	0.0077	0.0077	0.0078	0.0078	0.0078	0.0078
28.5	0.0025	0.0025	0.0026	0.0026	0.0026	0.0026
$by = 5$						
$time$						
1.5	0.0173	0.0171	0.0173	0.0172	0.0173	0.0173
4.5	0.1252	0.1263	0.1257	0.1262	0.1258	0.1260
7.5	0.2256	0.2296	0.2267	0.2286	0.2273	0.2283
10.5	0.2368	0.2423	0.2382	0.2409	0.2389	0.2403
13.5	0.1773	0.1818	0.1782	0.1804	0.1788	0.1799
16.5	0.1041	0.1064	0.1045	0.1057	0.1048	0.1054
19.5	0.0506	0.0514	0.0508	0.0512	0.0508	0.0510
22.5	0.0213	0.0215	0.0214	0.0215	0.0212	0.0213
25.5	0.0080	0.0080	0.0079	0.0079	0.0079	0.0079
28.5	0.0028	0.0027	0.0026	0.0026	0.0026	0.0026

Table 10: Simulation results for Brier score by applying cross-validation for shape=2

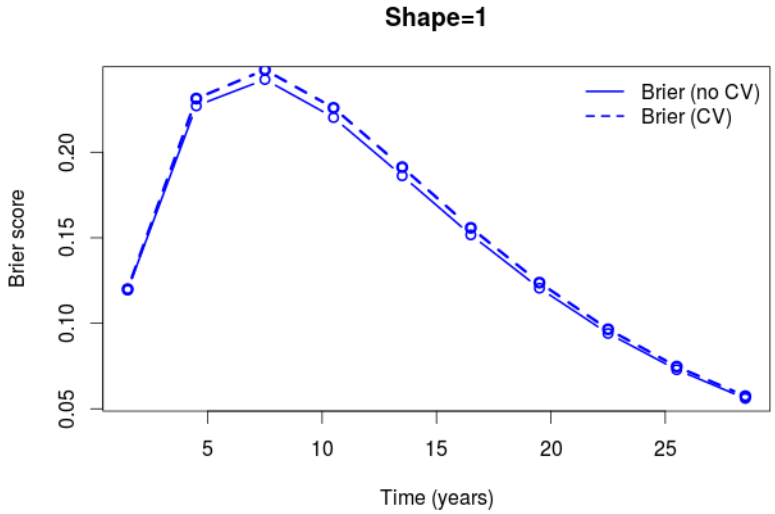


Figure 6.1: Plot of Brier scores with/without CV for n=250, by=5, shape=1

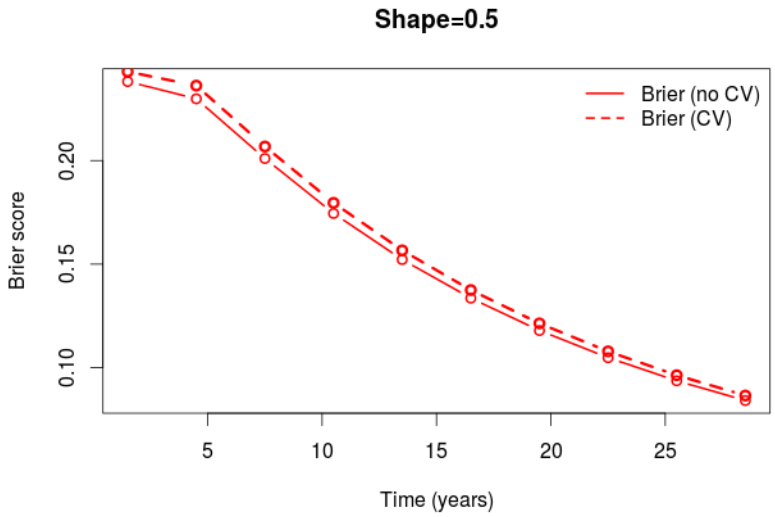


Figure 6.2: Plot of Brier scores with/without CV for n=250, by=5, shape=0.5

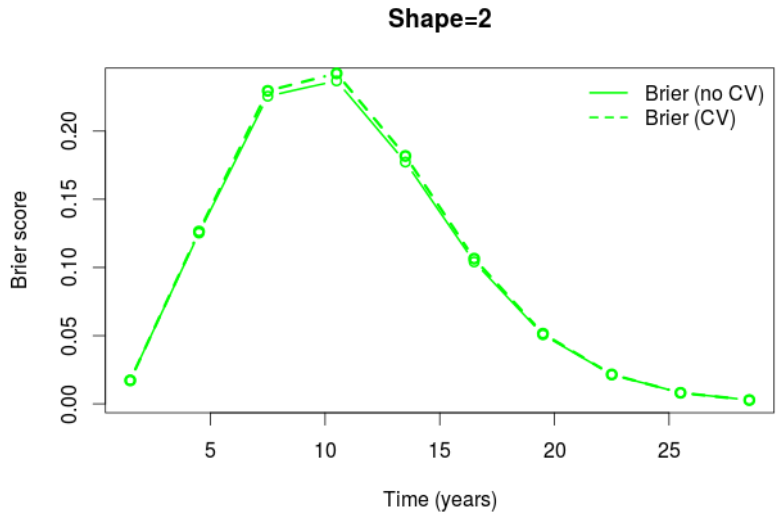


Figure 6.3: Plot of Brier scores with/without CV for n=250, by=5, shape=2

7 Application to the data

The `data(cav)`, introduced in Chapter 1, will be used to evaluate both measures described in this thesis. Assuming that $T \sim W(\lambda, \gamma)$, a model is fitted with all the covariates presented in the Introduction. The results are shown in Table 11.

	β	se	HR	LB(HR)	UB(HR)
Intercept	-2.8441	0.0134			
dage	0.0092	0.0055	1.0093	0.9983	1.0204
sex	0.0462	0.2062	1.0473	0.6990	1.5692
pdiag IHD	0.2718	0.1354	1.3123	1.0062	1.7115
pdiag Other	0.2715	0.3089	1.3120	0.7160	2.4039
where $\hat{\gamma}$ = 0.9011 and se($\hat{\gamma}$)= 0.0563					

Table 11: Output of the model when $T \sim W(\lambda, \gamma)$

The prognostic scores are fitted after estimating the β's through the formula $z_i = \hat{\beta}^T X_i$ taking into account that $T \sim W(\lambda, \gamma)$. (see Figure 7.1)

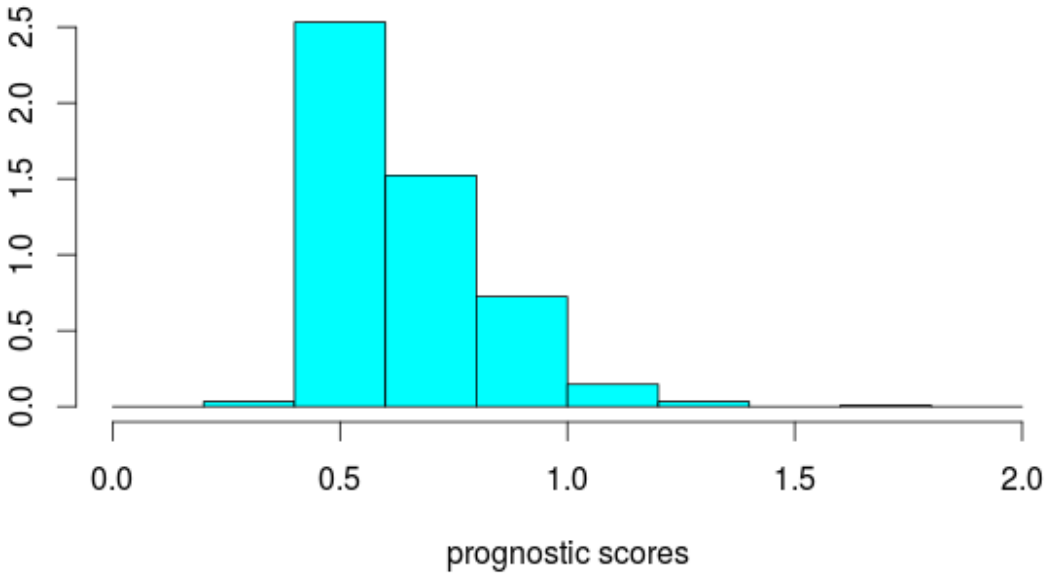


Figure 7.1: Prognostic scores $z_i = \beta^T X_i$ when $T \sim W(\lambda, \gamma)$

The results of c-index are given when our methods are applied to the data, assuming that T follows Weibull and exponential distributions. In addition, c-

index is calculated, assuming that $T \sim W(\lambda, \gamma)$ and cross-validation is applied as described in Chapter 6. The results are shown in Table 12.

	$T \sim \exp(\lambda)$
c exact	0.5651
c sim	0.5649
	$T \sim W(\lambda, \gamma)$
c sim	0.5607
Cross validation results	$T \sim W(\lambda, \gamma)$
c sim	0.5406

Table 12: C-index results for `data(cav)`

The results of Brier score are given when our methods are applied to the data, when cross-validation is used (Brier CV) and when it is not (Brier) and T is assumed to follow a Weibull distribution (see Table 13).

Time	**Brier**	**Brier CV**
1.5	0.0817	0.0826
4.5	0.1747	0.1768
7.5	0.2331	0.2358
10.5	0.2449	0.2476
13.5	0.2292	0.2313
16.5	0.2067	0.2082
19.5	0.1843	0.1851
22.5	0.1626	0.1629
25.5	0.1425	0.1423
28.5	0.1242	0.1238

Table 13: Brier score results for `data(cav)`

From the results in Table 13, the predictive accuracy derived from cross-validation is very close to the one without, but usually somewhat higher. This is expected since the same data are used to validate the model and the results without cross-validation give an over-optimistic view of the performance of the prediction model on new data. To visualize these difference, Figure 7.2 shows estimates of the Brier score calculated at a denser set of time points, along

with a curve of the Brier score for the model without any covariates, so based on fitting a Weibull model with shape γ and the same rate λ for all patients. It can be seen that the model without covariates has the highest prediction error and that including the covariates reduces the prediction error. The assessment without cross-validation, however, tends to give an over-optimistic idea of how much the prediction model including the covariates improves the prediction.

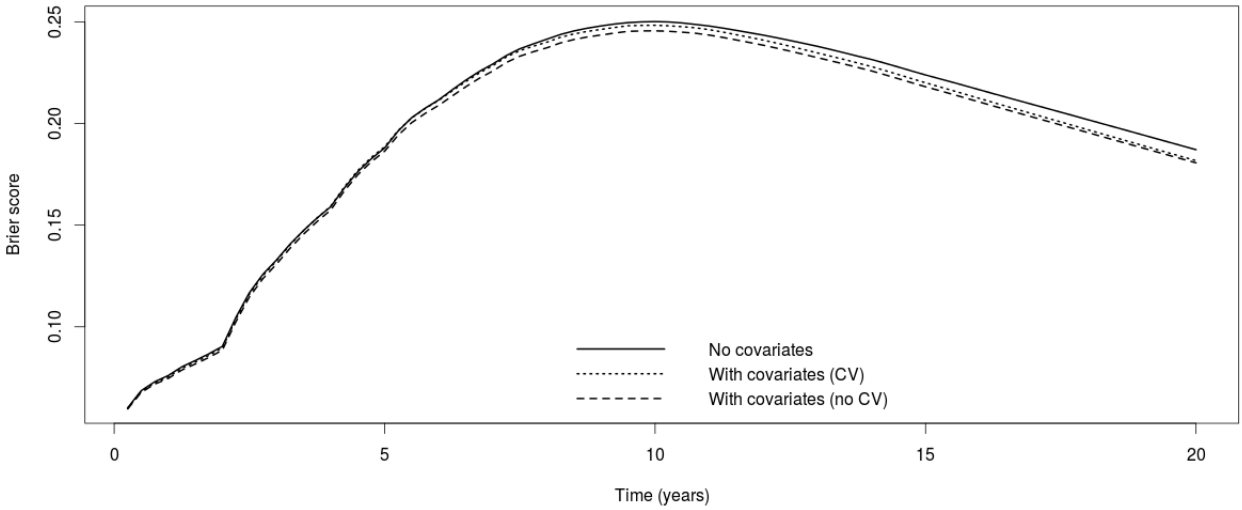


Figure 7.2: Brier scores in three different cases

8 Discussion

In this thesis, two different issues were investigated. The first one was to employ statistical probabilistic methodology to investigate c-index formulas concerning interval censored survival data. Censoring mechanism and conditional probabilities were used to address the first problem. Statistical issues in the specification of both exponential and Weibull distributions were considered. Challenging issues that may be involved in interval censored data were described: possibility that the event of interest and the exact time of the observation lie between the interval $(L_i, R_i]$ and we do not know if $P(T_i > t_0 | T_i \in (L_i, R_i])$ is zero or one.
The second goal of this thesis was to test these methods to our data. To overcome the danger of using the same data to validate the model and produce over-optimistic results of discrimination and calibration, leave-one-out cross-validation was used.

In the first part of the thesis, a new method for calculating the c-index to discriminate subjects was suggested for interval censored data. The two main distributions, exponential and Weibull, were considered. In the case of exponential explicit formulas were calculated while for Weibull a more general idea for calculating the $P(T_2 < T_1 | T_1 \in (L_1, R_1], T_2 \in (L_2, R_2])$ for each pair (i, j) is given. In addition, Brier score was also studied for interval censored data. The basic idea is when the indicator $(\mathbb{1}\{\mathbb{T}_{\mathbb{i}} > \mathbb{t}\} | T_i \in (L_i, R_i])$ is unknown, to estimate the $P(T_i > t_0 | T_i \in (L_i, R_i])$.

In the second part of the thesis, cross-validation was applied for the simulation study and for the `data(cav)`. In both cases, cross-validation revealed that the c-index was lower and the Brier scores were higher, concluding that fitting a model given the covariates is better than using cross-validation. Though, when the differences are not very large, more realistic results are produced through cross-validation.

Simulations suggest that the proposed methods provide estimates for the c-index and Brier score with relatively small bias and root-mean-squared error (RMSE). While applying these methods to `data(cav)`, the difference between the usual and the cross-validated c-index and Brier score is smaller.

For future research it may also be of interest to extend the proposed methods to other distributions (e.g. gamma, log-normal, log-logistic). In particular, for an

individual i, c-index could be calculated for any distribution given the density function ($f_{T_i}(t_i)$), the survival function at the left interval ($S_i(L_i)$) and at the right interval ($S_i(R_i)$), for $t_i \in (L_i, R_i]$. To define the Brier score in other distributions, $\hat{S}(t|X_i)$ should be settled assuming that T_i follows the desired distribution.

References

M. Gonen and G. Heller. Concordance probability and discriminatory power in proportional hazards regression. *Biometrika*, 92:965–970, 2005.

F.E. Harrell, R.M. Califf, D.B. Pryor, K.L. Lee, and R.A. Rosati. Evaluating the yield of medical tests. *The journal of the American Medical Association*, 247:2543–2546, 1982.

Frank E. Harrell, Kerry L. Lee, and Daniel B. Mark. Multivariate prognostic models: Issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors. *Statistics in Medicine*, 15:361–387, 1996.

Patrick J. Heagerty and Yingye Zheng. Survival model predictive accuracy and roc curves. *Biometrics*, 61:92–105, 2005.

X. Liu, Z. Jin, and J.H. Graziano. Comparing paired biomarkers in predicting quantitative health outcome subject to random censoring. *Statistical methods in medical research*, 2012.

Ian R. White and Eleni Rapsomaniki. Covariate-adjusted measures of discrimination for survival data. *Biometrical Journal*, 57:592–613, 2015.

A Appendix: R code

```
########## load libraries

library(msm)
library(survival)
library(KMsurv)
library(SurvRegCensCov)

source("Function definitions.R")

############ Start preliminary

head(cav)
names(cav) <- casefold(names(cav))
pats <- sort(unique(cav$ptnum))
n <- length(pats)
head(cav, n=18)
table(diff(which(cav$firstobs==1))) # no pats with only 1 line
cav1 <- subset(cav, firstobs==1)
whlast <- c(which(cav$firstobs==1)-1, nrow(cav))
cavlast <- cav[whlast, ]
cavminlast <- cav[-whlast, ]
whlast <- c(which(cavminlast$firstobs==1)-1, nrow(cavminlast))
cavlastbutone <- cavminlast[whlast, ]
dim(cav1)
dim(cavlast)
dim(cavlastbutone)

# Going to add two columns, la (lastalive) and fd (firstdead), containing
# time at which subject was last known to be alive and first to be dead;
# if right censored, fd will be infinity
table(cavlast$state)
# Those with 4 have died, rest are right censored
# Right censored
cav1$la <- 0
cav1$la[cavlast$state<4] <- cavlast$years[cavlast$state<4]
cav1$fd <- Inf
# Those who died, la comes from cavlastbutone, fd from cavlast
cav1$la[cavlast$state==4] <- cavlastbutone$years[cavlast$state==4]
cav1$fd[cavlast$state==4] <- cavlast$years[cavlast$state==4]
cav1 <- cav1[, c("ptnum", "la", "fd", "dage", "sex", "pdiag")]
# Check whether it is correct
head(cav, n=21)
tail(cav, n=21)
head(cav1)
tail(cav1, n=9)

# Remove the missing values (8 NA's in pdiag, 622-8=614 observations)

cav1<- cav1[-(which(is.na(cav1$pdiag))),]

# This should be done once and the result should be saved in

cav1$status <- as.numeric(is.finite(cav1$fd)) + 2
cav1$status <- ifelse(is.infinite(cav1$fd), 2, 3)
## If there are zeros in the data R returns the error: "Error in survreg:
## Invalid survival times for this distribution". A trick to get around this problem
```

```
## when you have intervals that begin at time zero is to add a small constant to all
## survival times in the dataset.

cav1$la[cav1$la == 0] <- 1e-04
whc <- complete.cases(cbind(cav1$dage, cav1$sex, cav1$pdiag))
cav1 <- cav1[whc, ]

# Change pdiag to factor with 3 levels, take IDC (reference), IHD and
# other

table(cav1$pdiag)
cav1$pdiag <- as.numeric(cav1$pdiag) - 2
cav1$pdiag[cav1$pdiag <= 0] <- 3  # to other
cav1$pdiag[cav1$pdiag >= 3] <- 3  # to other
cav1$pdiag <- factor(cav1$pdiag, levels = 1:3, labels = c("IDC", "IHD",
              "Other"))
table(cav1$pdiag)

cav1<- data.frame(cav1,status)
View(cav1)

save(cav1, file="cav.Rdata")


############ End preliminary

load("cav.Rdata")

######## Chapter 1

## Plot the 25 first individuals interval censored
select <- head(order(cav1$fd - cav1$la, decreasing = TRUE), 25)
data <- cav1[select, ]
lablist <- as.vector(c(1:20))
lablist2 <- as.vector(c(1:20))

plot(lablist, lablist2, type = "n", axes = FALSE, ann = FALSE, xlab = NA, ylab = NA)
axis(1, at = 1:20, side = 1, labels = c(0:19))
axis(2, at = c(1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21),
     labels = c(1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21), las = 1)
mtext("Infinity", side = 1, line = -4, outer = TRUE, adj = 1)
mtext("Range of intervals", side = 1, line = 4)
mtext("Individuals", side = 2, line = 3)
box()

arrows(20.5, -0.5, 21.2, -0.5, length = 0.1, xpd = TRUE)

arrows(cav1$la[10], 1, cav1$fd[10], 1, length = 0)
arrows(cav1$la[42], 2, cav1$fd[42], 2, length = 0)
arrows(data$la[6], 3, 51, 3, length = 0)
arrows(cav1$la[6], 4, cav1$fd[6], 4, length = 0)
arrows(data$la[9], 5, 51, 5, length = 0)
arrows(data$la[7], 6, 51, 6, length = 0)
arrows(cav1$la[58], 7, cav1$fd[58], 7, length = 0)
arrows(cav1$la[18], 8, cav1$fd[18], 8, length = 0)
arrows(cav1$la[59], 9, cav1$fd[59], 9, length = 0)
arrows(cav1$la[52], 10, cav1$fd[52], 10, length = 0)
arrows(data$la[8], 11, 51, 11, length = 0)
```

```
arrows(cav1$la[40], 12, cav1$fd[40], 12, length = 0)
arrows(cav1$la[49], 13, cav1$fd[49], 13, length = 0)
arrows(data$la[5], 14, 51, 14, length = 0)
arrows(cav1$la[44], 15, cav1$fd[44], 15, length = 0)
arrows(data$la[4], 16, 51, 16, length = 0)
arrows(data$la[1], 17, 51, 17, length = 0)
arrows(data$la[10], 18, 51, 18, length = 0)
arrows(data$la[3], 19, 51, 19, length = 0)
arrows(data$la[2], 20, 51, 20, length = 0)

text(x = cav1$la[10], y = 1, "(", cex = 1)
text(x = cav1$la[42], y = 2, "(", cex = 1)
text(x = data$la[6], y = 3, "(", cex = 1)
text(x = cav1$la[6], y = 4, "(", cex = 1)
text(x = data$la[9], y = 5, "(", cex = 1)
text(x = data$la[7], y = 6, "(", cex = 1)
text(x = cav1$la[58], y = 7, "(", cex = 1)
text(x = cav1$la[18], y = 8, "(", cex = 1)
text(x = cav1$la[59], y = 9, "(", cex = 1)
text(x = cav1$la[52], y = 10, "(", cex = 1)
text(x = data$la[8], y = 11, "(", cex = 1)
text(x = cav1$la[40], y = 12, "(", cex = 1)
text(x = cav1$la[49], y = 13, "(", cex = 1)
text(x = data$la[5], y = 14, "(", cex = 1)
text(x = cav1$la[44], y = 15, "(", cex = 1)
text(x = data$la[4], y = 16, "(", cex = 1)
text(x = data$la[1], y = 17, "(", cex = 1)
text(x = data$la[10], y = 18, "(", cex = 1)
text(x = data$la[3], y = 19, "(", cex = 1)
text(x = data$la[2], y = 20, "(", cex = 1)

text(x = cav1$fd[10], y = 1, "]", cex = 1)
text(x = cav1$fd[42], y = 2, "]", cex = 1)
text(x = cav1$fd[6], y = 4, "]", cex = 1)
text(x = cav1$fd[58], y = 7, "]", cex = 1)
text(x = cav1$fd[18], y = 8, "]", cex = 1)
text(x = cav1$fd[59], y = 9, "]", cex = 1)
text(x = cav1$fd[52], y = 10, "]", cex = 1)
text(x = cav1$fd[40], y = 12, "]", cex = 1)
text(x = cav1$fd[49], y = 13, "]", cex = 1)
text(x = cav1$fd[44], y = 15, "]", cex = 1)


# Age histogram

hist.age <- hist(cav1$dage, breaks = 10, col = "red", main = "", xlab = "Age in years",
                 xlim = c(0, 70), ylim = c(0, 130))
hist.age


######### Chapter 2,3

### EXPONENTIAL CASE - CREATE FUNCTIONS FOR EACH OF THE 6 CASES, C-INDEX
### CASE1(L1<L2<R1<R2), P(T1<T2), P(T1>T2)

f <- function(hazard1, hazard2, left1, left2, right1, right2)
{
```

```
  a <- -hazard1/(hazard1 + hazard2)
  b <- exp(-hazard1 * right1)
  c <- exp(-hazard2 * left2) - exp(-hazard2 * right1)
  d <- hazard2/(hazard1 + hazard2)
  e <- exp(-hazard2 * left2)
  z <- exp(-hazard1 * left2) - exp(-hazard1 * right1)

  out1 <- a * b * c + d * e * z
  denominator1 <- exp(-hazard1 * left1) - exp(-hazard1 * right1)
  denominator2 <- exp(-hazard2 * left2) - exp(-hazard2 * right2)

  joint <- denominator1 * denominator2

  out <- out1/joint
  out[out1 == 0 & joint == 0] <- 0 #may get NaN because of 0/0,the result should be 0

  return(out)
}

## CASE2(L1<L2<R2<R1), P(T1<T2), P(T1>T2)

h <- function(hazard1, hazard2, left1, left2, right1, right2)
{

  a <- exp(-hazard1 * left1)
  b <- exp(-hazard2 * left2) - exp(-hazard2 * right2)
  c <- hazard2/(hazard1 + hazard2)
  d <- exp(-(hazard1 + hazard2) * left2) - exp(-(hazard1 + hazard2) * right2)

  out1 <- a * b - c * d

  denominator1 <- exp(-hazard1 * left1) - exp(-hazard1 * right1)
  denominator2 <- exp(-hazard2 * left2) - exp(-hazard2 * right2)

  joint <- denominator1 * denominator2

  out <- 1 - (out1/joint)
  out[out1 == 0 & joint == 0] <- 0

  return(out)
}

## CASE4(L2<L1<R1<R2), P(T1<T2), P(T1>T2)

g <- function(hazard1, hazard2, left1, left2, right1, right2)
{

  a <- exp(-hazard2 * left2)
  b <- exp(-hazard1 * left1) - exp(-hazard1 * right1)
  c <- hazard1/(hazard1 + hazard2)
  d <- exp(-(hazard1 + hazard2) * left1) - exp(-(hazard1 + hazard2) * right1)

  out1 <- a * b - c * d
  denominator1 <- exp(-hazard1 * left1) - exp(-hazard1 * right1)
  denominator2 <- exp(-hazard2 * left2) - exp(-hazard2 * right2)

  joint <- denominator1 * denominator2
```

```
  out <- out1/joint
  out[out1 == 0 & joint == 0] <- 0

  return(out)
}

## CASE5(L2<L1<R2<R1), P(T1<T2), P(T1>T2)


k <- function(hazard1, hazard2, left1, left2, right1, right2)
{

  a <- -hazard2/(hazard1 + hazard2)
  b <- exp(-hazard2 * right2)
  c <- exp(-hazard1 * left1) - exp(-hazard1 * right2)
  d <- hazard1/(hazard1 + hazard2)
  e <- exp(-hazard1 * left1)
  z <- exp(-hazard2 * left1) - exp(-hazard2 * right2)

  out1 <- a * b * c + d * e * z
  denominator1 <- exp(-hazard1 * left1) - exp(-hazard1 * right1)
  denominator2 <- exp(-hazard2 * left2) - exp(-hazard2 * right2)

  joint <- denominator1 * denominator2

  out <- 1 - (out1/joint)
  out[out1 == 0 & joint == 0] <- 0

  return(out)
}

## C-INDEX IN THE EXPONENTIAL CASE

c.exact <- function(hazard, left, right)
{
  n <- length(hazard)

  ### exponential case
  allpairs <- t(combn(hazard, 2))
  indl <- t(combn(left, 2))
  indr <- t(combn(right, 2))
  data.new <- data.frame(allpairs, indl, indr)
  h11 <- data.new[, 1]
  h22 <- data.new[, 2]
  indl1 <- data.new[, 3]
  indl2 <- data.new[, 4]
  indr1 <- data.new[, 5]
  indr2 <- data.new[, 6]
  data.new <- data.frame(h11, h22, indl1, indl2, indr1, indr2)

  case1 <- data.new[which(indl1 <= indl2 & indl2 <= indr1 & indr1 <= indr2), ]
  case2 <- data.new[which(indl1 <= indl2 & indl2 <= indr2 & indr2 < indr1), ]
  case3 <- data.new[which(indl1 < indr1 & indr1 <= indl2 & indl2 < indr2), ]
  case4 <- data.new[which(indl2 < indl1 & indl1 <= indr1 & indr1 <= indr2), ]
  case5 <- data.new[which(indl2 < indl1 & indl1 <= indr2 & indr2 < indr1), ]
  case6 <- data.new[which(indl2 < indr2 & indr2 <= indl1 & indl1 < indr1), ]

  Pt21 <- f(case1$h11, case1$h22, case1$indl1, case1$indl2, case1$indr1, case1$indr2)
```

```
  Pt11 <- 1 - Pt21
  Pt22 <- h(case2$h11, case2$h22, case2$indl1, case2$indl2, case2$indr1, case2$indr2)
  Pt12 <- 1 - Pt22
  Pt23 <- numeric(nrow(case3))
  Pt13 <- rep(1, nrow(case3)) - Pt23
  Pt24 <- g(case4$h11, case4$h22, case4$indl1, case4$indl2, case4$indr1, case4$indr2)
  Pt14 <- 1 - Pt24
  Pt25 <- k(case5$h11, case5$h22, case5$indl1, case5$indl2, case5$indr1, case5$indr2)
  Pt15 <- 1 - Pt25
  Pt26 <- rep(1, (nrow(case6)))
  Pt16 <- rep(1, (nrow(case6))) - Pt26
  Pt2 <- c(Pt21, Pt22, Pt23, Pt24, Pt25, Pt26)
  Pt1 <- c(Pt11, Pt12, Pt13, Pt14, Pt15, Pt16)

  final <- rbind(case1[, 1:6], case2[, 1:6], case3[, 1:6], case4[, 1:6], case5[, 1:6],
            case6[,1:6])
  final <- data.frame(final, Pt2, Pt1)
  final1 <- final[final$h11 > final$h22, ]
  final2 <- final[final$h22 > final$h11, ]

  c.index <- (sum(final1$Pt1) + sum(final2$Pt2))/(nrow(final1) + nrow(final2))
  return(c.index)
}

generate <- function(lam, l, r)
{
  n <- length(lam)
  u <- runif(n)
  draws <- -log(exp(-lam * l) - u * (exp(-lam * l) - exp(-lam * r)))/lam
  return(draws)
}


generate2 <- function(lam, l, r, gam)
{
  n <- length(lam)
  u <- runif(n)
  draws <- (-log(exp(-lam * l^gam)-u *(exp(-lam * l^gam) -
            exp(-lam * r^gam)))/lam)^(1/gam)
  return(draws)
}


## GENERAL CASE
c.general <- function(lambda,left,right,M,shape,dist = c("exponential", "weibull"))
{
  dist <- match.arg(dist)

  n <- length(lambda)
  idx11 <- t(combn(1:n, 2))
  lambda11 <- t(combn(lambda, 2))
  left11 <- t(combn(left, 2))
  right11 <- t(combn(right, 2))
  all <- data.frame(lambda11, left11, right11)

  lambda1 <- all[, 1]
  lambda2 <- all[, 2]
  left1 <- all[, 3]
```

```
  left2 <- all[, 4]
  right1 <- all[, 5]
  right2 <- all[, 6]

  Pt1.simulate <- rep(NA, nrow(all))
  Pt1.simulate[right1 <= left2] <- 1
  Pt1.simulate[right2 <= left1] <- 0
  ## test which overlap
  overlap <- which(is.na(Pt1.simulate))

  ## probability for those that overlap
  n1 <- length(overlap)


  draws <- matrix(NA, n, M)
  if (dist == "exponential")
  {
    for (m in 1:M)
    {
      draws[, m] <- generate(lambda, left, right)
    }
  } else if (dist == "weibull")
  {
    for (m in 1:M)
    {
      draws[, m] <- generate2(lambda, left, right, shape)
    }
  }

  for (j in 1:n1)
  {
    subj1 <- idx11[overlap[j], 1]
    draws1 <- draws[subj1, ]
    subj2 <- idx11[overlap[j], 2]
    draws2 <- draws[subj2, ]
  # Put the result (mean number of draws for which draws1<draws2) into Pt1.simulate
    Pt1.simulate[overlap[j]] <- mean(ifelse(draws1 < draws2, 1, 0))
  }

  Pt2.simulate <- 1 - Pt1.simulate

  data.simulate <- data.frame(lambda1, lambda2, left1, left2, right1, right2,
                              Pt1.simulate, Pt2.simulate)

  final1.simulate <- data.simulate[data.simulate$lambda1 > data.simulate$lambda2, ]
  final2.simulate <- data.simulate[data.simulate$lambda2 > data.simulate$lambda1, ]

  c.index.simulate <- (sum(final1.simulate$Pt1.simulate) +
  sum(final2.simulate$Pt2.simulate))/(nrow(final1.simulate) + nrow(final2.simulate))
  return(c.index.simulate)
}


######### Chapter 4

## BRIER SCORE IN THE EXPONENTIAL CASE

prediction.error <- function(hazard, left, right, time)
```

```
{
  brier <- numeric(length(time))
  for (i in 1:length(time))
  {
    alive <- ifelse(right > time[i], 1, 0)
    uncertain1 <- which(alive == 1 & time[i] > left & is.finite(right))
    uncertain2 <- which(alive == 1 & time[i] > left & is.infinite(right))
    alive[uncertain1] <- (exp(-hazard[uncertain1] * time[i]) -
                          exp(-hazard[uncertain1] * right[uncertain1]))/
                          (exp(-hazard[uncertain1] * left[uncertain1])
                         * exp(-hazard[uncertain1] * right[uncertain1]))
    alive[uncertain2] <- (exp(-hazard[uncertain2] * time[i]))/
                          (exp(-hazard[uncertain2] * left[uncertain2]))
    survival <- exp(-hazard * time[i])
    statistic <- alive - survival
    statistic2 <- statistic^2
    brier[i] <- mean(statistic2)
  }
  return(brier)
}


## BRIER SCORE IN THE GENERAL CASE

brier.score <- function(lambda, left, right, shape, time = seq(1.5, 28.5, by = 3))
{
  brier <- numeric(length(time))
  for (i in 1:length(time))
  {
    alive <- ifelse(right > time[i], 1, 0)
    uncertain1 <- which(alive == 1 & time[i] > left & is.finite(right))
    uncertain2 <- which(alive == 1 & time[i] > left & is.infinite(right))
    alive[uncertain1] <- (exp(-lambda[uncertain1] * time[i]^shape) -
                           exp(-lambda[uncertain1] * right[uncertain1]^shape))/
                           (exp(-lambda[uncertain1] * left[uncertain1]^shape) -
                            exp(-lambda[uncertain1] * right[uncertain1]^shape))

    alive[uncertain2] <- (exp(-lambda[uncertain2] * time[i]^shape))/
                           (exp(-lambda[uncertain2] * left[uncertain2]^shape))
    survival <- exp(-lambda * time[i]^shape)

    brier[i] <- mean(alive * (1 - 2 * survival) + survival^2)
  }
  return(brier)
}


######################### Chapter 5,6


my.rweibull <- function(lam, gam)
{
  n <- length(lam)
  u <- runif(n)
  res <- ((-log(1 - u))/lam)^(1/gam)
  return(res)
}
```

```
### SIMULATE DATA

simulate <- function(n,betas,shape,dist = c("exponential","weibull"),by = 3)
{

  betas[1] <- log((gamma(1 + 1/shape)/10)^shape)

  ## Generate the covariates
  x1 <- rnorm(n, 0, 10)  ## age
  x2 <- sample(0:1, size = n, replace = TRUE, prob = c(0.5, 0.5)) - 0.5  #sex
  x3 <- sample(1:3, size = n, replace = TRUE, prob = c(0.4, 0.35, 0.25))
  x32 <- as.numeric(x3 == 2) - 0.35
  x33 <- as.numeric(x3 == 3) - 0.25

  ## Generate the hazard
  lambda.simulate <- numeric(n)

  X <- model.matrix(~x1 + x2 + x32 + x33)

  lambda.simulate <- as.numeric(exp(X %*% betas))


  ## Generate the exact event times
  if (dist == "exponential")
    exactTimes <- rexp(n, rate = lambda.simulate)
  else exactTimes <- my.rweibull(lam = lambda.simulate, gam = shape)

  ## Generate the left and right intervals
  obsIntervals <- seq(0, 30, by = by)
  time1 <- floor(exactTimes/by) * by
  time1[time1 == 0] <- 1e-08
  time1[exactTimes < 0] <- NA
  time1[exactTimes > 30] <- 30
  time2 <- time1 + by
  time2[exactTimes > 30] <- Inf

  id <- seq(n)
  return(data.frame(id = id, lambda = lambda.simulate, left = time1,
                    right = time2,x1 = x1, x2 = x2, x32 = x32, x33 = x33,
                    exactTimes = exactTimes))
}


# MONTE CARLO APPROXIMATION OF TRUE C-INDEX

true.c.index <- function(n, betas, shape)
{
  betas[1] <- log((gamma(1 + 1/shape)/10)^shape)
  dfr1 <- simulate(n = n, betas = betas, shape = shape, dist = "weibull")
  dfr1$one <- 1
  cexact <- coxph(Surv(exactTimes, one) ~ x1 + x2 + x32 + x33, data = dfr1)
  ctrue <- cexact$concordance[1]/(cexact$concordance[1] + cexact$concordance[2])
  return(ctrue)
}

# MONTE CARLO APPROXIMATION OF TRUE BRIER SCORE
```

```
true.brier <- function(n, betas, shape, time = seq(1.5, 28.5, by = 3))
{
  betas[1] <- log((gamma(1 + 1/shape)/10)^shape)
  dfr1 <- simulate(n = n, betas = betas, shape = shape, dist = "weibull")
  brier.true <- numeric(length(time))
  for (i in 1:length(time))
  {
    alive <- ifelse(dfr1$exactTimes > time[i], 1, 0)
    survival <- exp(-dfr1$lambda * time[i]^shape)
    statistic <- alive - survival
    statistic2 <- statistic^2
    brier.true[i] <- mean(statistic2)
  }
  return(brier.true)
}

#### SIMULATION RESULTS

sim.results <- function(n,nsim,betas,shape,by,time = seq(1.5, 28.5, by = 3))
{

  res.sim <- rep(NA, nsim)
  res.exact <- rep(NA, nsim)
  brier <- matrix(NA, nsim, length(time))
  res.simCV <- rep(NA, nsim)
  res.exactCV <- rep(NA, nsim)
  brierCV <- matrix(NA, nsim, length(time))

  # H Recenter betas[1] so that mean survival at mean of covariates = 10
  betas[1] <- log((gamma(1 + 1/shape)/10)^shape)
  for (j in 1:nsim) {
    # Simulate data
    dfr1 <- simulate(n = n, betas = betas, shape = shape, dist = "weibull")
    # H changed again !!!!!
    dfr1$right.na <- dfr1$right
    dfr1$right.na[is.infinite(dfr1$right.na)] <- NA
    # Estimate betas from the data
    model <- survreg(Surv(time = left, time2 = right.na,
                          type = "interval2") ~ x1 + x2 + x32 + x33, dist = "weibull",
                     data = dfr1)

    # reparametrization
    sigma <- model$scale
    mu <- summary(model)$coef[1]
    gamma <- 1/sigma
    lambda <- -mu/sigma
    p <- length(summary(model)$coef) - 1
    alpha <- summary(model)$coef[2:(p + 1)]
    betasj <- c(lambda, -alpha/sigma)
    # cbind(c(shape,betas), c(gamma,betasj))

    # Calculate hazard for each subject in the data and add it to the data
    X <- model.matrix(model)
    lambdastar <- exp(X %*% betasj)
    dfr1$lambda <- lambdastar

    # Same using cross-validation
```

```
    lambdamini <- gammamini <- numeric(n)
    for (i in 1:n) {
      dfrmini <- dfr1[-i, ]
      model <- survreg(Surv(time = left, time2 = right.na, type = "interval2") ~
                       x1 + x2 + x32 + x33, dist = "weibull", data = dfrmini)
      alpha <- summary(model)$coefficients
      sigma <- model$scale
      mu <- alpha[1]
      gammamini[i] <- 1/sigma
      beta0 <- -mu/sigma
      betamini <- c(beta0, -alpha[2:(p + 1)]/sigma)

      # Calculate cross-validated hazard for subject i
      # Use original model matrix
      lambdamini[i] <- exp(X[i, ] %*% betamini)
    }

    if (shape == 1) {
      res.sim[j] <- c.general(dfr1$lambda, dfr1$left, dfr1$right,
                              shape = shape, M = 100, dist = "exponential")
    } else {
      res.sim[j] <- c.general(dfr1$lambda, dfr1$left, dfr1$right,
                              shape = gamma, M = 100, dist = "weibull")
    }
    if (is.na(res.sim[j]))
      break
    res.exact[j] <- c.exact(dfr1$lambda, dfr1$left, dfr1$right)
    brier[j, ] <- brier.score(lambda = dfr1$lambda, left = dfr1$left,
                              right = dfr1$right, shape = gamma, time = time)

    # H Same thing for cross-validated versions
    if (shape == 1) {
      res.simCV[j] <- c.general(lambdamini, dfr1$left, dfr1$right,
                                shape = shape, M = 100, dist = "exponential")
    } else {
      res.simCV[j] <- c.general(lambdamini, dfr1$left, dfr1$right,
                                shape = gammamini, M = 100, dist = "weibull")
    }
    if (is.na(res.sim[j]))
      break
    res.exactCV[j] <- c.exact(lambdamini, dfr1$left, dfr1$right)
    brierCV[j, ] <- brier.score(lambda = lambdamini, left = dfr1$left,
                                right = dfr1$right, shape = gammamini, time = time)

  }

  return(list(csim = res.sim, cexact = res.exact, brier = brier,
              csimCV = res.simCV, cexactCV = res.exactCV, brierCV = brierCV))

}

summarize <- function(res, truec, truebrier) {
  meansim <- mean(res$csim)
  sdsim <- sd(res$csim)
  meanexact <- mean(res$cexact)
  sdexact <- sd(res$cexact)
  meanbrier <- colMeans(res$brier)
  sdbrier <- apply(res$brier, 2, sd)
```

```
  biasexact <- meanexact - truec
  RMSEexact <- sqrt(biasexact^2 + sdexact^2)
  biassim <- meansim - truec
  RMSEsim <- sqrt(biassim^2 + sdsim^2)
  biasbrier <- meanbrier - truebrier
  RMSEbrier <- sqrt(biasbrier^2 + sdbrier^2)
  dfrBrier <- data.frame(bias = biasbrier, RMSE = RMSEbrier)
  rownames(dfrBrier) <- seq(1.5, 28.5, by = 3)

  # Cross-validated results
  meansimCV <- mean(res$csimCV)
  meanexactCV <- mean(res$cexactCV)
  meanbrierCV <- colMeans(res$brierCV)

  return(list(cindex = data.frame(meanexact = meanexact, biasexact = biasexact,
                        RMSEexact = RMSEexact, meansim = meansim,
                        biassim = biassim, RMSEsim = RMSEsim), Brier = dfrBrier,
  cindexCV = list(exact = meanexactCV, sim = meansimCV), BrierCV = meanbrierCV))

  # biassim=biassim, RMSEsim=RMSEsim, biasexact=biasexact,
  # RMSEexact=RMSEexact, biasbrier=biasbrier, RMSEbrier=RMSEbrier))
}



######################### SIMULATION STUDY
ctrue1 <-true.c.index(n = 1e+06, betas = c(-3, 0.01, 0.05, 0.30, 0.25), shape = 1)
ctrue05 <-true.c.index(n =1e+06, betas = c(-3, 0.01, 0.05, 0.30, 0.25), shape = 0.5)
ctrue2 <-true.c.index(n = 1e+06, betas = c(-3, 0.01, 0.05, 0.30, 0.25), shape = 2)
truecs <-list(ctrue1 = ctrue1, ctrue05 = ctrue05, ctrue2 = ctrue2)
save(truecs, file = "truecs.Rdata")

load("truecs.Rdata")
ctrue1 <- truecs$ctrue1
ctrue05 <- truecs$ctrue05
ctrue2 <- truecs$ctrue2

true.brier1 <- true.brier(n = 1e+06, betas = c(-3, 0.01, 0.05, 0.30, 0.25), shape = 1)
true.brier05 <-true.brier(n = 1e+06, betas = c(-3, 0.01, 0.05, 0.30, 0.25), shape = 0.5)
true.brier2 <- true.brier(n = 1e+06, betas = c(-3, 0.01, 0.05, 0.30, 0.25), shape = 2)
true.briers <- list(true.brier1 = true.brier1, true.brier05 = true.brier05,
                    true.brier2 = true.brier2)
save(true.briers, file = "true.briers.Rdata")

load("true.briers.Rdata")
true.brier1 <- true.briers$true.brier1
true.brier05 <- true.briers$true.brier05
true.brier2 <- true.briers$true.brier2

## Actual simulations

# shape=1,by=1,n=250
date()
res <- sim.results(n = 250, nsim = 10, betas = c(-3, 0.01, 0.05, 0.30,
                   0.25), shape = 1, by = 1, time = seq(1.5, 28.5, by = 3))
date()
load("resn250shape1by1M1000.Rdata")
n250shape1by1 <- summarize(res, truec = ctrue1, truebrier = true.brier1)
```

```
n250shape1by1


# shape=1,by=1,n=500
date()
res <- sim.results(n = 500, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 1, by = 1, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn500shape1by1M1000.Rdata")
load("resn500shape1by1M1000.Rdata")
n500shape1by1 <- summarize(res, truec = ctrue1, truebrier = true.brier1)
n500shape1by1

# shape=1,by=1,n=1000
date()
res <- sim.results(n = 1000, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 1, by = 1, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn1000shape1by1M1000.Rdata")
load("resn1000shape1by1M1000.Rdata")
n1000shape1by1 <- summarize(res, truec = ctrue1, truebrier = true.brier1)
n1000shape1by1

# shape=1,by=3,n=250
date()
res <- sim.results(n = 250, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                   0.25), shape = 1, by = 3, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn250shape1by3M1000.Rdata")
n250shape1by3 <- summarize(res, truec = ctrue1, truebrier = true.brier1)
n250shape1by3

# shape=1,by=3,n=500
date()
res <- sim.results(n = 500, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                   0.25), shape = 1, by = 3, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn500shape1by3M1000.Rdata")
n500shape1by3 <- summarize(res, truec = ctrue1, truebrier = true.brier1)
n500shape1by3

# shape=1,by=3,n=1000
date()
res <- sim.results(n = 1000, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                   0.25), shape = 1, by = 3, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn1000shape1by3M1000.Rdata")
n1000shape1by3 <- summarize(res, truec = ctrue1, truebrier = true.brier1)
n1000shape1by3

# shape=1,by=5,n=250
date()
res <- sim.results(n = 250, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 1, by = 5, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn250shape1by5M1000.Rdata")
n250shape1by5 <- summarize(res, truec = ctrue1, truebrier = true.brier1)
n250shape1by5
```

```
# shape=1,by=5,n=500
date()
res <- sim.results(n = 500, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 1, by = 5, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn500shape1by5M1000.Rdata")
n500shape1by5 <- summarize(res, truec = ctrue1, truebrier = true.brier1)
n500shape1by5

# shape=1,by=5,n=1000
date()
res <- sim.results(n = 1000, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 1, by = 5, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn1000shape1by5M1000.Rdata")
n1000shape1by5 <- summarize(res, truec = ctrue1, truebrier = true.brier1)
n1000shape1by5


#####################################################################

# shape=0.5,by=1,n=250
date()
res <- sim.results(n = 250, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 0.5, by = 1, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn250shape05by1M1000.Rdata")
n250shape05by1 <- summarize(res, truec = ctrue05, truebrier = true.brier05)
n250shape05by1

# shape=0.5,by=1,n=500
date()
res <- sim.results(n = 500, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 0.5, by = 1, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn500shape05by1M1000.Rdata")
n500shape05by1 <- summarize(res, truec = ctrue05, truebrier = true.brier05)
n500shape05by1

# shape=0.5,by=1,n=1000
date()
res <- sim.results(n = 1000, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 0.5, by = 1, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn1000shape05by1M1000.Rdata")
n1000shape05by1 <- summarize(res, truec = ctrue05, truebrier = true.brier05)
n1000shape05by1

# shape=0.5,by=3,n=250
date()
res <- sim.results(n = 250, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 0.5, by = 3, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn250shape05by3M1000.Rdata")
n250shape05by3 <- summarize(res, truec = ctrue05, truebrier = true.brier05)
n250shape05by3
```

```
# shape=0.5,by=3,n=500
date()
res <- sim.results(n = 500, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 0.5, by = 3, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn500shape05by3M1000.Rdata")
n500shape05by3 <- summarize(res, truec = ctrue05, truebrier = true.brier05)
n500shape05by3

# shape=0.5,by=3,n=1000
date()
res <- sim.results(n = 1000, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 0.5, by = 3, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn1000shape05by3M1000.Rdata")
n1000shape05by3 <- summarize(res, truec = ctrue05, truebrier = true.brier05)
n1000shape05by3

# shape=0.5,by=5,n=250
date()
res <- sim.results(n = 250, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 0.5, by = 5, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn250shape05by5M1000.Rdata")
n250shape05by5 <- summarize(res, truec = ctrue05, truebrier = true.brier05)
n250shape05by5

# shape=0.5,by=5,n=500
date()
res <- sim.results(n = 500, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 0.5, by = 5, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn500shape05by5M1000.Rdata")
n500shape05by5 <- summarize(res, truec = ctrue05, truebrier = true.brier05)
n500shape05by5

# shape=0.5,by=5,n=1000
date()
res <- sim.results(n = 1000, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 0.5, by = 5, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn1000shape05by5M1000.Rdata")
n1000shape05by5 <- summarize(res, truec = ctrue05, truebrier = true.brier05)
n1000shape05by5


######################################################################


# shape=2,by=1,n=250
date()
res <- sim.results(n = 250, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                   0.25), shape = 2, by = 1, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn250shape2by1M1000.Rdata")
n250shape2by1 <- summarize(res, truec = ctrue2, truebrier = true.brier2)
n250shape2by1
```

```
# shape=2,by=1,n=500
date()
res <- sim.results(n = 500, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 2, by = 1, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn500shape2by1M1000.Rdata")
n500shape2by1 <- summarize(res, truec = ctrue2, truebrier = true.brier2)
n500shape2by1

# shape=2,by=1,n=1000
date()
res <- sim.results(n = 1000, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                   0.25), shape = 2, by = 1, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn1000shape2by1M1000.Rdata")
n1000shape2by1 <- summarize(res, truec = ctrue2, truebrier = true.brier2)
n1000shape2by1

# shape=2,by=3,n=250
date()
res <- sim.results(n = 250, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 2, by = 3, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn250shape2by3M1000.Rdata")
n250shape2by3 <- summarize(res, truec = ctrue2, truebrier = true.brier2)
n250shape2by3

# shape=2,by=3,n=500
date()
res <- sim.results(n = 500, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 2, by = 3, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn500shape2by3M1000.Rdata")
n500shape2by3 <- summarize(res, truec = ctrue05, truebrier = true.brier05)
n500shape2by3

# shape=2,by=3,n=1000
date()
res <- sim.results(n = 1000, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 2, by = 3, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn1000shape2by3M1000.Rdata")
n1000shape2by3 <- summarize(res, truec = ctrue2, truebrier = true.brier2)
n1000shape2by3

# shape=2,by=5,n=250
date()
res <- sim.results(n = 250, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 2, by = 5, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn250shape2by5M1000.Rdata")
n250shape2by5 <- summarize(res, truec = ctrue2, truebrier = true.brier2)
n250shape2by5

# shape=2,by=5,n=500
date()
res <- sim.results(n = 500, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                   0.25), shape = 2, by = 5, time = seq(1.5, 28.5, by = 3))
```

```
date()
save(res, file = "resn500shape2by5M1000.Rdata")
n500shape2by5 <- summarize(res, truec = ctrue2, truebrier = true.brier2)
n500shape2by5

# shape=2,by=5,n=1000
date()
res <- sim.results(n = 1000, nsim = 1000, betas = c(-3, 0.01, 0.05, 0.30,
                  0.25), shape = 2, by = 5, time = seq(1.5, 28.5, by = 3))
date()
save(res, file = "resn1000shape2by5M1000.Rdata")
n1000shape2by5 <- summarize(res, truec = ctrue2, truebrier = true.brier2)
n1000shape2by5

######################### Chapter 7

cav1$fd.na <- cav1$fd
cav1$fd.na[is.infinite(cav1$fd.na)] <- NA
WeibullReg(Surv(time = la, time2 = fd.na, type = "interval2") ~
             dage + sex + pdiag, data = cav1, conf.level = 0.95)
model <- survreg(Surv(time = la, time2 = fd.na, type = "interval2") ~
                   dage + sex + pdiag, dist = "weibull", data = cav1)
summary(model)
alpha <- summary(model)$coefficients
alpha

sigma <- model$scale
mu <- alpha[1]
gamma <- 1/sigma
beta0 <- -mu/sigma
p <- length(alpha) - 1
cavbeta <- c(beta0, -alpha[2:(p + 1)]/sigma)
cavbeta

# Calculate hazard for each subject in the data
X <- model.matrix(model)
lambda <- exp(X %*% cavbeta)

# Calculate c-index for the actual data
system.time(ce <- c.exact(lambda, cav1$la, cav1$fd))
ce

# General c-index from Weibull distribution
system.time(cg <- c.general(lambda, cav1$la, cav1$fd, shape = gamma, M = 1000,
                           dist = "weibull"))
cg

# Brier score at selected time points
tseq <- seq(1.5, 28.5, by = 3)
cavBS <- brier.score(lambda, cav1$la, cav1$fd, shape = gamma, time = tseq)
data.frame(time=tseq, Brier=cavBS)

# With cross-validation
print(date())
n <- nrow(cav1)
lambdamini <- gammamini <- numeric(n)
for (i in 1:n) {
  cavmini <- cav1[-i, ]
```

```
  model <- survreg(Surv(time = la, time2 = fd.na, type = "interval2") ~
                     dage + sex + pdiag, dist = "weibull", data = cavmini)
  alpha <- summary(model)$coefficients
  sigma <- model$scale
  mu <- alpha[1]
  gammamini[i] <- 1/sigma
  beta0 <- -mu/sigma
  betamini <- c(beta0, -alpha[2:(p + 1)]/sigma)

  # Calculate cross-validated hazard for subject i
  # Use original model matrix
  lambdamini[i] <- exp(X[i, ] %*% betamini)
}

# General c-index from Weibull distribution
CVcg <- c.general(lambdamini, cav1$la, cav1$fd, shape = gammamini, M = 1000,
                  dist = "weibull")
CVcg

CVBS <- brier.score(lambdamini, cav1$la, cav1$fd, shape = gammamini, time = tseq)
data.frame(time=tseq, Brier=cavBS, BrierCV=CVBS)

print(date())
```