Master Thesis

Piecewise constant models for ICU infection problems

Mathematics

Track: statistical science for the life and behavioral sciences

Edwin Thoen

October 2013

Leiden University Medical Center

Universiteit Leiden

Thesis Advisor:
Prof. dr. H. Putter

Supervisor:
Prof. dr. J.J. Meulman

Abstract

The health of patients at the ICU is endangered by hospital-acquired infections. For some of these infections it is unclear whether the infection prolongs ICU stay. Simply comparing the lengths of stay of uninfected and infected patients would not give a valid answer. This comparison suffers from the immortal time bias, by which we compare survival of groups that are formed during the actual survival. The three-state illness-death model does allow us to make claims about the lengthening effect of infection. We distinguish three data types; complete information (infection state monitored daily), panel data (fragmented monitoring of infection state during stay), and endpoint-only data (no monitoring of infection state during stay). We show that from each of the three data types piecewise constant transition rates can be estimated. All methods are implemented in `R`, as well as a number of functions that can be applied on fitted models. Simulation studies and applications to real data show that estimating piecewise constant models from endpoint-only data often results in very wide confidence intervals for the transition rates. Confidence intervals can be narrowed somewhat with model restrictions, however useful results are difficult to obtain from endpoint-only data. Results are discussed, with suggestions for further research.

Contents

1 ICU Infection Data

1.1 Surviving the ICU

Ironically the intensive care unit (ICU) of a hospital is a hazardous place to reside. Critically ill patients are at risk of getting further diseased by a number of hospital infections. Patients fighting for their lives also have to fight off bacteria which are threatening them with further illness. While some hospital-acquired (or nosociomal) infections are relatively harmless for the patient, others can cause an immediate threat for the patient's life. The effect of an infection on the length of the hospital stay of the patient is of great interest from both a medical and an economical perspective (Reed & Kemmerly, 2009 [1]).

Medical doctors might wonder whether acquiring a certain infection does lengthen ICU stay. Answering this question is less straightforward than it might look at first sight. Just comparing the lengths of stay (LOS) of infected and uninfected patients does not provide an answer, because infected patients will always have a longer LOS even when the infection was harmless and had no effect on LOS. Simply because patients with a longer LOS, due to causes other than getting infected, are exposed longer to the threat of acquiring the infection. Even though acquiring an infection does not effect the LOS, the LOS still effects the acquiring of infection.

The following R simulation illustrates this. We take a thousand patients and sample both the LOS and the time it takes to get infected independently from an exponential distribution with $\lambda = 1/30$. Patients who have a LOS shorter than the infection time will stay uninfected, those with a LOS longer than the infection time will get the infection. We then calculate the average LOS of both the infected and uninfected patients. Repeating this a thousand times and averaging the average LOS gives the following result

```
> sim <- function(n){
>   LOS <- rexp(n, 1/30); Inf.time <- rexp(n, 1/30)
>   Infec <- factor(LOS > Inf.time, labels = c('Uninfected', 'Infected'))
>   tapply(LOS, Infec, mean)}

> rowMeans(replicate(1000, sim(1000)))

Uninfected   Infected
  14.97847   45.01360
```

Patients who have acquired the infection during the stay have an expected LOS that is three times as long as that of patients who did not get infected. Not because the infection prolonged ICU stay, but because the patients who happened to stay longer at the ICU were more prone to getting infected. This is a common phenomenon in survival theory and is called the *Immortal Time Bias* or the *Oscar Fallacy*.[1] We cannot claim that a discriminating factor shortens or prolongs survival if the group discrimination takes place after the onset point at which we have started measuring the survival time. In our case the claim that the acquiring of an infection during ICU stay prolongs ICU stay cannot be made if we observe that infected patients have a

[1]Redelmeier & Singh (2001) [2] showed that the life expectancy of individuals nominated for an Academy Award was about four years longer than that of the average American. They claimed that this was due to the high social status of the actors, which supposedly enhanced general health. However one has to make it at least until early adulthood before one is able to receive a nomination, therefore discriminating between Oscar winning actors and ordinary citizens is a selection bias. After correcting for the bias, Oscar winners appeared to live as long as anybody else (Sylvestre, Huszti & Hanley, 2006 [3])

longer LOS than uninfected patients. We need a more sophisticated model to disentangle the effects of LOS on infection and of infection on LOS.

1.2 The Illness-Death Model

A three-state model provides the solution to this problem. Initially all patients reside in the first state which is being at the ICU without having acquired the nosociomal infection. As time progresses two things can happen to the patient that makes him move to a different state. He[2] can acquire the infection, which moves him to the infected state, or he can leave the ICU, which moves him to the discharged state.[3] Figure 1 gives a schematic representation of this model. This three-state model is commonly referred to as the illness-death model, because it is often used for situations at which initially healthy patients can either die without getting ill first, or fall ill from the infection of interest before dying.

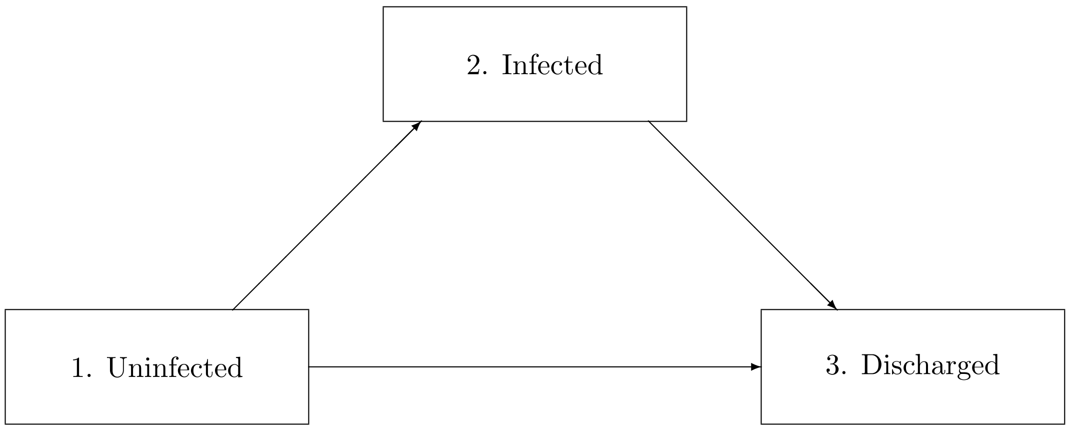

Figure 1: Schematic representation of the three-state, or Illness-Death, model.

From the data we can estimate for each time point t the probability rate of moving from one state to another, called the *transition intensities* or *rates* (comparable to the *hazard rate* in classical survival analysis), usually denoted by $\lambda_{lk}(t)$: the approximate probability of moving from state l to state k at time point t. Our illness-death model contains three transitions:

- moving from *Uninfected* to *Infected*, with transition rate $\lambda_{12}(t)$
- moving from *Uninfected* to *Discharged*, with transition rate $\lambda_{13}(t)$
- moving from *Infected* to *Discharged*, with transition rate $\lambda_{23}(t)$

After estimation of the three parameters from data we can consider ratio $\lambda_{13}/\lambda_{23}$ to have an indication whether infection prolongs ICU stay.

1.3 Three Data Types

Different data types can be obtained from studies of ICU infection. The amount of information in the data influences the accuracy with which the three parameters can be estimated. Below we describe the three data types that we can distinguish when doing ICU research, decreasing

[2]or She for that matter.

[3]For simplicity we do not discriminate between those being discharged, and those deceased at the ICU. They are all considered being discharged.

in both the amount of information present in the data and in the burden they are for the researcher.

1.3.1 Complete Information Data

In the ideal case we monitor the patient every single day during his ICU stay. If he was infected during this stay we know exactly at which day this infection took place. From these data we can estimate the transition intensities either parametrically or nonparametrically. In the latter case we do not need to assume the transition times to be drawn from a specified probability distribution, we simply estimate the transition intensities from the data for each time point. This is the multi-state elaboration of the Kaplan-Meier estimator. For estimating the transition intensities from complete information data, with or without covariates, the `mstate` package in `R` can be used (de Wreede *et al.*, 2011 [4]).

1.3.2 Panel Data

Often it is not feasible to monitor the infection state of the patients on a daily basis. Testing for infections might be time consuming and/or invasive for the patient, permitting only a few tests during ICU stay. We refer to data sets that have fragmented observations of the infection state as panel data. The time points at which the patients are tested can be equal for all patients, but can also differ from patient to patient. In both cases the `msm` package in `R` can be used to estimate the transition intensities, also allowing for covariates (Jackson, 2011 [5]). A disadvantage of this data type is that it does not allow for time-dependent covariates, including time itself. We are thus restricted to transition intensities that are constant over time, making the transition rate of moving from j to k equal for all t. The only relaxation of this restriction is the estimation of piecewise constant transition rates. The rates are then constant over intervals within the total study time, but can differ between the intervals.

1.3.3 Endpoint-Only Data

The minimal observations from which we can still estimate the transition intensities is when we only observe the infection state of the patients at the moment of discharge. The only information we have for each patient is the time of discharge and whether he left the ICU either infected or uninfected. Bootsma (2005) [6] recognised that from these data transition rates can still be estimated, however only when we let them to be constant over time. He derived the likelihood of the three constant parameters for this data type. By assuming transition intensities to be constant over time, we assume the three possible transitions of the illness-death model to be random variables following an exponential distribution.

1.4 Piecewise Constant Transition Rates

As noted by Bootsma (2005) [6], assuming the transition rates to be constant over the entire time period is often unrealistic. In many situations it would make sense to allow for more flexibility by letting the transitions intensities vary over time points. For example, many patients might be discharged uninfected in the first few days of the ICU stay, reflected by a high transition rate λ_{13}. Patients that are not discharged the first few days are expected to stay a long time, the transition rate then drops as t gets larger. If we can only fit models with fixed transition rates, our models might fit poorly to data of this nature. Many other scenarios are possible that would yield poor fit. If we have complete information data we can estimate the transition rates nonparametrically and allow for the necessary flexibility. In case of panel or endpoint-only data non-parametric transition rates cannot be estimated, but as we will show it is possible to estimate piecewise constant transition rates. If we divide the entire time period into a number of intervals we can estimate the transition rates for each of the intervals, allowing for some flexibility over time.

In the case of complete information data this thus is a suboptimal thing to do, since the transition rates can also be estimated nonparametrically. But for the panel and endpoint-only data this is an improvement to models at which the transition rates are fully time-independent. The `msm` package already allows for the estimation of piecewise constant parameters in the multi-state setting, but in case of the illness-death model we can derive the likelihood explicitly. Piecewise constant models have not been applied before at endpoint-only data, as far as we know. Since the endpoint-only data are the most convenient and inexpensive data to gather it is of interest to compare the parameters estimated from this data type to those estimated from the more informative data types.

2 Preliminaries in Survival Analysis and Multi-State Theory

This section provides the building blocks for our piecewise constant models. First a small overview of classical survival analysis is provided, since multi-state models are an elaboration of this. Especially the properties of the exponential model for survival analysis are explored. Finally we will look at the multi-state theory for the illness-death model.

2.1 Survival Analysis

Survival analysis deals with time to event data, at which for each subject the time from a starting point until a predefined event is measured. It is usually applied in a medical setting, often to measure the surviving time of severely ill patients (where the event is death). A number of interrelated parameters are used to describe survival characteristics, the following results are all adopted from Klein and Moeschberger (2003) [7].

2.1.1 The Survival Function

The survival function reflects the probability for an individual to stay event-free (surviving) until a certain time point since his time onset. If we denote the random variable of experiencing the event by T, the survival function is

$$S(t) = P(T > t). \tag{1}$$

Since all subjects are still event-free at time onset $S(0) = 1$. From $t = 0$ the survival function strictly decreases making $S(t) \geq S(t + i)\ \forall\ t, t + i$, where $i \geq 0$.

2.1.2 The Probability Density Function

Like all continuous random variables the probability for T to take on a certain value t is described by the density function $f(t) = P(T = t)$. The corresponding distribution function reflects the probability that someone has experienced the event between the onset and a time point $F(t) = P(T < t)$, which makes the survival function the complement of the distribution function

$$S(t) = 1 - F(t) = \int_t^\infty f(s)\mathrm{d}s. \tag{2}$$

The density is related to the survival function by

$$f(t) = -\frac{\mathrm{d}}{\mathrm{d}t} S(t). \tag{3}$$

2.1.3 The Hazard Function

The final parameter to describe the characteristics of survival data is the hazard function, or hazard rate. The hazard rate is the approximate probability rate of experiencing the event at the next instance from time t, given that one has not experienced the event before t

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{P(t \leq T < t + \Delta t \mid T \geq t)}{\Delta t}. \tag{4}$$

The hazard rate is related to the survival in the following way

$$\lambda(t) = \frac{f(t)}{S(t)} = -\frac{d}{dt} \ln(S(t)), \tag{5}$$

and conversely the survival function is related to the hazard by

$$S(t) = \exp\left(- \int_0^t \lambda(s) \mathrm{d}s \right). \tag{6}$$

So we can express the density as the product of the hazard rate and the survival

$$f(t) = \lambda(t) S(t). \tag{7}$$

2.2 Estimating the Survival Parameters

We can estimate the survival parameters either parametrically or nonparametrically. In the latter case we don't assume the event as coming from a known probability distribution. We simply estimate the survival for each time point t from the data with the famous Kaplan-Meier estimator (Kaplan & Meier, 1958 [8]). We can also assume the survival times as coming from a parametric probability distribution. Then we estimate the parameters from the survival times using maximum likelihood. Since we will adopt multi-state models at which we assume our data as coming from a parametric distribution, we will only explore parametric estimation.

2.2.1 Parametric Estimation

Common probability distributions that are assumed for survival data are the exponential distribution and the Weibull distribution, but many others are possible (Klein & Moeschberger, 2003 [7]). Maximum likelihood estimation of the distribution parameters is complicated by the special nature of survival data. Possibly we don't observe the survival times of a patient, we only know he did survive until a certain time. We say that this patient is censored at time t_i, if we have no further information about the patient after this time point.[4] Observed events and censored patients add different information to the likelihood.

Let t_i be the time at which subject i either experiences the event or is lost to follow-up. Let δ_i be an indicator, which is 1 if the subject experienced the event and 0 if he is lost to follow-up. If subject i has experienced the event at time t_i his contribution to the likelihood is

[4]To be precise, this is called right-censoring, we also distinguish left-censoring and interval-censoring. These are less common and will not be treated here.

the density at t_i. If he was lost to follow-up his contribution to the likelihood is the survival until t_i. Combining the above, the contribution of each patient to the likelihood is

$$P(t_i, \delta_i) = f(t_i)^{\delta_i} S(t_i)^{1-\delta_i}. \tag{8}$$

Using equation (7) we can construct the full likelihood as

$$\mathcal{L} = \prod_{i=1}^{n} \lambda(t_i)^{\delta_i} S(t_i). \tag{9}$$

2.2.2 MLE for the Exponential

Since the exponential distribution will be used extensively, we look at the maximum likelihood results for this distribution. Knowing that the density of the exponential distriubtion is $f(t) = \lambda e^{-\lambda t}$, it follows from (2) that $S(t) = e^{-\lambda t}$ and subsequently from (5) that $\lambda(t) = \lambda$. Thus the hazard rate $\lambda(t)$ does not depend on t since it is λ at each time point, implying that patients have an equal probability of experiencing the event at all time, given that they have not experienced the event by that time. This property makes the exponential distribution quite restrictive. For many survival problems the assumption that the survival times are a sample from the exponential distribution is unrealistic, making the model usually fit poorly.

If out of the total n subjects m experience the event, equation (9) for the exponential distribution can be written as

$$\mathcal{L} = \lambda^m \prod_{i=1}^{n} e^{-\lambda t_i}.$$

Taking the log and solving for λ we see that the mle of λ is

$$\hat{\lambda} = \frac{m}{\sum_{i=1}^{n} t_i}. \tag{10}$$

The mle is thus the number of people experiencing the event, divided by the sum of observed event times and censoring times. In case of no censoring the mle would become the ordinary mle of the exponential distribution $\frac{n}{\sum t_i}$. $\hat{\lambda}$ gets smaller as a larger part of the sample is censored for the same t_i, increasing expected survival.

2.3 Multi-State Theory

The basic survival model as described in the previous section can be regarded as a two-state model. As long as the subjects are event free they are in the first state, as soon as they experience the event they move to the second state. The hazard is the transition rate for moving from state 1 tot state 2. The illness-death model shown in Figure 1 adds a third state to the classicial survival model to which patients can move. We now have three possible state transitions, for each time point t there are three transition intensities, $\lambda_{12}(t)$, $\lambda_{13}(t)$, and $\lambda_{23}(t)$.[5] In the classical survival case we express the probability that one is still event-free by the survival function, making the probability of having experienced the event its complement. If we want to do prediction in the three-state model we have more options to consider. The following probabilities are all adopted from Putter, Fiocco & Geskus (2007) [9]. Let R be the time at which a patient got infected, and T be the time a patient was discharged. We want to predict

[5]We assume that infected patients will not move back to the uninfected state during ICU stay. Patients in state 3 have left the ICU once they are infected, and thus will not move back to states 1 and 2. Connections between states are thus all unidirectional in this situation.

the state of a patient at time point t given his state at time point u. We denote the event history at u by $H_k(u)$, meaning that the patient is in state k at time u. In our illness-death model there are six different options to consider

$$\begin{aligned}
P_{11}(u,t) &= P(R > t, T > t | H_1(u)),\\
P_{12}(u,t) &= P(R \leq t, T > t | H_1(u)),\\
P_{13}^1(u,t) &= P(T \leq t, T < R | H_1(u)),\\
P_{13}^2(u,t) &= P(R \leq T \leq t | H_1(u)),\\
P_{22}(u,t) &= P(T > t | H_2(u)),\\
P_{23}(u,t) &= P(T \leq t | H_2(u)),
\end{aligned} \tag{11}$$

here P_{13}^1 and P_{13}^2 indicate the probabilities of having moved from state 1 to state 3 by time t, without and with moving through state 2 in between respectively. All these probabilities can be expressed in terms of survival parameters. Since all patients are in state 1 at the beginning of the study we can define $S_1(t)$ as the survival in 1, the probability of still being in state 1 at time point t. In this situation we have two transition rates, so we can expand equation (6) to $S_1(t) = \exp\left(-\int_0^t \lambda_{12}(s) + \lambda_{13}(s)\mathrm{d}s\right)$. A second survival is residing in state 2 after being infected. The probability for a patient to be still in 2 by t, given that he moved to state 2 at r is $S_{2,r}(t) = \exp\left(-\int_r^t \lambda_{23}(s)\mathrm{d}s\right)$. From the two survival parameters and the three transition intensities we can construct the six probabilities.

The probability of still being in the ICU without being infected at time t given that the patient was uninfected at u, is given by

$$P_{11}(u,t) = \frac{S_1(t)}{S_1(u)}. \tag{12}$$

If a patient was infected by u, the probability he is still at the ICU at time t is given by

$$P_{22,r}(u,t) = \frac{S_{2,r}(t)}{S_{2,r}(u)}. \tag{13}$$

The probability of being still in the ICU and being infected at t given that he was uninfected by u is given by

$$P_{12}(u,t) = \frac{\int_u^t \lambda_{12}(r)S_1(r)P_{22,r}(r,t)\mathrm{d}r}{S_1(u)}. \tag{14}$$

In this scenario the patient moves from state 1 to state 2 at time r, where $u < r \leq t$ and stays in state 2 at least until t. From equation (7) we see that the numerator is the integral over the density of R between u and t multiplied by the probability of staying in state 2 until the end of the interval.

The probability for a patient who was uninfected at u of being discharged uninfected at t is given by

$$P_{13}^1(u,t) = \frac{\int_u^t \lambda_{13}(s)S_1(s)\mathrm{d}s}{S_1(u)} \tag{15}$$

and of being discharged infected at t if given by

$$P_{13}^2(u,t) = \frac{\int_u^t \lambda_{12}(r)S_1(r)P_{23,r}(r,t)\mathrm{d}r}{S_1(u)}. \tag{16}$$

Finally a patient that is already infected at time point u has a probaility of being discharged by t

$$P_{23,r}(u,t) = \frac{\int_u^t \lambda_{23}(s) S_{2,r}(s) \mathrm{d}s}{S_{2,r}(u)} = \frac{S_{2,r}(u) - S_{2,r}(t)}{S_{2,r}(u)} = 1 - \frac{S_{2,r}(t)}{S_{2,r}(u)}. \tag{17}$$

From these building blocks we can start develop the methods to estimate the piecewise constant model from the three data types.

3 Piecewise Constant Models

Before we turn to piecewise constant models, we will derive the constant rate likelihood for endpoint-only data. This was previously done by Bootsma (2005) [6], but we do it from a multi-state perspective. We then proceed by estimating piecewise constant models from complete information, panel, and endpoint-only data.

3.1 Endpoint-only Data, Constant Parameters

In the endpoint-only situation, when we only have the discharge time and the infection status at discharge available, there are two options. If the patient leaves the ICU uninfected, he has moved to state 3 without moving through state 2. In case he was infected, he has moved from state 1 to state 3 through state 2. The probability that a patient is not yet discharged and uninfected at time point t is given by

$$S_1(t) = P_{11}(0,t) = e^{-(\lambda_{12}+\lambda_{13})t}, \tag{18}$$

when both transition rates are constant over time. From equation (15) in the previous section it follows that with a constant transition rate λ_{13} the probability of being discharged before time point t while being uninfected is given by

$$P_{13}^1(0,t) = \int_0^t \lambda_{13}\ e^{-(\lambda_{12}+\lambda_{13})s} \mathrm{d}s, \tag{19}$$

which is the distribution function for being discharged uninfected. We arrive at the density of being discharged uninfected by simply dropping the integral

$$f_{13}^1(t) = \lambda_{13}\ e^{-(\lambda_{12}+\lambda_{13})t}. \tag{20}$$

The probability of being discharged while having an infection follows from equation (16)

$$\begin{aligned} P_{13}^2(0,t) &= \int_0^t \lambda_{12}\ \ e^{-(\lambda_{12}+\lambda_{13})r}\ \left(1 - \frac{S_2(t)}{S_2(r)}\right) \mathrm{d}r \\ &= \int_0^t \lambda_{12}\ \ e^{-(\lambda_{12}+\lambda_{13})r}\ (1 - e^{-\lambda_{23}(t-r)}) \mathrm{d}r, \end{aligned}$$

where r is the unknown moment of getting infected. Since r is unknown to us we cannot simply drop the integral to obtain the density. Therefore the integral needs to be solved first, which yields

$$P_{13}^2(0,t) = \frac{\lambda_{12}}{\lambda_{1.}} + \frac{\lambda_{12}\lambda_{23}}{\lambda_{1.}(\lambda_{1.} - \lambda_{23})} e^{-(\lambda_{12}+\lambda_{13})t} - \frac{\lambda_{12} e^{-\lambda_{23}t}}{\lambda_{1.} - \lambda_{23}}. \tag{21}$$

Here $\lambda_{1.} = \lambda_{12} + \lambda_{13}$, it is the sum of the transition rates of leaving state 1. From this we can obtain the density by differentiating with respect to t

$$f_{13}^2(t) = \frac{\lambda_{12}\lambda_{23}}{\lambda_{1.} - \lambda_{23}} \left(e^{-\lambda_{23}t} - e^{-\lambda_{1.}t}\right). \tag{22}$$

From (20) and (22) we can construct the likelihood. Let δ_i be a boolean that codes for leaving the ICU either infected or uninfected

$$\delta_i = \begin{cases} 1 & \text{if } r_i < t_i \\ 0 & \text{if } r_i \geq t_i \end{cases}.$$

The full likelihood becomes

$$\mathcal{L} = \prod_{i=1}^{n} \left[\lambda_{13}\ e^{-\lambda_{1.}t_i}\right]^{1-\delta_i} \left[\frac{\lambda_{12}\lambda_{23}}{\lambda_{1.} - \lambda_{23}} \left(e^{-\lambda_{23}t_i} - e^{-\lambda_{1.}t_i}\right)\right]^{\delta_i}. \tag{23}$$

Using a general purpose algorithm like R's `optim` the log of this likelihood can be maximized over the parameters to obtain the estimates $\hat{\lambda}_{12}, \hat{\lambda}_{13}$, and $\hat{\lambda}_{23}$.

3.2 Piecewise Constant Models

This section covers the piecewise constant models. First we consider estimation of the model from complete information data. Next we build on the above model to obtain the piecewise constant model estimation from panel and endpoint-only data.

3.2.1 Complete Information Data

In the complete information case we exactly know in which state the patient resides at every day of his ICU stay. If we consider the transition times R and T to be distributed exponentially, we assume the transition rates to be constant over time. The maximum likelihood estimates of the parameters are given by the number of patients experiencing the transition, divided by the total number of days that the patients were at risk for that transition. If out of the n patients in the study m are somewhere during their stay, the mle of the parameters are given by

$$\hat{\lambda}_{12} = \frac{m}{\sum t_{i1}} \quad \hat{\lambda}_{13} = \frac{n-m}{\sum t_{i1}} \quad \hat{\lambda}_{23} = \frac{m}{\sum t_{i2}},$$

here t_{i1} and t_{i2} are the time periods in days that patient i resided in state 1 and state 2 respectively.

If we don't want to assume constant transition rates over time because this seems unrealistic for our data type, we can allow for some flexibility by dividing the total study time into j intervals and estimate the transition rates for each of these intervals. Each parameter is then estimated j times. The mle's are now the number of patients experiencing the transition in the interval, divided by the total number of days the patients were at risk for experiencing the transition during the interval. If we define $n_{lk}^{(j)}$ as the number of patients that have moved from state l to state k during interval j, and $t_{il}^{(j)}$ the number of days patient i resided in state l during interval j. Then mle's of the three parameters for interval j are

$$\hat{\lambda}_{12}^{(j)} = \frac{n_{12}^{(j)}}{\sum t_{i1}^{(j)}} \quad \hat{\lambda}_{13}^{(j)} = \frac{n_{13}^{(j)}}{\sum t_{i1}^{(j)}} \quad \hat{\lambda}_{23}^{(j)} = \frac{n_{23}^{(j)}}{\sum t_{i2}^{(j)}}. \tag{24}$$

The function `icu.pwc.fi` in appendix A can be used to obtain the estimates on a data set with the variables infection times and discharge times present.

3.2.2 Panel Data

Now we look at the sitation in which the infection state of the patient is monitored at fixed time points τ_k, with $k = 1, ..., j-1$, creating j intervals. For each patient the infection state is assessed after spending a predetermined number of days in the ICU, given that he is not yet discharged at this time point. Because all patients are uninfected at the time of ICU admission and are checked at the end of each interval, we know for all infected patients in which interval the infection took place. As with the endpoint-only data the patients are also checked for infection at discharge. If τ_j^b and τ_j^e mark the beginning and end of interval j respectively, there are in total six possibilities of the states of a patient at these interval seperating points.[6] Each possibility will have its own contribution to the likelihood.

A patient who started the interval uninfected and who is still uninfected at the end of the interval has survived in state 1. This probability follows from equation (12)

$$P_{11}(\tau_j^b, \tau_j^e) = e^{-\lambda_{1.}^{(j)}(\tau_j^e - \tau_j^b)}. \tag{25}$$

If a patient started the interval uninfected, the probability that he will be infected and not yet discharged by the end of the interval is given by equation (14). If we plug in the exponential parameters for hazard and survival and solve the integral, we will find this survival to be

$$P_{12}(\tau_j^b, \tau_j^e) = \frac{\lambda_{12}^{(j)}}{\lambda_{1.}^{(j)} - \lambda_{23}^{(j)}} \left(e^{-\lambda_{23}^{(j)}(\tau_j^e - \tau_j^b)} - e^{-\lambda_{1.}^{(j)}(\tau_j^e - \tau_j^b)} \right). \tag{26}$$

The final survival option is a patient who started the interval already infected and who was not discharged during the interval

$$P_{22}(\tau_j^b, \tau_j^e) = e^{-\lambda_{23}^{(j)}(\tau_j^e - \tau_j^b)}. \tag{27}$$

Next we turn to the likelihood contributions for patients who are discharged during interval j. Firstly a patient who was uninfected at the beginning of the interval and was discharged uninfected at $\tau_j^b < t \leq \tau_j^e$. This is the same density as (20), but with the difference that it applies to discharge in interval j only

$$f_{13}^1(t - \tau_b^j) = \lambda_{13}^{(j)} \; e^{-\lambda_{1.}^{(j)}(t - \tau_j^b)}. \tag{28}$$

The density for a patient who was known to be uninfected at τ_j^b, who was discharged during interval j, and who appeared to be infected is similar to the density previously seen in equation (22), but now only for interval j

$$f_{13}^2(t - \tau_b^j) = \frac{\lambda_{12}^{(j)} \lambda_{23}^{(j)}}{\lambda_{1.}^{(j)} - \lambda_{23}^{(j)}} \left(e^{-\lambda_{23}^{(j)}(t - \tau_b^j)} - e^{-\lambda_{1.}^{(j)}(t - \tau_b^j)} \right). \tag{29}$$

Finally a patient who started the interval infected and who is discharged during the interval contributes

$$f_{23}(t - \tau_j^b) = \lambda_{23}^{(j)} \; e^{-\lambda_{23}^{(j)}(t - \tau_j^b)}. \tag{30}$$

The Markov assumption for multi-state models states that transition rates to move from state l to state k at time t does depend on state l only and not on the states frequented before t

[6]We let the split points of the intervals coincide with the measuring of the infection state. However this is not strictly necessary, as mentioned in Jackson (2011) piecewise constant models can also be estimated when measuring points and split points differ, or even when measuring points differ from patient to patient.

(Putter, Fiocco & Geskus, 2007) [9]. Using this property we can write the full densities for being discharged either uninfected or infected at time t as a product of interval survivals and densities, which is an example of a Chapman-Kolmogorov equation. To be discharged uninfected during interval j necessarily implies that the patient has resided in state 1 during intervals 1 to $j-1$

$$f^1_{13}(t) = P_{11}(0, \tau^b_j) f^1_{13}(t - \tau^b_j) = P_{11}(0, \tau^e_1) \dots . P_{11}(\tau^b_{j-1}, \tau^e_{j-1}) f^1_{13}(t - \tau^j_b). \tag{31}$$

A patient who is discharged infected has moved from 1 to 2 before moving to 3 somewhere between 0 and t. Again if t falls into the j^{th} interval, the probability of being discharged infected is

$$f^2_{13}(t) = P_{11}(0, \tau^b_j) f^2_{13}(t - \tau^b_j) + P_{12}(0, \tau^b_j) f_{23}(t - \tau^b_j). \tag{32}$$

There will be multiple routes to move from state 1 to state 2 between time 0 and τ^b_j if $j > 2$.

Each patient that is still at the ICU during interval j contributes either one of the three survivals (25) – (27) or one of the three densities (28) – (30) to the likelihood for the given interval. From these contributions we can construct the interval likelihood, which can be maximized over the parameters to obtain the three parameter estimates for the specific interval. By recording the infection states of all patients at the beginning and end of the interval, as well as by discharge during the interval, the interval likelihoods do not depend on each other and can be maximized seperately. We first introduce some auxiliary variables that allow us to define the likelihood in a concise fashion. Let t_{ij} be the time that patient i spent in the ICU during interval j

$$t_{ij} = \begin{cases} 0 & \text{if } t_i \leq \tau^b_j \\ \tau^e_j - \tau^b_j & \text{if } t_i > \tau^e_j \\ t_i - \tau^b_j & \text{if } \tau^b_j < t_i \leq \tau^e_j \end{cases} . \tag{33}$$

Next we define three indicator variables. The indicator δ^b_{ij} codes for being infected at the beginning of interval j

$$\delta^b_{ij} = \begin{cases} 1 & \text{if } r_i \leq \tau^b_j \\ 0 & \text{if } r_i > \tau^b_j \end{cases} , \tag{34}$$

and δ^e_{ij} for being infected at the end of interval j

$$\delta^e_{ij} = \begin{cases} 1 & \text{if } r_i \leq \tau^e_j \\ 0 & \text{if } r_i > \tau^e_j \end{cases} . \tag{35}$$

Finally γ_{ij} indicates wheter the patient was discharged during interval j

$$\gamma_{ij} = \begin{cases} 1 & \text{if } \tau^b_j < t_i \leq \tau^e_j \\ 0 & \text{otherwise} \end{cases} . \tag{36}$$

Now we can construct the likelihood for interval j as

$$\mathcal{L}^j = \prod_{i=1}^{n} \left[\lambda_{13}^{(j)\gamma_{ij}} e^{-\lambda^{(j)}_{1.} t_{ij}} \right]^{(1-\delta^b_{ij})(1-\delta^e_{ij})} \left[\lambda_{23}^{(j)\gamma_{ij}} \frac{\lambda^{(j)}_{12}}{\lambda^{(j)}_{1.} - \lambda^{(j)}_{23}} \left(e^{-\lambda^{(j)}_{23} t_{ij}} - e^{-\lambda^{(j)}_{1.} t_{ij}} \right) \right]^{(1-\delta^b_{ij})\delta^e_{ij}}$$
$$\left[\lambda_{23}^{(j)\gamma_{ij}} e^{-\lambda^{(j)}_{23} t_{ij}} \right]^{\delta^b_{ij}} . \tag{37}$$

The first part of the likelihood is for those who start and end the interval uninfected. The second part shows the contribution of those who started the interval uninfected, but end up being infected at the end of the interval. The final part is the contribution for patients who start the interval already infected. Each of the pieces of the above likelihood are of the shape $\lambda(t)S(t)$, just as we have seen previously with the likelihood for ordinary survival (equation (9)). If the patient was not discharged during the interval the transition intensity is 'switched off' by the γ_{ij}, making the contribution a survival. If the patient was discharged during the interval the λ_{lk} was 'switched on', making it a density. To obtain the parameters the log of the likelihood can be optimized using a general purpose algorithm.

The estimation of the parameters is implemented in the `R` function `icu.pwc.pan`, of wich the code can be found in appendix A.

3.2.3 Endpoint-Only Data

Finally we turn to the situation at which we only have observed the time of discharge and the infection state at discharge. Even though we have not observed the infection state at fixed time points as with the panel data, we can still estimate a piecewise constant model. We can split the total study time at any point, creating j intervals.[7] To distinguish from the situation at which we observe the patient's infection state at the split point, we no longer indicate the split points with τ_k but with ϕ_k. For patients who are discharged uninfected the infection state at all ϕ_k is known, since we assume moving back from the infected to the uninfected state impossible. Patients that are discharged infected have an unknown infection state at the ϕ_k points, since we have no information at all when patients moved from the uninfected to the infected state. In case of an unknown state at a time point of interest we can still determine the likelihood contribution of a patient. For the infected patient we know that his state at all the ϕ_j's (for which he was not yet discharged) was either uninfected or infected. His likelihood contribution is the sum over all the possible routes through these two states that will lead to infected discharge at t_i (Jackson, 2011) [5].

Let us first consider the two interval situation, at which we only have one split point ϕ_1. Again we need some auxiliary variables in order to construct the likelihood. The interval time of patient i in interval j is

$$t_{ij} = \begin{cases} 0 & \text{if } t_i \leq \phi_j^b \\ \phi_j^e - \phi_j^b & \text{if } t_i > \phi_j^e \\ t_i - \phi_j^b & \text{if } \phi_j^b < t_i \leq \phi_j^e \end{cases} . \tag{38}$$

The variable γ_{ij} codes for discharge of individual i in interval j

$$\gamma_{ij} = \begin{cases} 1 & \text{if } \phi_j^b < \ t_i \leq \phi_j^e \\ 0 & \text{otherwise} \end{cases} . \tag{39}$$

And δ_i codes for being discharged infeceted or not

$$\delta_i = \begin{cases} 1 & \text{if } r_i < t_i \\ 0 & \text{if } r_i \geq t_i \end{cases} . \tag{40}$$

In the two interval situation there are four options; being either discharged infected or uninfected in either the first or the second interval. Three of the four options give a certain likelihood contribution, but in case of infected discharge in the second interval we are uncertain about

[7]Though we need a number of observations in each interval in order to estimate the parameters.

the infection state at ϕ_1 because we don't know in which interval the patient was infected. Therefore for these patients the likelihood contribution is the sum over the likelihood of getting infected in the first interval and being discharged in the second and the likelihood of getting infected in the second interval. The full likelihood then will be

$$
\begin{aligned}
\mathcal{L} = \prod_{i=1}^{n} & \left(\lambda_{13}^{(1)\gamma_{i1}} e^{-\lambda_{1.}^{(1)} t_{i1}} \left[\lambda_{13}^{(2)} e^{-\lambda_{1.}^{(2)} t_{i2}} \right]^{\gamma_{i2}} \right)^{(1-\delta_i)} \Bigg(\left[\frac{\lambda_{12}^{(1)} \lambda_{23}^{(1)}}{\lambda_{1.}^{(1)} - \lambda_{23}^{(1)}} \left(e^{-\lambda_{23}^{(1)} t_{i1}} - e^{-\lambda_{1.}^{(1)} t_{i1}} \right) \right]^{\gamma_{i1}} \\
& \Bigg[e^{-\lambda_{1.}^{(1)} t_{i1}} \frac{\lambda_{12}^{(2)} \lambda_{23}^{(2)}}{\lambda_{1.}^{(2)} - \lambda_{23}^{(2)}} \left(e^{-\lambda_{23}^{(2)} t_{i2}} - e^{-\lambda_{1.}^{(2)} t_{i2}} \right) + \frac{\lambda_{12}^{(1)}}{\lambda_{1.}^{(1)} - \lambda_{23}^{(1)}} \left(e^{-\lambda_{23}^{(1)} t_{i1}} - e^{-\lambda_{1.}^{(1)} t_{i1}} \right) \\
& \lambda_{23}^{(2)} e^{-\lambda_{23}^{(2)} t_{i2}} \Bigg]^{\gamma_{i2}} \Bigg)^{\delta_i} .
\end{aligned} \tag{41}
$$

The second and third line in the above equation denote the sum of the likelihood possibilities for patients that are discharged infected in the second interval. We see that the likelihood already becomes quite cumbersome for two intervals only, at which there are only two pathways for patients with uncertainty about their infection state. If we were to add a third interval, a patient discharged infected in the third interval would have three possible pathways. The likelihood in equation in (41) then would be expanded with this pathway through the three states. This is not a very appealing option, not to mention adding even more intervals.

We turn to matrix notation to keep the likelihoods manageable. Let us remind us that in total there are six options for a patient to happen in each interval; three survivals (being not yet discharged at the end of the interval) and three densities (discharged during interval). These probabilities are similar to the ones given in equations (25) until (30), with the only difference that the τ_j's are replaced by ϕ_j's. For each interval we can construct a P-matrix and an f-matrix containing the probabilities of respecitively being still at the ICU at the end of the interval or being discharged during the interval. If we consider being discharged infected or uninfected as different states, both are 4x4 matrices, with most entries being zero since most moves between states are impossible.[8] The P-matrix for interval j is defined as

$$
\mathbf{P_j} = \begin{pmatrix} P_{11}(\phi_j^b, \phi_j^e) & P_{12}(\phi_j^b, \phi_j^e) & 0 & 0 \\ 0 & P_{22}(\phi_j^b, \phi_j^e) & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} .
$$

The rows reflect the state at the beginning of the interval, the columns the state at the end of the interval. Similarly the f-matrix for interval j is defined as

$$
\mathbf{f_j}(t_j) = \begin{pmatrix} 0 & 0 & f_{13}^1(t_j) & f_{13}^2(t_j) \\ 0 & 0 & 0 & f_{23}(t_j) \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} ,
$$

here t_j are all time points t falling in interval j with ϕ_j^b subtracted from it. To obtain the pathways through all possible state spaces between intervals 1 and $j-1$ we simply multiply the

[8]For simplicity the current P-matrix only contains the survival probabilities, these are the only elements we need when using the P-matrix for obtaining densities. In the next section we will use P-matrices to predict from the model parameters. Then we will complete the matrix with the distribution probabilies.

matrices $\mathbf{P_1}$ until $\mathbf{P_{j-1}}$. We only have to multiply the result with the density matrix $\mathbf{f_j}(t_j)$ to get the densities of the two infection states at time t in density matrix $\mathbf{f}(t)$

$$\mathbf{f}(t) = \mathbf{P_1}(0, \phi_1^e) \dots \mathbf{P_{j-1}}(\phi_{j-1}^b, \phi_{j-1}^e)\mathbf{f_j}(t_j).$$

When we define $\mathbf{P_{j-1}^*}$ as the product of all P-matrices from $\mathbf{P_1}$ until $\mathbf{P_{j-1}}$, then the density for patient i being discharged at time t, which falls in interval j is given by

$$\mathbf{f}_i = \mathbf{P_{j-1}^*} \begin{pmatrix} 0 & 0 & (1-\delta_i)f_{13}^1(t_j) & \delta_i f_{13}^2(t_j) \\ 0 & 0 & 0 & \delta_i f_{23}(t_j) \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \tag{42}$$

If the patient is discharged in the first interval $\mathbf{f_j}(t_j) = \mathbf{f}(t)$ and there is no multplication by a $\mathbf{P_{j-1}^*}$ matrix. The likelihood for patient i is simply the only non-zero element in $\mathbf{f}_i$, either the third or the fourth element on the first row. To obtain the full likelihood we multiply the individual likelihood elements.

Parameter estimation can be done with the function `icu.pwc` also to be found in appendix A. It can handle up to four intervals. The matrix form of the likelihood is very helpful for likelihood notation and calculation of probabilities once the parameters are estimated, as we will see in the next section. However parameter estimation is slowed down substantially when the likelihood is implemented in matrix form. Therefore the `icu.pwc` function is based on the fully written likelihood. Standard errors are obtained from the Hessian matrix, as returned by the `optim` function. Transition intensities are of course strictly positive, however with estimates close to zero we can get confidence interval lower bounds below 0. To prevent this we maximize the log of the parameters instead of the parameters itself. The confidence interval is calculated on the log scale, after which we transform back it to the ordinary scale. This yields asymmetric confidence intervals on the ordinary scales with higher lower and upper bounds than confidence intervals calculated on the ordinary scale. An additional advantage of maximizing the likelihood over the log of the parameters is that this yields more stable estimates than estimating the parameters on the ordinary scale. The optimization is less likely to fail or to get stuck in local optima.

4 Model Assessment

After fitting a piecewise constant model we need to check how well our model fits the data. First we explore a graphical way of looking at the model fit. Then we compare models fitted on the same data by a likelihood ratio test. Finally we look at how we should predict a state of patient after t days.

4.1 Diagnostics Plot

Model fit can be assessed informally with a diagnostics plot. By overlaying the theoretical density based on model fit and the empirical density estimate we have a graphical impression of how well our model fits the data. The theoretical densities are obtained by multiplying for each time point t the matrix $\mathbf{f_j}(t_j)$ with the relevant $\mathbf{P}$-matrices, as described in the previous section. We use two simulated data sets to illustrate its working with estimation from endpoint-only data.[9] Source codes for all the functions used can be found in appendix A.

[9]This section will apply the methods to endpoint-only estimates, because this was the approach of our primary interest. However all results can be applied to the other two data types as well.

The function `prob.icu` gives the density of uninfected and infectd patients on a grid between two time points, based on the parameter estimates from a `icu.pwc` model object. Its plotting function `plot.prob.icu` uses the `R` package `ggplot2` to produce the density plot. First we generate a data set of 2000 patients with the following parameter values, $\lambda_{12} = .08$, $\lambda_{13} = .08$ and $\lambda_{23} = .06$. Subsequently we estimate a piecewise constant model with split points at $\phi_1 = 7$ and $\phi_2 = 15$ and plot the theoretical density. Using the function `stat_density` from `ggplot2` we can overlay the empirical density.

```
> eo.dat <- cre.dat(2000, .08, .08, .06)
> two.phi <- icu.pwc(eo.dat, 'Discharge', 'Infection', 1, c(7, 15))
> plot(prob.icu(two.phi, 0, 50, .1)) +
+ stat_density(data = eo.dat, aes(x = Discharge, y = ..count../2000,
+ group = factor(Infection)), col = 'red',
+ position = 'identity', geom = 'line', lty = 2) +
+ xlim(c(0,50))
```

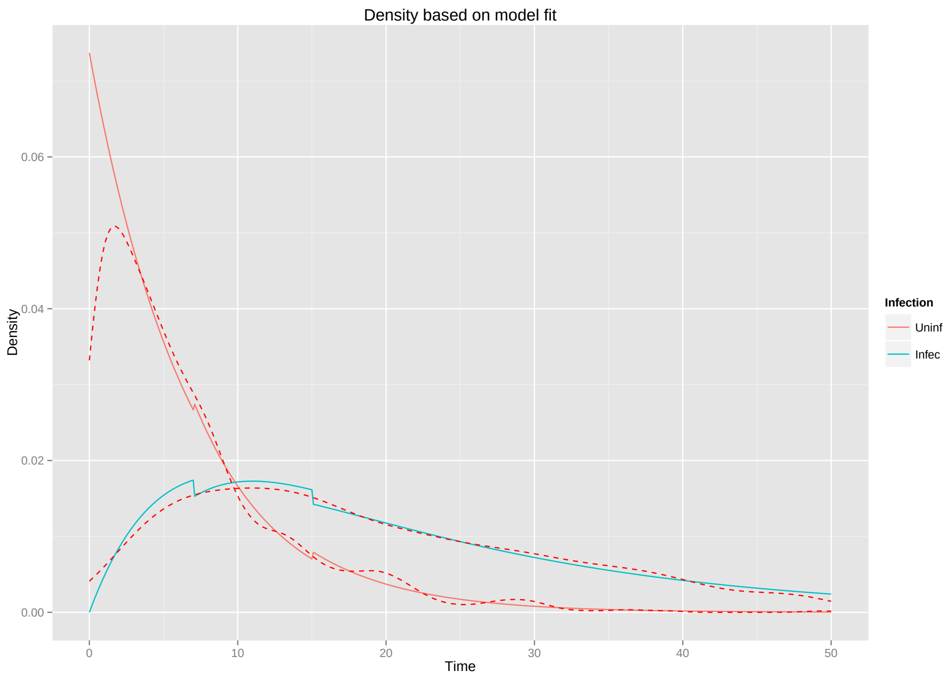

Figure 2: Comparing the two split point model density (solid lines) to the empirical density (red dashed lines).

The result is displayed in Figure 2. The model fits very well since all three random variables (infection, discharge uninfected, discharge infected) are drawn from exponential distributions. The theoretical density makes jumps at the points where the parameters change, which differ a little over the intervals due to randomness. Note that the bending of the empirical density near the y-axis is due to the kernel smoothing method, which averages over the density near a point. Since the density for negative values of t is 0, the density is flattened here.[10] We know that

[10]Alternatively the diagnostics plot can be made with a histogram. The disadvantage of this however is that it will show more fluctuation due to randomness, especially in smaller studies. The best approach is to glance at the histogram before constructing the diagnostics plot in order to see whether the empirical density is ascending during the first few days.

estimating a piecewise constant model is not necessary in this situation, since the parameters are constant over the full study. The plot also suggests that we might be overfitting by allowing for split points, and we might fit a model without split points as well. The diagnostics plot of the model without split points in Figure 3 reflects that this model indeed fits as well as the model with piecewise constant parameters.

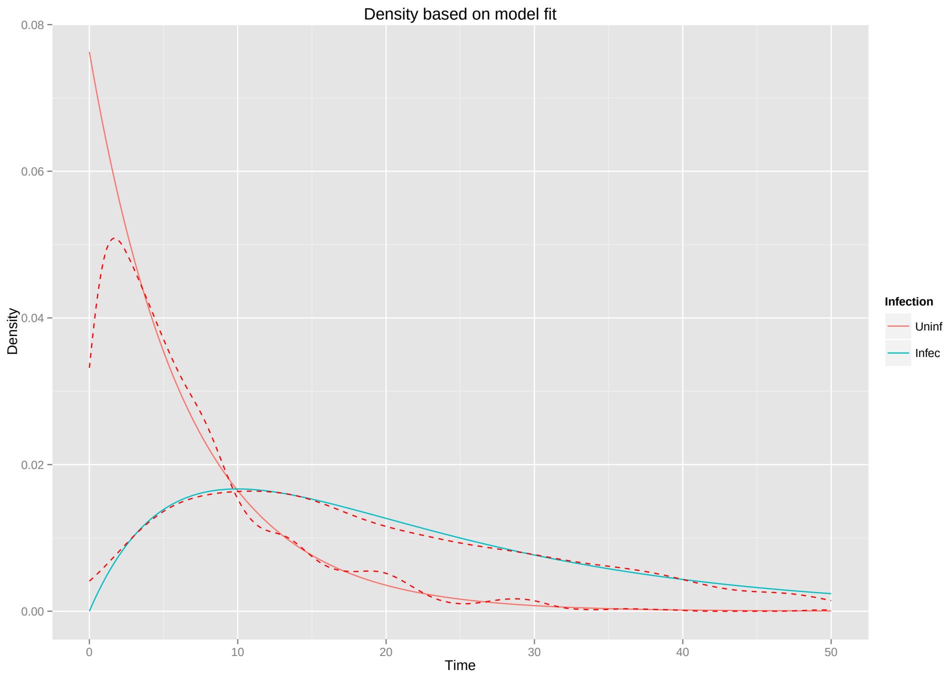

Figure 3: Diagnostics plot for the model with constant transition rates.

In real life transitions intensities are almost never fully equal over time. To make the simulation a little more realistic we next sample all three events from a Weibull distrubution of which the transition rates vary over time as long as the shape parameter is not equal to 1. We set the shape parameter of the three distributions to 1.5, and keep the rate parameters equal to .08, .08 and .06. After sampling we first estimate a model of which the transition rates are equal over time, without split points.

The density plot in Figure 4 shows that our model without split points fits poorly. Let's refit the model, now with split points at $t = 5$ and $t = 15$. At these places the empirical densities seem to change. The theoretical density based on the parameter estimates now resembles the empirical density more closely (Figure 5). Still we have some misfit in the beginning of the uninfected density. This time this is not completely due to the smoothing method, the density of uninfected discharge truly has its peak around five days. Here we stumble upon a limitation of the piecewise constant method. By assuming all parameters piecewise constant we cannot deal with ascending densities for uninfected discharge. The density for uninfected patients (equation (28)) is a decreasing function no matter what values λ_{12} and λ_{13} take on. Although we improve the model fit after $t = 5$ by allowing for a split points, we cannot get a proper fit for the time where the uninfected density is ascending.

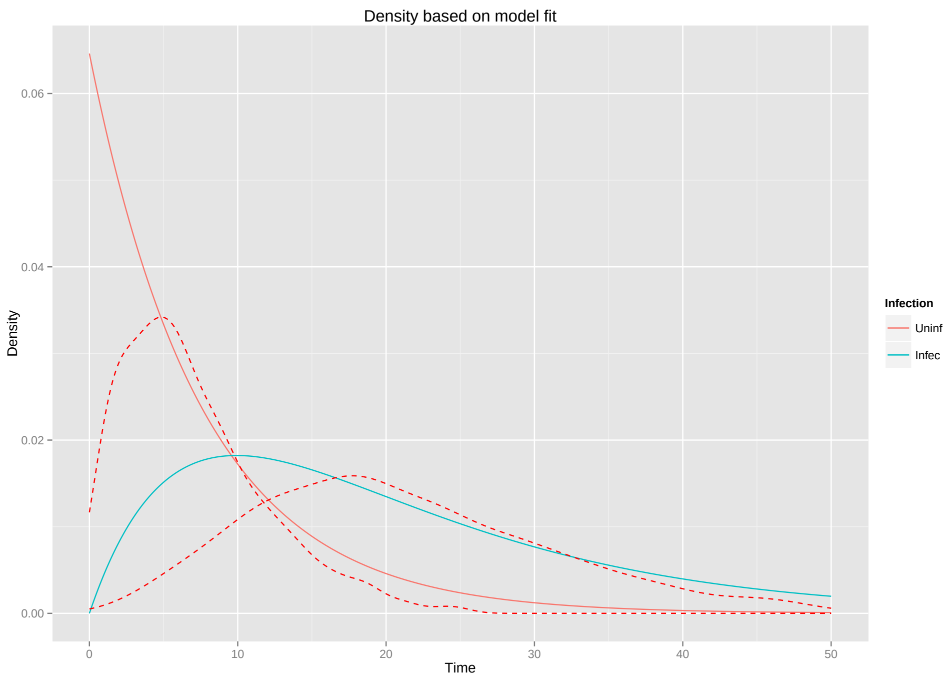

Figure 4: Diagnostics plot for the model with constant transition rates, infection and discharge sampled from a Weibull distribution.

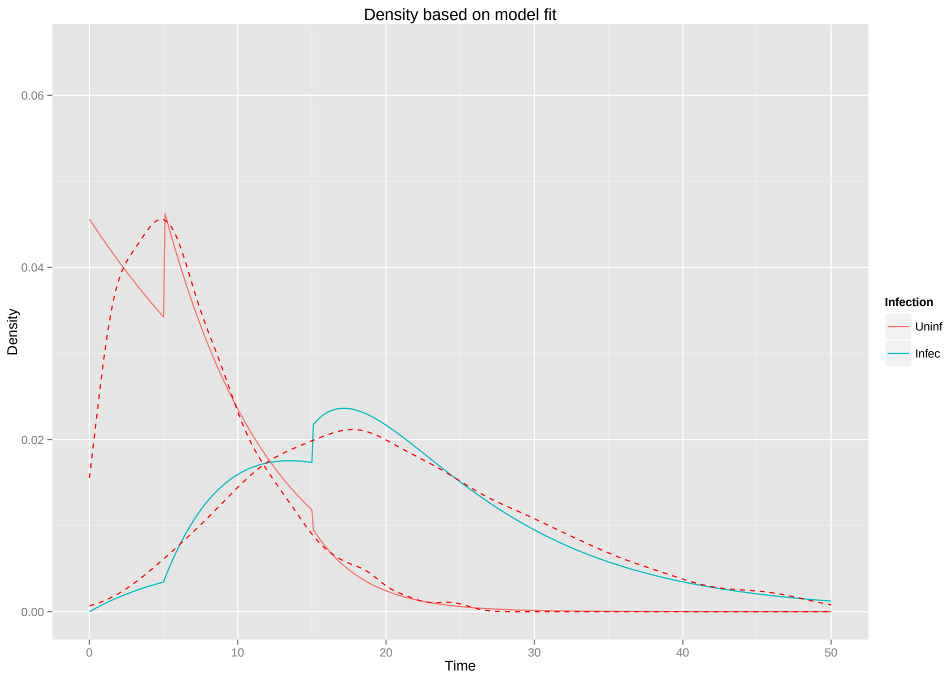

Figure 5: Diagnostics plot for the model with split points at 5 and 15 days. Infection and discharges sampled from a Weibull distribution.

4.2 Likelihood Ratio Test

We can test for the need of extra split points with a likelihood ratio test. The difference in likelihood of the larger model (with more intervals) and the smaller model follows a χ^2 distribution with the number of degrees of freedom equal to the number of parameters estimated extra in the larger model (thus three for each extra interval). A core assumption of the likelihood ratio test is that the compared models are nested. For piecewise constant models this is only true if the split points of the smaller model are also in the larger model. The larger model thus splits one or more intervals of the smaller model, but does not shift the split points of the smaller model. In the first of the two examples above the model without split points seems to have a fit that was about as good as the one with two split points. Performing a likelihood ratio test for the hypothesis *split points are redundant, a model with constant hazard rate suffices* with the function `LRT.icu`

```
> LRT.icu(no.phi, two.phi)
Stat        DF       P-value
 9.025       6        0.172
```

This confirms the assessment we made graphically, a model without split points is adequate to explain our data. Let's perform the same test on the models that were fitted on the data that were drawn from the Weibull distributions

```
> LRT.icu(wei.mod.no.phi, wei.mod.two.phi)
Stat        DF       P-value
 288.751     6        <2e-16
```

The p-value is very small, the model with two split points certainly has a better fit. We are very confident that the true tranisition rates are not equal over time.

4.3 Prediction from Model Fit

A medical doctor could be interested in the probabilities of being in each of the states after a certain number of days t. We can use the P-matrix introduced previously to obtain these probabilities. As mentioned this matrix only contained the survival probabilities, now we complete them with the distribution probabilities. These are the probabilities of having left the ICU either infected or uninfected by time point t.

Previously we have seen the definitions of the three conditional distribution probabilities in equation (11). First the probability of being discharged uninfected between u and t, given that the patient is still at the ICU and still uninfected at u is obtained from equations (7), (15) and (20)

$$P^1_{13}(u,t) = \frac{\int_u^t f^1_{13}(s)\mathrm{d}s}{S_1(u)} = \frac{\lambda_{13}}{\lambda_{1.}}\left[e^{-\lambda_{1.}(u)} - e^{-\lambda_{1.}(t)}\right]. \tag{43}$$

Similarly the probability that he is discharged infected by t, given that he was not infected at u is obtained from equation (31)

$$P^2_{13}(u,t) = \frac{\int_u^t f^2_{13}(s)\mathrm{d}s}{S_1(u)} = \frac{\lambda_{12}\lambda_{23}}{\lambda_{1.} - \lambda_{23}}\left[\frac{1 - e^{-\lambda_{23}(t-u)}}{\lambda_{23}} - \frac{1 - e^{-\lambda_{1.}(t-u)}}{\lambda_{1.}}\right]. \tag{44}$$

Finally from equation (32) we obtain the probability for a patient who is already infected at time point u to be discharged by time point t

$$P_{23}(u,t) = \frac{\int_u^t f_{23}(s)\mathrm{d}s}{S_2(u)} = 1 - e^{-\lambda_{23}(t-u)}. \tag{45}$$

Patients who are discharged by u (either infected or uninfected) cannot move from this state and are therefore in the same state at t. This makes the full P-matrix for the state at time point t, given the state at time point u

$$\mathbf{P(u,t)} = \begin{pmatrix} P_{11}(u,t) & P_{12}(u,t) & P^1_{13}(u,t) & P^2_{13}(u,t) \\ 0 & P_{22}(u,t) & 0 & P_{23}(u,t) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Again the rows represent the states at time point u and the columns the states at time point t. If we again define $\mathbf{P_j}$ as the P-matrix for interval j, it would be expanded to

$$\mathbf{P_j}(\phi^{\mathbf{b}}_{\mathbf{j}}, \phi^{\mathbf{e}}_{\mathbf{j}}) = \begin{pmatrix} P_{11}(\phi^b_j,\phi^e_j) & P_{12}(\phi^b_j,\phi^e_j) & P^1_{13}(\phi^b_j,\phi^e_j) & P^2_{13}(\phi^b_j,\phi^e_j) \\ 0 & P_{22}(\phi^b_j,\phi^e_j) & 0 & P_{23}(\phi^b_j,\phi^e_j) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

The subscript j denotes that the begin and end points of the conditional probabilities are in the same interval. To allow also for prediction for time points that do not coincide with the split points ϕ we allow for matrices in the interval of t to run untill t instead of to the next split point. The probabilities of being in one of the four states at time point t are found on the first row of matrix $\mathbf{P}(t)$, which is obtained by

$$\mathbf{P}(t) = \mathbf{P_1}(0,\phi^e_1) \dots \mathbf{P_{j-1}}(\phi^b_{j-1},\phi^e_{j-1})\mathbf{P_j}(\phi^b_j, t). \tag{46}$$

The above are the Chapman-Kolmogorov equations implemented in matrix notation, so all transition probabilities are calculated simultaneously.

To take the two previous examples with the data drawn from exponential and Weibull distributions we apply the best fitting models. The probabilities for a patient from the simulation study at which the events are drawn from the exponential distribution to be in each of the states after 15 days are

```
> predict.icu(no.phi, 15)
  ICU.Uninf      ICU.Inf Disch.Uninf   Disch.Inf
     0.1000       0.2511      0.4473      0.2016
```

And the state probabilities after three weeks for a patient in the simulation study at which the events were drawn from the Weibull distribution

```
> predict.icu(wei.mod.two.phi, 21)
  ICU.Uninf      ICU.Inf Disch.Uninf   Disch.Inf
     0.0349       0.1967      0.4814      0.2870
```

5 Simulation Study

A simulation study was done to see whether our parameter estimates are unbiased and to compare the standard errors of the parameters estimated from the three different data types. One hundred data sets of 1500 patients were generated with the following parameter values;

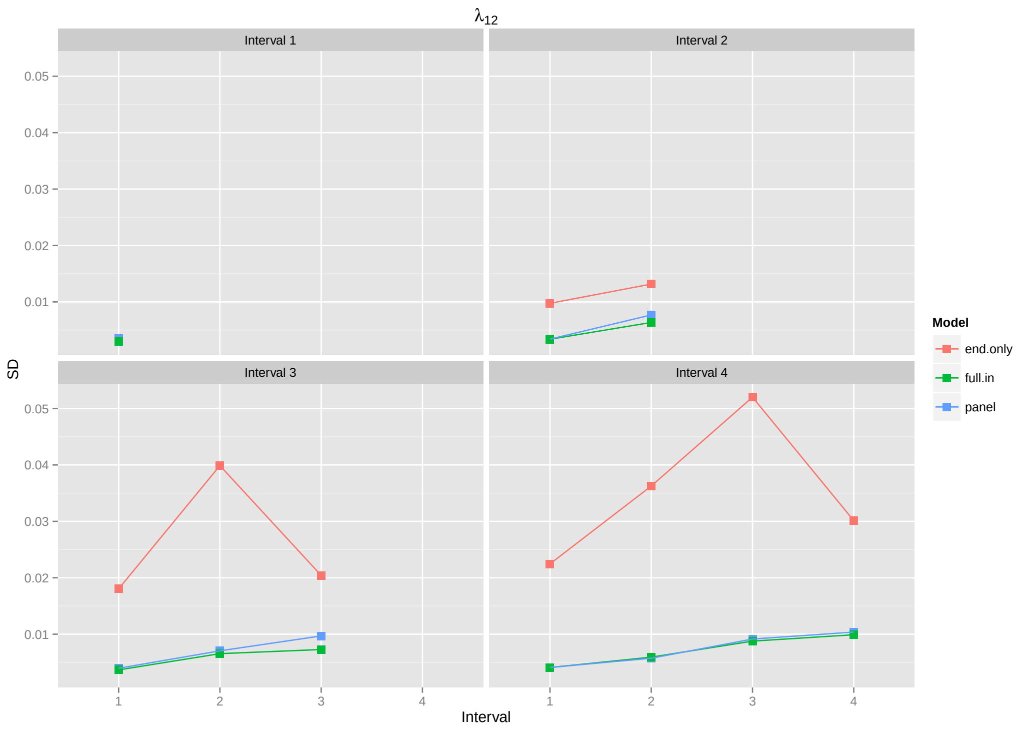

Figure 6: Standard deviations of the sampling distributions of λ_{12}, for the three different data types.

$\lambda_{12} = 0.08$, $\lambda_{13} = 0.08$, $\lambda_{23} = 0.06$. Estimation time of the parameters from panel and endpoint-only data did not allow for simulation with a larger number of data sets. The parameters were equal over the entire study time, making a model without piecewise constant parameters sufficient. Estimating piecewise constant models therefore should yield equal parameter estimates for each of the intervals. All models were applied with 1 interval (constant transition rates), 2 intervals (split at 10 days), 3 intervals (splits at 7 and 12 days), and 4 intervals (splits at 5, 10, and 15 days). We looked at the mean and median as well as the standard deviations of the distributions of the three parameters over the three data types, and at the ratio $\lambda_{13}/\lambda_{23}$. The mean and median of all sampling distributions were close to the true parameter value. Those parameters with a small difference between the mean and median of the estimates from the simulation and the true value, were those with a large standard error. Redoing the simulation a couple of times with a smaller number of data sets showed no systematic deviation from the true value in a particular direction. Therefore the deviations from the true parameter value are probably not systematic but are a result of relatively small simulation data sets combined with a large parameter standard error.

Figures 6 upto 9 show the standard deviations of the sampling distributions of the parameter estimates, as an indication of the parameter standard errors. Each figure contains four plots with the standard deviations of the sampling distributions of a model with no split points (top left), one split point (top right), two split points (bottom left), and three split points (bottom right). If we look at the top left panel of each of the four figures we see that the standard errors of the parameters are very similar for all three data types for the models without split points. (Note that without split points the results for the panel data and the endpoint-only data are exactly similar, and their points overlap.) This is no surprise since the additional information complete information data contains, is not used when estimating constant hazard rates.

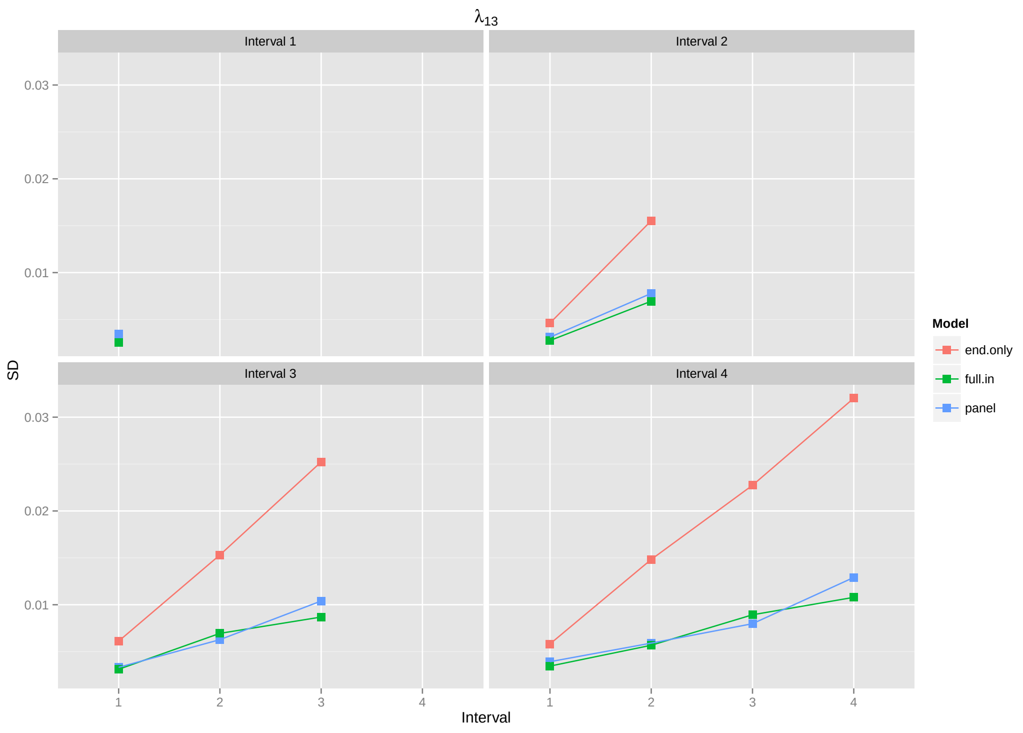

Figure 7: Standard deviations of the sampling distributions of λ_{13}, for the three different data types.

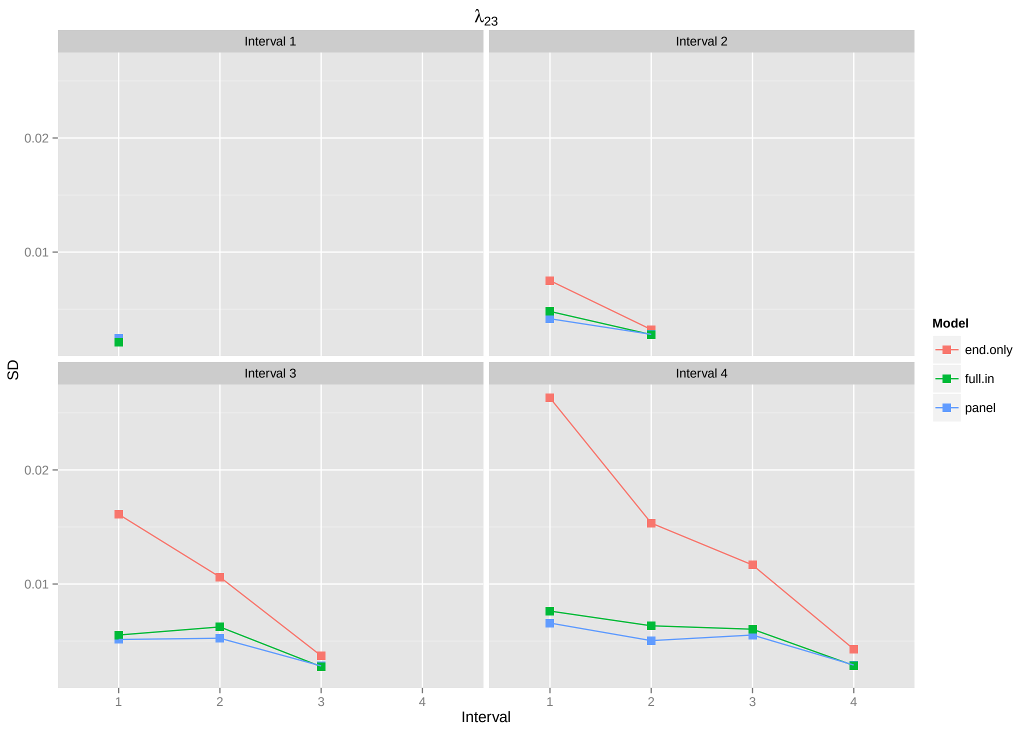

Figure 8: Standard deviations of the sampling distributions of λ_{23}, for the three different data types.

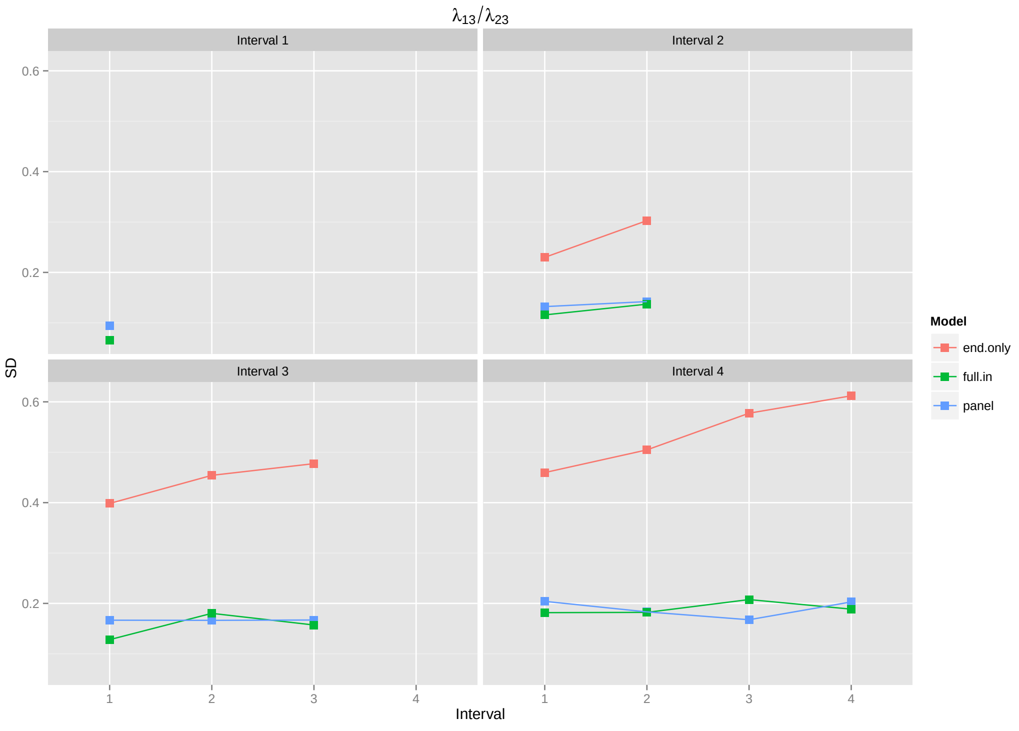

Figure 9: Standard deviations of the sampling distributions of the ratio $\lambda_{13}/\lambda_{23}$, for the three different data types.

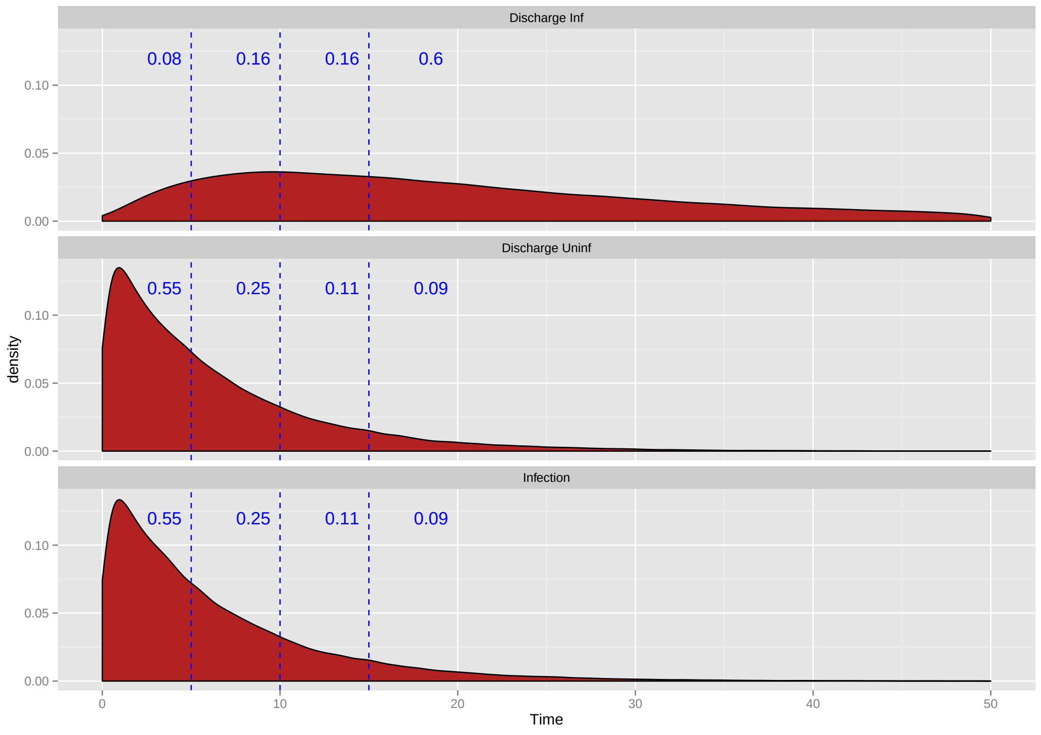

Figure 10: Density plots of the three events (discharge infected, discharge uninfected, infection) of all the 150,000 sampled patients. Vertical lines indicate the interval split points in the four interval situation. Numbers reflect the density in the given interval.

Table 1: Relative efficiency of the parameters estimated from panel and endpoint-only data, compared to the baseline parameters estimated from complete information data.

		λ_{12}	λ_{12}	λ_{13}	λ_{13}	λ_{23}	λ_{23}	ratio	ratio
nr.int	int	pan.	e.o.	pan.	e.o.	pan.	e.o.	pan.	e.o.
1	1	86	86	74	74	86	86	70	70
2	1	99	35	89	60	115	64	88	50
	2	83	48	89	45	99	86	96	45
3	1	93	20	94	51	108	34	77	32
	2	93	16	111	45	119	59	108	40
	3	75	36	84	34	98	74	94	33
4	1	100	18	88	59	116	29	89	40
	2	103	16	96	38	126	41	99	36
	3	96	17	112	39	109	52	124	36
	4	95	33	84	34	99	67	93	31

The standard errors of the piecewise constant model parameters show the same trend for all three parameters. First of all standard errors are larger for intervals that contain relatively little information of the given parameter for all three estimation methods. For illustration Figure 10 shows the density plots of the three events (discharge infected, discharge uninfected and infection) for all 150,000 patients. Uninfected patients are mainly discharged in the beginning of the study, leaving little patients to be discharged uninfected in the later intervals. The standard errors of λ_{13} (Figure 7) are therefore lowest for the first interval and rise with each interval. For λ_{23} this is the other way around, as infected patients are discharged primarily in the later intervals. Figure 8 shows dropping standard errors over the intervals for this parameter. The behavior of the standard errors of λ_{12} in Figure 6 shows a less straightforward picture. We would expect them to follow a similar pattern as λ_{13}, with most information in the first interval. This is indeed the case for the parameters estimated from the complete information and panel data, but the endpoint-only data show a clear peak of the standard errors in the middle intervals. Because in the endpoint-only situation λ_{12} is influenced by both discharge types (it is involved in all likelihood contributions), it performs the poorest in intervals with the least total discharges. Under these parameter and split point values, that is in the middle intervals.

Of even greater interest is the comparison of standard errors of the different data types. In addition to the visual display in the Figures 6 to 9, Table 1 shows the relative efficiencies of the estimates from panel and endpoint-only data. The standard deviation of the complete information estimates is divided by the standard deviation of the estimates of the other two data types and multiplied by a hundred. The relative efficiency is reversely related to the amount of data that must be collected in order to estimate a parameter with the same precision as is done from complete information data. For example if the relative efficiency of endpoint-only data for a certain parameter is 50%, we would need twice as many patients if we were collecting endpoint-only instead of complete information data to get the same amount of certainty about the parameter value. First of all we see confirmed that using complete information data to estimate piecewise constant models is a waste, because parameters estimated from panel data perform just as well. Panel data estimates tend to perform even slightly better for some of the parameters, which has to be due to simulation error. As expected the parameters estimated from endpoint-only data perform worse than the estimates from the other two data types. The most striking thing is the interaction between the amount of parameter information in the interval and the data type. For intervals with a lot of parameter information (the first interval for λ_{13}, the last interval for λ_{23}) the parameters estimated from the endpoint-only data perform almost as well as those estimated from the other two data types. However if the interval contains little parameter information the performance of the parameters from the endpoint-only data becomes very poor, whereas the estimates from the other two data types suffer much less.

Because of the long estimation times for panel and endpoint-only data we only scrutinized the behavior of the estimated parameters under one set of parameter values and one set of split points. Both theoretical explanations for the found results and smaller simulations with different parameter values and split point values indicate that the found results will generalize to other settings. A graphical display of the sampling distributions of the parameters estimated from the endpoint-only data can be found in the first section of appendix B.

6 Applying the Models to Real Data

Three data sets were available to us. The `icu.pneu` data set can be found in the `kmi` package in `R` (Allignol, 2011 [10]). Martin Bootsma kindly distributed two data sets that were used in his previous research. The `R` code used to perform the analysis of the `icu.pneu` data set can be found in appendix B.

6.1 Pneumonia

The `icu.pneu` data set is a random sample from a larger study that was conducted to assess the effects of the nosociomal infection pneumonia on the length of ICU stay. The data set contains censored observations, something that was not implemented in our software. Since this exercise is about the behavior of the methods on real data and not about the actual outcome of the analysis we treat the censored observations as observed discharges. The data type is complete information, so we can compare the results of the endpoint-only and the panel data estimators.

6.1.1 Data Inspection

One outlier of a patient staying 460 days at the ICU was removed because it would have a large influence on parameter estimation. The study contains 1312 uninfected and 108 infected patients. The mean discharge times are 12.6 days for the uninfected and 30.1 days for the infected patients. From the survival curves (Figure 11) and the histograms of discharge times (Figure 12) we also see clearly that an uninfected patient is expected to be discharged earlier than a patient that is infected during ICU stay. But does the acquiring of infection actually prolong ICU stay?

Table 2: Maximum likelihood estimates of the three parameters of the endpoint-only models without split points.

	mle	95.CI.low	95.CI.high
λ_{12}	0.00604	0.00500	0.00729
λ_{13}	0.07332	0.06934	0.07754
λ_{23}	0.05821	0.04616	0.07340

6.1.2 Endpoint-Only Estimation

We first estimate the parameters from the endpoint-only data thereby ignoring the information we have on transition times for moving from the uninfected to the infected state. We start

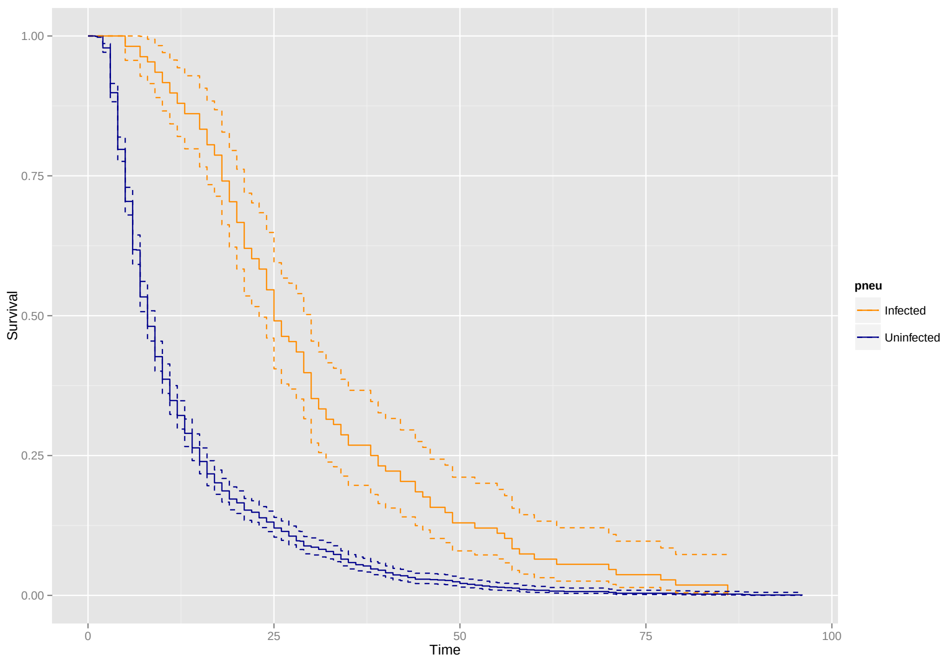

Figure 11: Kaplan-Meier plot of the survival times of infected and uninfected patients of the `kmi.pneu` data set. Plot produced with the `ggsurv` function, of which the source code can be found in appendix D.

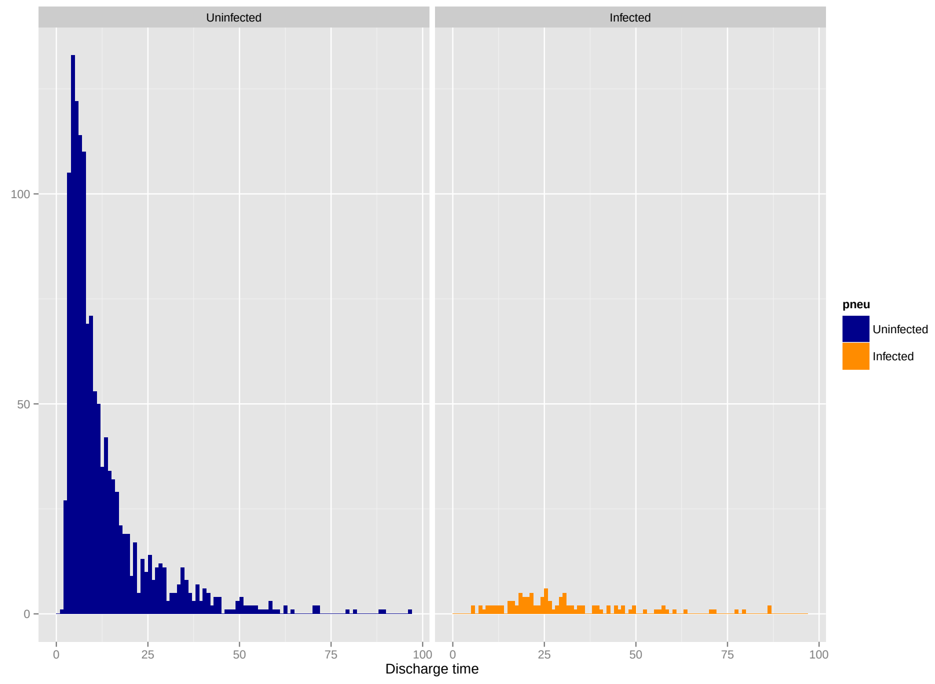

Figure 12: Histogram of the discharge times of infected and uninfected patients of the `kmi.pneu` data set.

with a model without split points and make a diagnostics plot to see how it fits to the data. The model estimates are shown in Table 2. The diagnostics plot in Figure 13 indicates poor

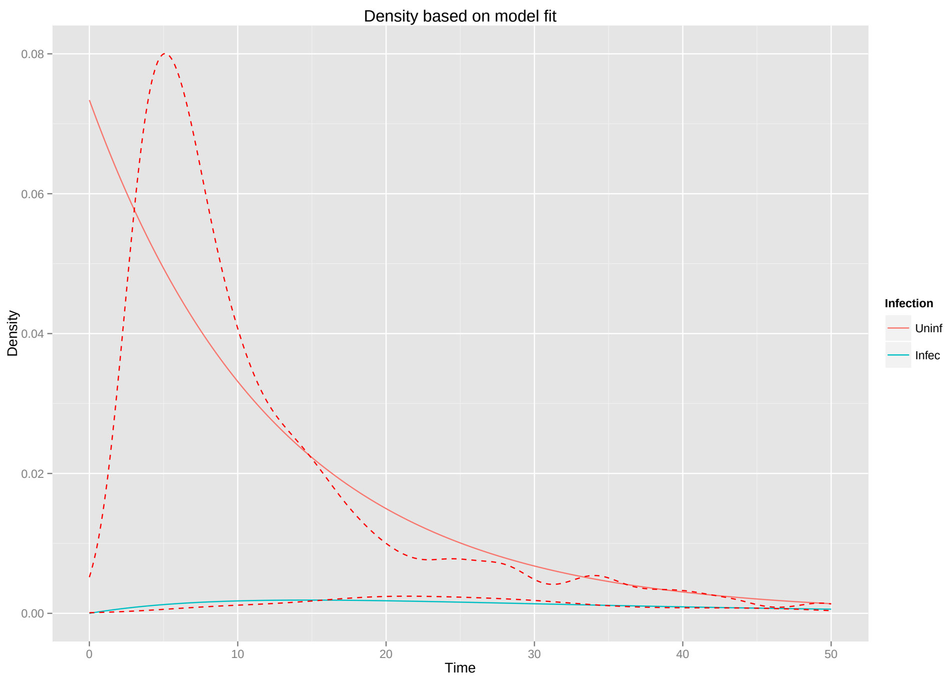

Figure 13: Comparing the empirical density to the density based on the model without split points.

model fit, especially for patients that left the ICU uninfected. From Figure 12 we see that the empirical density of the uninfected patients is truly ascendig, the ascending density at the first few days is not due to the kernel density estimator characteristics. Unfortunately we have seen in the previous section that our model cannot deal with this type of decreasing density for uninfected patients. The density plot suggests that we could improve our model fit if we would allow for split points at time points $t = 5$ and $t = 16$. The fit of this model was a lot better for the uninfected patients than the model without split points (estimates and diagnostics plot not shown). However it resulted in very wide confidence intervals, especially for the estimates of λ_{23}. From Figure 13 we see that the fit for the uninfected patients was already quite well and that estimating λ_{23} piecewise constant might not be necessary. In the function `icu.pwc` two options are implemented that allow for fixed parameters within the piecewise constant setting. Either λ_{23} can be fixed over the intervals, estimating one parameter instead of a seperate parameter for each interval, or λ_{23} is assumed to be related to λ_{13} through a fixed hazard ratio. This will make λ_{23} a function of λ_{13} for each interval. Both restrictions yield fewer estimated parameters and smaller confidence intervals for the estimated parameters.

We fit a model with split points at $t = 5$ and 16, and with λ_{23} being fixed over the intervals. First we compare this model to the model with the same split points, but with λ_{23} varying over the intervals. A likelihood ratio test confirms we have no need for varying λ_{23}'s over the intervals ($\chi^2 = 3.117$, on 2 d.f., $p = 0.2105$). The estimates of the model with fixed λ_{23} are shown in Table 6.1.2 and the density plot in Figure 14. We notice that for the uninfected patients this model fits the data a lot better than the model without split points that was fitted earlier. Although fixing λ_{23} over the intervals reduced the standard error of this parameter considerably, the confidence interval is still wide. There is still too much uncertainty about λ_{23} to make any claims about the lengthening effect of getting infected during ICU stay. The confidence interval of λ_{23} has overlap with the confidence intervals of all the λ_{13} parameters.

We also tried to estimate the parameters with λ_{13} and λ_{23} under the restriction that they are related by a fixed hazard ratio to each other. Unfortunately the estimation method estimates this hazard ratio to be 1 if some intervals contain very little information about a parameter. Therefore this restriction is not useful for data sets at which we have little discharges of one of the two groups in some of the intervals.

	mle	95.CI.low	95.CI.high
$\lambda_{12}^{(1)}$	0.00200	0.00077	0.00520
$\lambda_{13}^{(1)}$	0.05835	0.05282	0.06446
$\lambda_{12}^{(2)}$	0.00527	0.00201	0.01378
$\lambda_{13}^{(2)}$	0.09248	0.08478	0.10088
$\lambda_{12}^{(3)}$	0.01201	0.00703	0.02049
$\lambda_{13}^{(3)}$	0.05509	0.04629	0.06556
λ_{23}	0.10618	0.04079	0.27639

Table 3: Parameter estimates for the endpoint-only data with split points at 5 and 16 days and with λ_{23} fixed over the intervals.

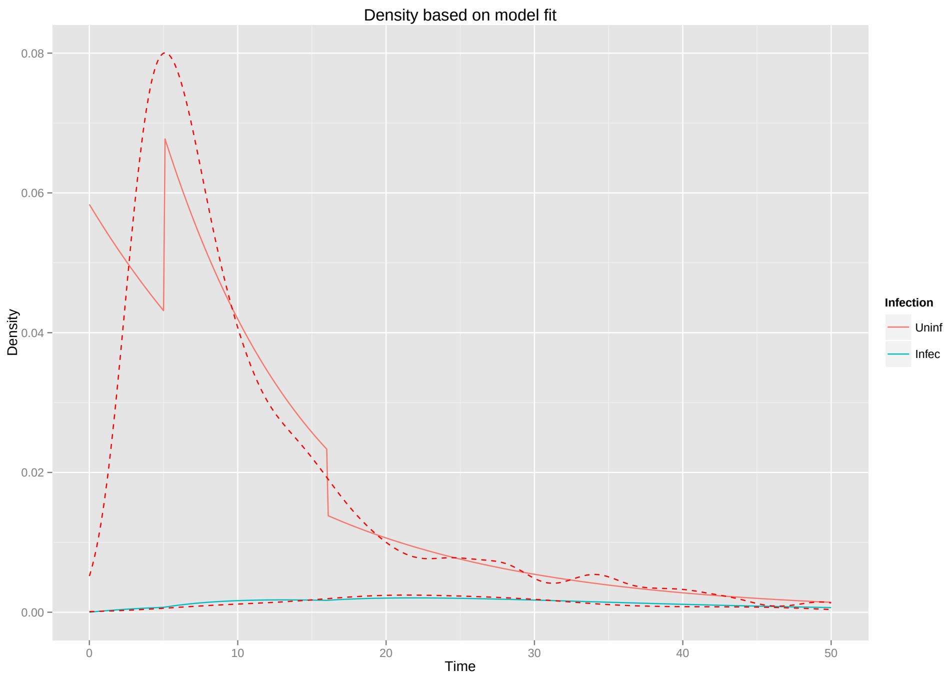

Figure 14: Comparing the empirical density to the density based on the model with split points at $t = 5$ and $t = 16$, with λ_{23} being fixed over the intervals.

6.1.3 Panel Data Estimation

We can compare the results of the endpoint-only data to results of models fitted on panel data. Again we apply split points at $t = 5$ and $t = 16$. Figure 15 shows that the model fit is very similar to the model fit when the parameters are estimated from endpoint-only data (Figure 14). Since the standard errors of the parameter estimates (Table 4) are much smaller for the panel data, these estimates do allow us to draw conclusions about the effect of the infection on ICU

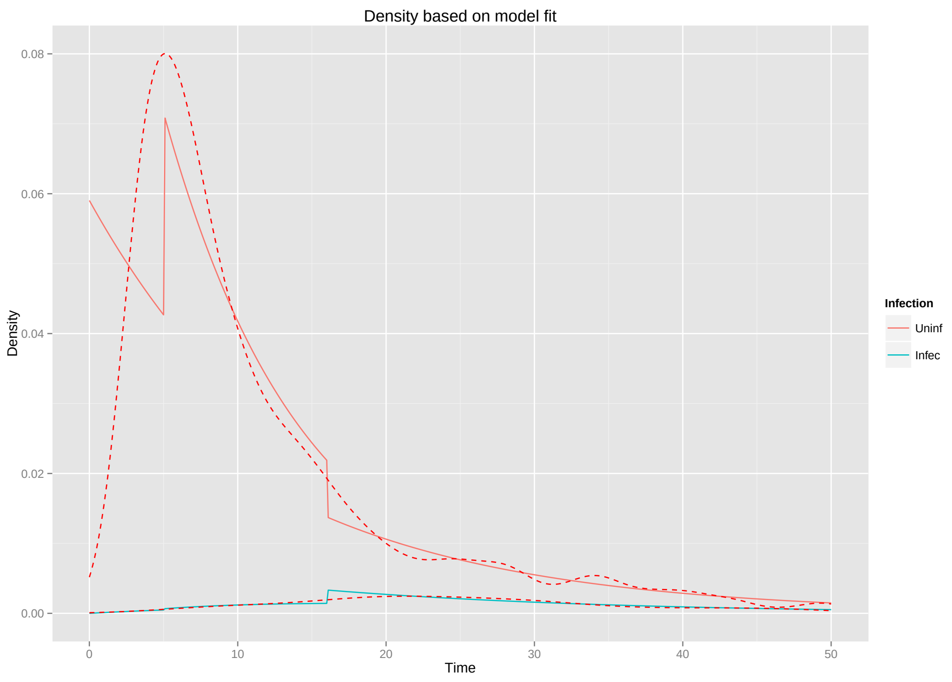

Figure 15: Density diagnostics plot for the model estimated on the panel data, with split points $\tau_1 = 5$ and $\tau_2 = 16$.

Table 4: The parameter estimates of the model based on the panel data.

	mle	CI.low	CI.high
$\lambda_{12}^{(1)}$	0.00578	0.00421	0.00794
$\lambda_{13}^{(1)}$	0.05903	0.05343	0.06520
$\lambda_{23}^{(1)}$	0.02030	0.00502	0.08210
$\lambda_{12}^{(2)}$	0.09900	0.09158	0.10701
$\lambda_{13}^{(2)}$	0.02779	0.01764	0.04376
$\lambda_{23}^{(2)}$	0.00306	0.00181	0.00517
$\lambda_{12}^{(3)}$	0.00306	0.00181	0.00517
$\lambda_{13}^{(3)}$	0.06229	0.05535	0.07011
$\lambda_{23}^{(3)}$	0.06381	0.05108	0.07971

stay. Between five and sixteen days of ICU stay patients that were infected with pneumonia were much less likely to be discharged than non-infected patients, $\widehat{\lambda_{23}^{(2)}/\lambda_{13}^{(2)}} = 0.281$ with 95% CI 0.020-0.409. In this time period patients that were infected were about three times less likely to be discharged than uninfected patients. Thus for those who get infected stay is lengthened, making early discharge unlikely. However this lengthening effect is not lasting, those who stay long have the same chance of being discharged, no matter whether they are infected or not.

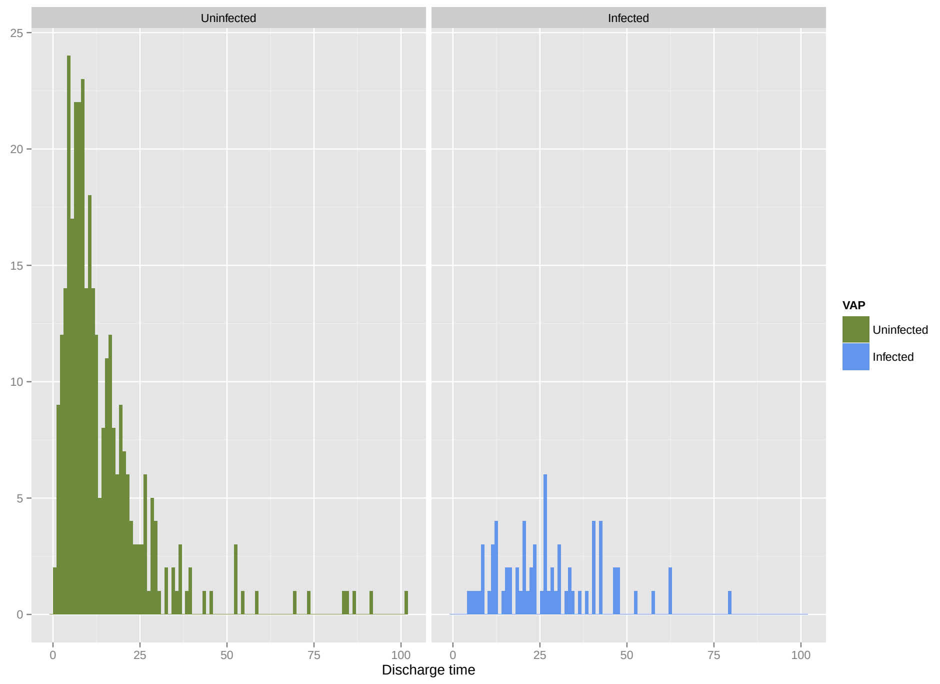

Figure 16: Histogram of the discharge times for uninfected and infected patients for the VAP study.

6.2 VAP

Next we look at two studies that were analyzed previously by Bootsma (2005). Also these studies looked at pneumonia, but this time the specific type ventilator-associated pneumonia.[11] Unfortunately the sizes of both studies are a lot smaller than the `kmi` study, making piecewise constant models dificult to estimate. One study contains as few as 21 infected discharges, assuring that standard errors will be large. Since both studies concern the exact same infection we combine them, creating a data set with 70 infected and 330 uninfected patients.

First we look at the distribution of the data. Figure 16 shows that the distribution of the discharge times is remarkably similar to the one of the `kmi.pneu` study (Figure 12). Next we will fit three models, with constant transition rates, with a split point at $\phi_1 = 6$, and with two split points at $\phi_1 = 6$ and $\phi_2 = 22$. The density diagnostics plot for the three models are shown in Figure 17.

We see in the upper plot that the empirical density of the infected patients is already matched well with constant transition rates. The fit for the uninfected patients is improved greatly by allowing for piecewise constant parameters. Likelihood ratio tests indicate that the last model with two split points is the best fitting ($\chi^2 = 9.47$, on 3 d.f., $p = 0.024$). However the standard errors got so large that we cannot draw any conclusions about the ratio of parameters. Especially the estimate of λ_{23} suffers. Standard errors could be reduced by fixing parameters either by fixing λ_{23} over the intervals, or by restricting λ_{23} to be proportional to λ_{13}. Unfortunately confidence intervals were still too wide to draw any conclusions about the lengthening effect of infection on ICU stay.

Both the `kmi.pneu` and the `VAP` study have shown there is a trade-off between model fit

[11]This is a subtype of general pneumonia, that can only be acquired when receiving mechanical ventilation in a hospital `http://en.wikipedia.org/wiki/Ventilator-associated_pneumonia`.

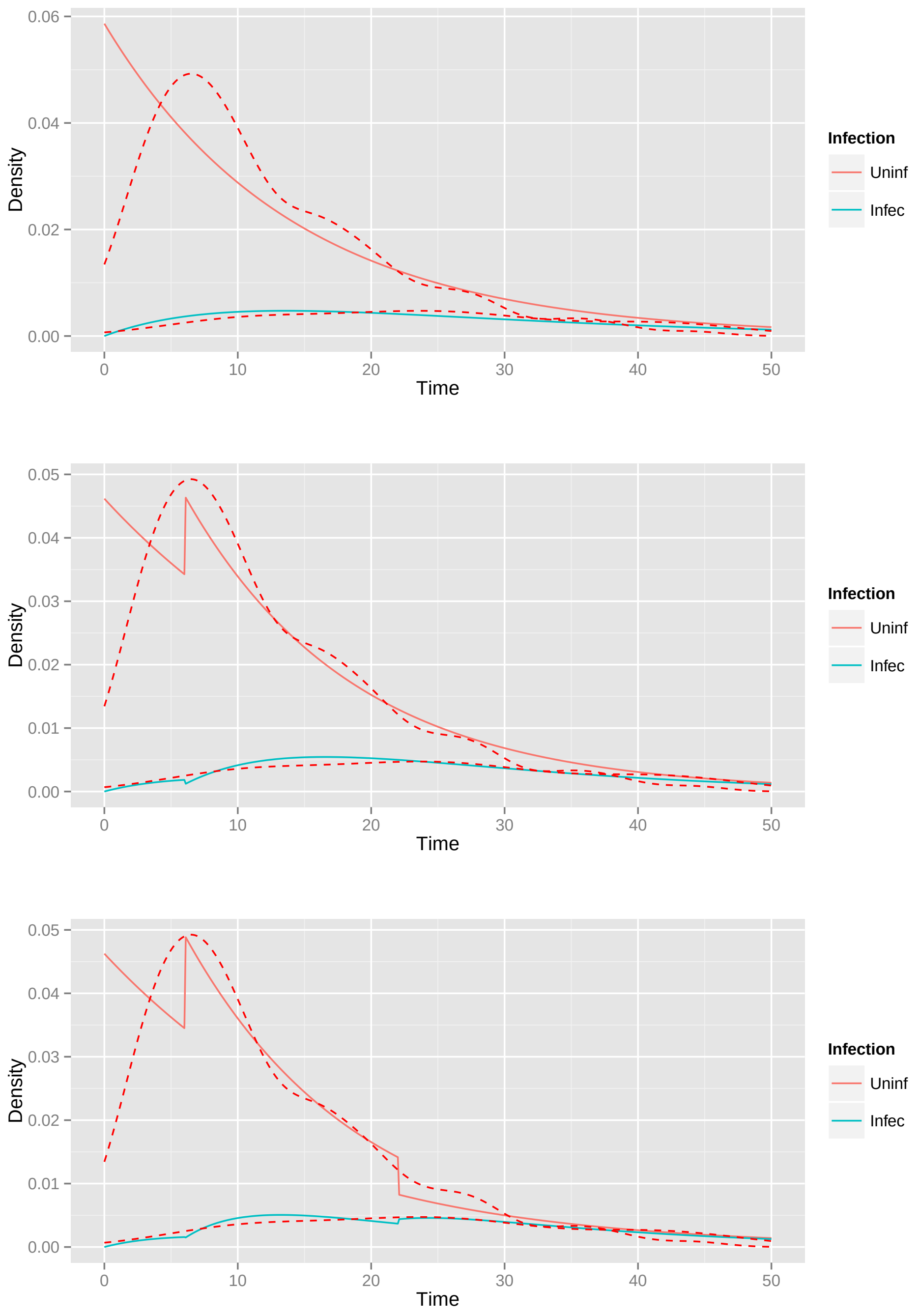

Figure 17: Diagnostics plots for the three models, with no split point (top), a split point at 6 days (middle), and split points at 6 and 22 days (bottom).

and the size of the standard errors. Estimating the parameters from endpoint-only data seems to put us in a Catch-22, preventing us from making claims about the ratio $\lambda_{13}/\lambda_{23}$. Either the model fits poorly and is clearly not a realistic representation of the data or the information is diluted by estimating multiple parameters such that large standard errors give so much uncertainty about the parameters that confidence intervals always overlap. We have tried to fight the large standard errors by putting restrictions on the model. This indeed resulted in confidence intervals that were narrower, but that unfortunately remained still too wide to draw any meaningful conclusions based on estimates from endpoint-only data.

7 Discussion

The three parameter illness-death model can answer questions about the prolonging effect of nosociomal infections on ICU stay. However the possibilities are very limited if transition intensities are fixed over time. The primary objective of this research was to extend the possibilities of estimating transition intensities from endpoint-only data. By allowing for piecewise constant transition intensities model fit can be improved considerably, as shown by diagnostics plots and likelihood ratio tests on both real life and simulated data.

7.1 Conclusion

From simulation studies we learned that a considerable price has to be paid for flexibility gained by estimating piecewise constant transition intensities from endpoint-only data. We only obtain information from this data type at the moment of patient discharge. For intervals with few relevant discharges for a given parameter standard errors get very large. Few interval discharges of uninfected patients (λ_{13}), infected patients (λ_{23}) or both (λ_{12}) results in a lot of uncertainty of the estimate for that interval. Restricting the model, either by fixing λ_{23} or the ratio of $\lambda_{13}/\lambda_{23}$ over the intervals, reduces standard errors and allows us to apply split points even when there are little or no discharges of one of the two groups at that time point. Application to real world data sets showed however that these restrictions don't reduce confidence intervals enough to draw meaningful conclusions. Especially about λ_{23} we have too much uncertainty, since data sets usually contain few infected patients.

Panel data seem a more promising direction for this line of research. Simulation shows that the standard errors of parameters estimated from this data type are only going up slightly when they have to be estimated from intervals with little relevant discharges. Parameters appeared to be equally efficiently estimated from panel and complete information data. In the form we studied panel data, the split points of the piecewise constant model completely coincided with the measuring of infection state at the patients still present at the ICU. With absolute certainty about the infection states at the split points these are the optimal panel data we can get as also shown by the relative efficiency of the parameter estimates. However just as with endpoint-only data, panel data can also allow for the infection states being unknown at the split points (Jackson, 2011 [5]). This would give researchers more flexibility when applying split points. Split points can be applied afterwards to improve model fit, which is not the case when split points have to coincide with measuring points. Another advantage is that patients can be measured at different time points, and even the number of measurements per patient can differ. The more information is collected on patients the more efficient the parameters will be estimated.

7.2 Suggestions for Further Research

A number of improvements to the piecewise constant models can be thought of. First of all we didn't foresee that observations could be censored at the end of the study because of the relatively short duration of ICU stay. The data set `kmi.pneu` showed however that censored observations can occur in this type of study, probably due to study termination while some patients were yet to be discharged. The models should therefor be expanded to allow for censoring.

A second point of interest is splitting up the discharge event into actual discharge and death at the ICU, thus adding a fourth state to the model. We did not make the distinction between death and discharge, both for simplicity and because there are relatively little death events in the datasets. Given the estimation problems that occur when optimizing the current piecewise constant model with endpoint-only data and the large parameter standard errors, splitting up the event and thereby diluting the information even further does not seem a sensible thing to do. However estimated from a more informative data type such as panel data, this model should provide useful results for reasonably large data sets. It is to be expected that infected patients will be discharged later, but die sooner than uninfected patients. If we let λ_{14} and λ_{24} be the transition rates to move to the death state for respectively uninfected and infected patients, we would expect $\lambda_{13}/\lambda_{23} \geq 1$, but $\lambda_{14}/\lambda_{24} \leq 1$. By considering the two states as one we might conclude that there is no effect of infection on LOS, because the patients who died instead of being discharged are pressing the parameter ratio to 1. Theoretically it is quite simple to extend the model with a death state, because all the likelihood results are similar to those for moving to state 3 in the current model.

The current model could also be further improved by allowing for covariates. This would enable us to see whether endogenous (sex, age) or exogenous (treatment, hospital of stay) patient characteristics influence the transition intensities and thereby the LOS of both infected and uninfected patients. Moreover the use of covariates might reduce variance in LOS due to causes other than being infected or not, thereby allowing to estimate the parameters with more certainty. In this light also the recent studies for the use of frailty models is of interest (Putter & van Houwelingen, 2013 [11]).

We already concluded that using panel data is more promising than endpoint-only data, because the standard errors are considerably smaller when the parameters are estimated from this data type. As mentioned we don't have to restrict ourselves to panel data at which the split points (ϕ) coincide with the universal measering points (τ). The model is also estimable from panel data at which patients are measured at different rates and days. We could even allow for a mix of panel and endpoint-only data from which the model is estimated. Expanding the current results to such a hybrid form seems the most promising way to turn from here.

References

[1] Reed D, Kemmerly SA. Infection control and prevention: A review of hospital-acquired infections and the economic implications. *The Oschner Journal,* 2009; **9(1)**:27-31.

[2] Redelmeier DA, Singh SM. Survival in academy award-winning actors and actresses. *Annals of Internal Medicine,* 2001; **134(10)**:955-962.

[3] Sylvestre MP, Huszti E, Hanley JA. Do OSCAR winners live longer than less successful peers? A reanalysis of the evidence. *Annals of Internal Medicine,* 2006; **145(5)**:361-363.

[4] de Wreede LC, Fiocco M, Putter H. mstate: An R package for the analysis of competing risks and multi-state models. *Journal of Statistical Software,* 2011; **38(7)**:1-30.

[5] Jackson CH. Multi-state models for panel data: The msm package for R. *Journal of Statistical Software,* 2011; **38(8)**:1-28.

[6] Bootsma MCJ. Mathematical studies of the dynamics of antibiotic resistance. Ph.D. thesis, 2005; Chapter 4; pages 97-124.

[7] Klein JP, Moeschberger ML. *Survival Analysis. Techniques for Censored and Truncated Data* (2nd edn). Statistics for Biology and Health. Springer: New York, 2003.

[8] Kaplan EL, Meier, P. Nonparametric estimation from incomplete observations. *Journal of the American Statistical Association,* 1958; **53**:457-481.

[9] Putter H, Fiocco M, Geskus RB. Tutorial in biostatistics: Competing risks and multi-state models. *Statistics in Medicine,* 2007; **26(11)**:2389-2430.

[10] Allignol A. kmi: Kaplan-Meier multiple imputation for the analysis of cumulative incidence functions in the competing risks setting. R package version 0.4.

[11] Putter H, van Houwelingen HC. Frailty multi-state models: Are they identifiable? Do we need them? 2013; *in print.*

A Source Code of Functions Used

This appendix contains the source code of all the relevant functions that were written in this project.

A.1 Data Generators

The first function creates endpoint-only data with equal transition intensities. Its arguments are the size of simulated study, and the values of the three parameters

```
cre.dat <- function(n, lam12, lam13, lam23){
  LOS.U <- rexp(n, lam13)
  TimeInf <- rexp(n, lam12)
  LOS.I <- rexp(n, lam23)

  data.frame(Discharge = ifelse(LOS.U < TimeInf, LOS.U, LOS.I + TimeInf),
             Infection = ifelse(LOS.U < TimeInf, 0, 1))
}
```

A second function creates the same data, but now complete information. Same arguments as previous function.

```
cre.dat.fi <- function(n, lam12, lam13, lam23){
  LOS.U <- rexp(n, lam13)
  TimeInf <- rexp(n, lam12)
  LOS.I <- rexp(n, lam23)

  ret.dat <- data.frame(Discharge = (ifelse(LOS.U < TimeInf, LOS.U, LOS.I + TimeInf)),
             Infection = (ifelse(LOS.U < TimeInf, 0, TimeInf)))
  ret.dat$Inf.Ind <- ifelse(ret.dat$Infection == 0, 0, 1)
  ret.dat
}
```

The last generator converts the complete information data into panel data. Its arguments are a data frame with panel data and a vector with split points.

```
make.panel.dat <- function(dat, tau){
  dis <- dat[,1]; inf <- dat[,2]
  ret.dat <- matrix(0, ncol = length(tau), nrow = nrow(dat))

  for(i in 1:ncol(ret.dat)){
    ret.dat[,i] <- ifelse(dis <= tau[i], 0,
                          ifelse(inf <= tau[i], 1, 0))
  }
  colnames(ret.dat) <- paste('tau',1:length(tau), sep = '')
  data.frame(ret.dat,
             Discharge = dat[,1],
             Infection = dat[,3])
}
```

A.2 Estimation Functions

The function `icu.pwc` estimates the parameters from endpoint-only data. Its arguments; a data frame with the variables discharge time and infection, a string with the name of the time variable, a string with the name of the infection variable, a string with the name of the level indicating a patient is infected, a vector with split points, the width of the confidence interval, the initial parameter values of for the internally used `optim` function, and an option to fix parameters (either "l23" of "prop.haz"). Function will return an object of class `icu.mod`, containing the parameters estimates and standard errors, the value of the loglikelihood, the result from the `optim` function, the vector with split points, and the original time and infection variables.

```
icu.pwc <- function(data,
                    time,
                    inf,
                    inf.level,
                    phi='none',
                    CI = .95,
                    init = 'random',
                    fix.par = "no"){

  nr.int <- if(phi[1] == 'none') 1 else length(phi) + 1
  time <- data[ ,colnames(data) %in% time]
  inf <- data[ ,colnames(data) %in% inf]
  stopifnot(class(time) == 'numeric')

  nr.par <- if(phi[1] == 'none') {
    3
  } else if(identical(fix.par, "no")){
    nr.int * 3
  } else {
    nr.int * 2 + 1
  }

  gammai1 <- if(nr.int > 0) {as.numeric(time <= phi[1])}
  gammai2 <- if(nr.int > 2) {as.numeric(time <= phi[2] & time > phi[1])
  } else if(nr.int == 2) {as.numeric(time > phi[1])}
  gammai3 <- if(nr.int > 3) {as.numeric(time <= phi[3] & time > phi[2])
  } else if(nr.int == 3) {as.numeric(time > phi[2])}
  gammai4 <- if(nr.int > 3) {as.numeric(time > phi[3])}
  psii <- as.numeric(inf == inf.level)
  phi1 <- phi[1]; phi2 <- phi[2]; phi3 <- phi[3]

  P11_1 <- function(t, l1._1) exp(-l1._1*t)
  P11_2 <- function(t, l1._2) exp(-l1._2*t)
  P11_3 <- function(t, l1._3) exp(-l1._3*t)
  P12_1 <- function(t, l12_1, l1._1, l23_1){
    (l12_1/(l1._1 - l23_1)*(exp(-l23_1*t) - exp(-l1._1*t)))}
  P12_2 <- function(t, l12_2, l1._2, l23_2){
    (l12_2/(l1._2 - l23_2)*(exp(-l23_2*t) - exp(-l1._2*t)))}
```

```
P12_3 <- function(t, l12_3, l1._3, l23_3){
  (l12_3/(l1._3 - l23_3)*(exp(-l23_3*t) - exp(-l1._3*t)))}
P22_2 <- function(t, l23_2) exp(-l23_2*t)
P22_3 <- function(t, l23_3) exp(-l23_3*t)
f131_1 <- function(t, l13_1, l1._1) l13_1 * exp(-l1._1*t)
f131_2 <- function(t, l13_2, l1._2) l13_2 * exp(-l1._2*t)
f131_3 <- function(t, l13_3, l1._3) l13_3 * exp(-l1._3*t)
f131_4 <- function(t, l13_4, l1._4) l13_4 * exp(-l1._4*t)
f132_1 <- function(t, l12_1, l23_1, l1._1){
  ((l12_1*l23_1)/(l1._1 - l23_1)*(exp(-l23_1*t) - exp(-l1._1*t)))}
f132_2 <- function(t, l12_2, l23_2, l1._2){
  ((l12_2*l23_2)/(l1._2 - l23_2)*(exp(-l23_2*t) - exp(-l1._2*t)))}
f132_3 <- function(t, l12_3, l23_3, l1._3){
  ((l12_3*l23_3)/(l1._3 - l23_3)*(exp(-l23_3*t) - exp(-l1._3*t)))}
f132_4 <- function(t, l12_4, l23_4, l1._4){
  ((l12_4*l23_4)/(l1._4 - l23_4)*(exp(-l23_4*t) - exp(-l1._4*t)))}
f23_2 <- function(t, l23_2) l23_2 * exp(-l23_2*t)
f23_3 <- function(t, l23_3) l23_3 * exp(-l23_3*t)
f23_4 <- function(t, l23_4) l23_4 * exp(-l23_4*t)

LL1int <- function(par, t, pi){
  l12_1 <- exp(par[1]); l13_1 <- exp(par[2]); l23_1 <- exp(par[3])
  l1._1 <- exp(par[1]) + exp(par[2])
  sum(log(
    f131_1(t, l13_1, l1._1)^(1-pi)*
      f132_2(t, l12_1, l23_1, l1._1)^pi ) ) }

LL2int <- function(par, t, phi, gi1, pi){
  if(identical(fix.par, "no")){
    l12_1 <- exp(par[1]); l13_1 <- exp(par[2]); l23_1 <- exp(par[3])
    l1._1 <- exp(par[1]) + exp(par[2]); l12_2 <- exp(par[4]); l13_2 <- exp(par[5])
    l23_2 <- exp(par[6]); l1._2 <- exp(par[4]) + exp(par[5])
  } else if (identical(fix.par, 'l23')){
    l12_1 <- exp(par[1]); l13_1 <- exp(par[2]); l23_1 <- exp(par[5])
    l1._1 <- exp(par[1]) + exp(par[2]); l12_2 <- exp(par[3]); l13_2 <- exp(par[4])
    l23_2 <- exp(par[5]); l1._2 <- exp(par[3]) + exp(par[4])
  } else {
    l12_1 <- exp(par[1]); l13_1 <- exp(par[2]); l23_1 <- exp(par[5]) * exp(par[2])
    l1._1 <- exp(par[1]) + exp(par[2]); l12_2 <- exp(par[3]); l13_2 <- exp(par[4])
    l23_2 <- exp(par[5]) * exp(par[4]); l1._2 <- exp(par[3]) + exp(par[4])
  }

  sum(log(
    (P11_1 (phi, l1._1)*f131_2(t-phi, l13_2, l1._2))^((1-gi1)*(1-pi))*
      (P11_1(phi, l1._1)*f132_2(t-phi, l12_2, l23_2, l1._2) +
         P12_1(phi, l12_1, l1._1, l23_1) * f23_2(t-phi, l23_2))^((1-gi1)*pi)*
      f131_1(t, l13_1, l1._1)^(gi1*(1-pi)) *
      f132_1(t, l12_1, l23_1, l1._1)^(gi1*pi) ) ) }
```

```
LL3int <- function(par, t, phi1, phi2, gi1, gi2, gi3, pi){
  if(identical(fix.par, "no")){
    l12_1 <- exp(par[1]); l13_1 <- exp(par[2]); l23_1 <- exp(par[3])
    l1._1 <- exp(par[1]) + exp(par[2]); l12_2 <- exp(par[4]); l13_2 <- exp(par[5])
    l23_2 <- exp(par[6]); l1._2 <- exp(par[4]) + exp(par[5]); l12_3 <- exp(par[7])
    l13_3 <- exp(par[8]); l23_3 <- exp(par[9]); l1._3 <- exp(par[7]) + exp(par[8])
  } else if (identical(fix.par, 'l23')){
    l12_1 <- exp(par[1]); l13_1 <- exp(par[2]); l23_1 <- exp(par[7])
    l1._1 <- exp(par[1]) + exp(par[2]); l12_2 <- exp(par[3]); l13_2 <- exp(par[4])
    l23_2 <- exp(par[7]); l1._2 <- exp(par[3]) + exp(par[4]); l12_3 <- exp(par[5])
    l13_3 <- exp(par[6]); l23_3 <- exp(par[7]); l1._3 <- exp(par[5]) + exp(par[6])
  } else {
    l12_1 <- exp(par[1]); l13_1 <- exp(par[2]); l23_1 <- exp(par[7]) * exp(par[2])
    l1._1 <- exp(par[1]) + exp(par[2]); l12_2 <- exp(par[3]); l13_2 <- exp(par[4])
    l23_2 <- exp(par[7]) * exp(par[4]); l1._2 <- exp(par[3]) + exp(par[4])
    l12_3 <- exp(par[5]); l13_3 <- exp(par[6]); l23_3 <- exp(par[7]) * exp(par[6])
    l1._3 <- exp(par[5]) + exp(par[6])
  }

  sum(log(
    (f131_1(t, l13_1, l1._1)^(1-pi) *
       f132_1(t, l12_1, l23_1, l1._1)^pi)^gi1*

      ((P11_1(phi1, l1._1)*f131_2(t-phi1, l13_2, l1._2)) ^ (1-pi) *
         (P11_1(phi1, l1._1)*f132_2(t-phi1, l12_2, l23_2, l1._2) +
            P12_1(phi1, l12_1, l1._1, l23_1)* f23_2(t - phi1, l23_2))^pi )^gi2 *

      ((P11_1(phi1, l1._1)*P11_2(phi2-phi1, l1._2)*
          f131_3(t-phi2, l13_3, l1._3))^(1-pi) *

         (P11_1(phi1, l1._1)*P11_2(phi2-phi1, l1._2)*
            f132_3(t-phi2, l12_3, l23_3, l1._3)+
            P11_1(phi1, l1._1)*P12_2(phi2-phi1, l12_2, l1._2, l23_2)*
            f23_3(t-phi2, l23_3) +
            P12_1(phi1, l12_1, l1._1, l23_1)*P22_2(phi2-phi1, l23_2)*
            f23_3(t-phi2,l23_3))^pi)^gi3
  ))}

LL4int <- function(par, t, phi1, phi2, phi3, gi1, gi2, gi3, gi4, pi){
  if(identical(fix.par, "no")){
    l12_1 <- exp(par[1]); l13_1 <- exp(par[2]); l23_1 <- exp(par[3])
    l1._1 <- exp(par[1]) + exp(par[2]); l12_2 <- exp(par[4]); l13_2 <- exp(par[5])
    l23_2 <- exp(par[6]); l1._2 <- exp(par[4]) + exp(par[5]); l12_3 <- exp(par[7])
    l13_3 <- exp(par[8]); l23_3 <- exp(par[9]); l1._3 <- exp(par[7]) + exp(par[8])
    l12_4 <- exp(par[10]); l13_4 <- exp(par[11]); l23_4 <- exp(par[12]);
    l1._4 <- exp(par[10]) + exp(par[11])
  } else if(identical(fix.par, 'l23')){
    l12_1 <- exp(par[1]); l13_1 <- exp(par[2]); l23_1 <- exp(par[9])
    l1._1 <- exp(par[1]) + exp(par[2]); l12_2 <- exp(par[3]); l13_2 <- exp(par[4])
    l23_2 <- exp(par[9]); l1._2 <- exp(par[3]) + exp(par[4]); l12_3 <- exp(par[5])
    l13_3 <- exp(par[6]); l23_3 <- exp(par[9]); l1._3 <- exp(par[5]) + exp(par[6])
    l12_4 <- exp(par[7]); l13_4 <- exp(par[8]); l23_4 <- exp(par[9]);
    l1._4 <- exp(par[7]) + exp(par[8])
  } else {
    l12_1 <- exp(par[1]); l13_1 <- exp(par[2]); l23_1 <- exp(par[9]) * exp(par[2])
    l1._1 <- exp(par[1]) + exp(par[2]); l12_2 <- exp(par[3]); l13_2 <- exp(par[4])
    l23_2 <- exp(par[9]) * exp(par[4]); l1._2 <- exp(par[3]) + exp(par[4])
    l12_3 <- exp(par[5]); l13_3 <- exp(par[6]); l23_3 <- exp(par[9]) * exp(par[6])
    l1._3 <- exp(par[5]) + exp(par[6]); l12_4 <- exp(par[7]); l13_4 <- exp(par[8])
    l23_4 <- exp(par[9]) * exp(par[8]); l1._4 <- exp(par[7]) + exp(par[8])
  }

  sum(log(
    (f131_1(t, l13_1, l1._1)^(1-pi) *
       f132_1(t, l12_1, l23_1, l1._1)^pi)^gi1*

      ((P11_1(phi1, l1._1)*f131_2(t-phi1, l13_2, l1._2)) ^ (1-pi) *
         (P11_1(phi1, l1._1)*f132_2(t-phi1, l12_2, l23_2, l1._2) +
            P12_1(phi1, l12_1, l1._1, l23_1)* f23_2(t - phi1, l23_2))^pi )^gi2 *

      ((P11_1(phi1, l1._1)*P11_2(phi2-phi1, l1._2)*
          f131_3(t-phi2, l13_3, l1._3))^(1-pi) *

         (P11_1(phi1, l1._1)*P11_2(phi2-phi1, l1._2)*
            f132_3(t-phi2, l12_3, l23_3, l1._3)+
            P11_1(phi1, l1._1)*P12_2(phi2-phi1, l12_2, l1._2, l23_2)*
            f23_3(t-phi2, l23_3) +
            P12_1(phi1, l12_1, l1._1, l23_1)*P22_2(phi2-phi1, l23_2)*
            f23_3(t-phi2,l23_3))^pi)^gi3*

      ( (P11_1(phi1, l1._1)*P11_2(phi2-phi1, l1._2)*
           P11_3(phi3-phi2,l1._3)*f131_4(t-phi3, l13_4, l1._4))^(1-pi)*

          (P11_1(phi1, l1._1)*P11_2(phi2-phi1, l1._2)*
             P11_3(phi3-phi2,l1._3)*f132_4(t-phi3, l12_4, l23_4, l1._4)+
             P11_1(phi1, l1._1)*P11_2(phi2-phi1, l1._2)*
             P12_3(phi3-phi2, l12_3, l1._3, l23_3)*f23_4(t-phi3, l23_4)+
             P11_1(phi1, l1._1)*P12_2(phi2-phi1, l12_2, l1._2, l23_2)*
             P22_3(phi3-phi2, l23_3)*f23_4(t-phi3, l23_4)+
             P12_1(phi1, l12_1, l1._1, l23_1)*P22_2(phi2-phi1, l23_2)*
             P22_3(phi3-phi2, l23_3)*f23_4(t-phi3, l23_4))^pi)^gi4
  ))}

init <- if (init[1] == 'random') {runif(nr.par, 0, .2)
} else {init}
init <- log(init)

opt.res <- if(nr.int == 1) {
    optim(init, LL1int,, t = time, pi = psii,
          hessian = T, method = 'L-BFGS-B', lower = log(.002), upper = 0,
          control = list(fnscale=-1, maxit = 5000))
  } else if(nr.int == 2){
    optim(init, LL2int, ,phi = phi1, gi1 = gammai1, t = time, pi = psii,
          hessian = T,  method = 'L-BFGS-B',lower = log(.002), upper = 0,
          control = list(fnscale=-1, maxit = 5000))
  } else if(nr.int == 3){
    optim(init, LL3int, , time, phi1, phi2, gammai1, gammai2,
          gammai3, psii, hessian=T, method = 'L-BFGS-B', lower = log(.002), upper = 0,
          control = list(fnscale=-1, maxit=5000))
  } else {optim(init, LL4int, , time, phi1, phi2, phi3, gammai1,
                gammai2, gammai3, gammai4,  psii, hessian = T,
                method = 'L-BFGS-B', lower = log(.002), upper = 0,
                control = list(fnscale=-1, maxit=5000))}

  se <- suppressWarnings(sqrt(diag(solve(opt.res$hessian*-1))))
  quant <- qnorm(1-(1-CI)/2)
  CI.low <- opt.res$par - se*quant; CI.high <- opt.res$par + se*quant
  est <- round(cbind(mle = exp(opt.res$par), CI.low = exp(CI.low),
                     CI.high = exp(CI.high)), 5)

  rownames(est) <- if (identical(fix.par, "no")){
    paste(c('l12', 'l13', 'l23'), rep(1:nr.int, each =3), sep = '.')
  } else if(identical(fix.par, 'prop.haz')) {
    c(paste(c("l12", "l13"), rep(1:nr.int, each = 2), sep = "."), "HRl23/l13")
  } else {
    c(paste(c("l12", "l13"), rep(1:nr.int, each = 2), sep = "."), "l23")
  }

  colnames(est)[2:3] <- paste(CI*100, colnames(est)[2:3], sep = '.')

  LL <- opt.res$value

  ret <- list(est = est, LL = LL, opt.res = opt.res, phi = phi, time = time,
              infection = inf, fix.par = fix.par)
  class(ret) <- 'icu.mod'
  ret
}

print.icu.mod <- function(obj) print(obj$est)
```

The function `fix.par.fix` reorders the estimats of a `icu.mod` object of which parameters were fixed. The estimates of λ_{23} are placed at the right positions, either being the same for each interval (if λ_{23} was fixed) or calculated from the estimate of λ_{13} for the interval and het fixed ratio. This function should be applied before other functions, such as the diagnostics plots, are applied to the output.

```
fix.par.fix <- function(icu.mod){
  if(identical(icu.mod$fix.par, "l23")){
    weave.l23 <- function(vec){
      l23 <- tail(vec, 1); vec <- vec[-length(vec)]
      ind <- seq(2, 11, 3)[1:(length(vec)/2)]
      for(i in seq_along(ind)) vec <- append(vec, l23, ind[i])
      vec
    }
    icu.mod$est <- apply(icu.mod$est, 2, weave.l23)

  } else if(identical(icu.mod$fix.par, "prop.haz")){

    im <- icu.mod$est
    hr <- im[nrow(im), ]
    im <- im[-nrow(im), ]
    lam13 <- im[1:nrow(im) %% 2 == 0, 1]
    l23 <- t(apply(as.matrix(lam13), 1, function(x) x * hr))

    ind <- seq(2, 11, 3)[1:nrow(l23)]
    for(i in seq_along(ind)) {
      im <-  rbind(im[1:ind[i], ], l23[i,], im[-(1:ind[i]) , ])
    }
    rownames(im)[ind+1] <- paste("l23", 1:nrow(l23), sep = ".")
    icu.mod$est <- round(im, 5)
  }
  return(icu.mod)
}
```

Function to estimate the parameters from panel data, its arguments match those of the `icu.pwc` function. Returned object contains the parameters estimates and standard errors and the value of the loglikelihood. Returns only the parameter estimates.

```
icu.pwc.pan <- function(data,
                        time,
                        inf,
                        inf.level,
                        tau='none',
                        init = 'random'){

  nr.int <- if(tau[1] == 'none') 1 else length(tau) + 1
  time <- data[ ,colnames(data) %in% time]
  inf <- data[ ,colnames(data) %in% inf]
  inf <- as.numeric(inf == inf.level)
  stopifnot(class(time) == 'numeric')

  bounds <- c(0, tau, max(time))

  # matrix with the gamma_ij
  gmat <- matrix(0, ncol = nr.int, nrow = length(time))
  for (i in 1:nr.int){
    gmat[,i] <- ifelse(time <= bounds[i], 0,
                       ifelse(time > bounds[i+1], 0 , 1))
  }

  # matrix with the delta_ij
  dmat <- (cbind(begin = 0, data[,1:3], last = 0))
  for(i in 1:nr.int){
    dmat[,i+1] <- ifelse(inf == 0, 0,
                         ifelse(dmat[i+1] == 1, 1,
                                ifelse(time > bounds[i] & time <= bounds[i+1], 1, 0)))
  }

  # matrix with the t_ij
  tmat <- mapply(function(tau1, tau2, ti) {ifelse(ti <= tau1, 0,
                          ifelse(ti <= tau2, ti-tau1, tau2-tau1))},
                 tau1 = bounds[1:length(bounds)-1],
                 tau2 = bounds[2:length(bounds)],
                 MoreArgs = list(ti = time))

  lik <- function(par, del.b, del.e, g, t){
    l12 <- par[1]; l13 <- par[2]; l23 <- par[3]
    l1. <- sum(par[1:2])
    -1*(sum(log(
      ((l13^g * exp(-l1.*t))^((1-del.b)*(1-del.e))) *
      (l23^g * ( (l12/(l1. - l23)) *(exp(-l23*t) - exp(-l1.*t))))^
        ((1-del.b)*del.e) *
      (l23^g * exp(-l23*t))^(del.b)
        )))
  }

  opt.res <- vector('list', nr.int)
  init <- if(identical(init,'random')) runif(3, 0, .3) else init

  for(i in 1:nr.int){
  opt.res[[i]] <- optim(init, lik, , del.b = dmat[,i],
                        del.e = dmat[,i+1], gmat[,i], t = tmat[,i],
                        method = 'L-BFGS-B', lower = .002, upper = 1,
                        control = list(maxit = 5000), hessian =T)
  }

  par <- do.call('rbind', lapply(opt.res, function(x) x[[1]]))
  se <- do.call('rbind', lapply(opt.res, function(x) sqrt(diag(solve(x[[6]])))))
  colnames(par) <- colnames(se) <- c('l12', 'l13', 'l23')
  rownames(par) <- rownames(se) <- paste('int', 1:nr.int)
  par <- round(par, 5); se <- round(se, 5)
  low <- par - 1.96 *se; high <- par + se * 1.96

  ret <- data.frame(mle = matrix(t(par)),
                    se = matrix(t(se)),
                    CI.low = matrix(t(low)),
                    CI.high = matrix(t(high)))

  rownames(ret) <- paste(c('l12', 'l13', 'l23') ,
                          rep(1:nr.int, each = 3), sep = '.')

  list(ret = ret, LL = opt.res$value)
}
```

Finally the function to estimate the parameters from complete information data. Which has as arguments a data frame of which the first column are the discharge time and the second column are the transition times, and a vector with split points.

```
icu.pwc.fi <- function(dat,
                       phi = NULL){
  stop1 <- ifelse(dat[,2] == 0, dat[,1], dat[,2])
  start2 <- dat[,2][dat[,2] != 0]
  stop2 <- dat[,1][dat[,2] != 0]

  bounds <- c(0, phi, max(dat[,1]))
  at.risk1 <- matrix(0, nrow = length(stop1), ncol = length(bounds)-1)
  for(i in 1:ncol(at.risk1)){
    at.risk1[,i] <- ifelse(stop1 <= bounds[i+1], stop1-bounds[i], bounds[i+1] - bounds[i])
  }
  at.risk1 <- ifelse(at.risk1 > 0, at.risk1, 0)
  i=1
  int.start <- int.end <- matrix(0, nrow = length(stop2), ncol = length(bounds)-1)
  for(i in 1:ncol(int.start)){
    int.start[,i] <- ifelse(start2 >= bounds[i+1], bounds[i+1],
                            ifelse(start2 > bounds[i], start2,
                                   bounds[i]))
    int.end[,i] <- ifelse(stop2 <= bounds[i], 0,
                          ifelse(stop2 >= bounds[i+1], bounds[i+1], stop2))
  }
  at.risk2 <- int.end - int.start
  at.risk2 <- ifelse(at.risk2 > 0, at.risk2, 0)

  inc12 <- inc13 <- inc23 <- numeric(ncol(at.risk1))
  no.inf <- subset(dat, dat[,2] == 0); inf <- subset(dat, dat[,2] != 0)
  for(i in 1:length(inc12)){
    inc13[i] <- sum(no.inf[,1] <= bounds[i+1] & no.inf[,1] > bounds[i])
    inc12[i] <- sum(inf[,2] <= bounds[i+1] & inf[,2] > bounds[i])
    inc23[i] <- sum(inf[,1] <= bounds[i+1] & inf[,1] > bounds[i])
  }
 ret <- data.frame(
  l12 = round(inc12 /colSums(at.risk1), 5),
  l13 = round(inc13 / colSums(at.risk1), 5),
  l23 = round(inc23 / colSums(at.risk2), 5) )
  ret <- t(ret); colnames(ret) <- 1:(length(bounds) - 1)
  ret
}
```

A.3 Support Functions

Function to obtain the $\mathbf{P_j}$-matrices from a `icu.pwc` object. Returns a list with the appropriate matrices, return is `NULL` if the object has no split points.

```
get.Pmat <- function(icu.mod){
  phi <- icu.mod$phi; par <- icu.mod$est[,1]
  P11 <- function(t, l1.) exp(-l1.*t)
  P12 <- function(t, l12, l1., l23){
       (l12/(l1. - l23)*(exp(-l23*t) - exp(-l1.*t)))}
  P22 <- function(t, l23) exp(-l23*t)
  P132 <- function(t, l12, l23, l1.){
    ( (l12 * l23) / (l1. - l23) ) *
      ( ( (1-exp(-l23*t)) /l23) - ( (1-exp(-l1.*t)) / l1.) )}
  P131 <- function(t, l1., l13) (l13/l1.) * (1 - exp(-l1.*t))
  P23 <- function(t, l23) 1 - exp(-l23*t)

  Pmat1 <- if(phi[1] != 'none') {
           matrix(c(P11(phi[1], sum(par[1:2])),
                    P12(phi[1], par[1], sum(par[1:2]), par[3]),
                    P131(phi[1], sum(par[1:2]), par[2]),
                    P132(phi[1], par[1], par[3], sum(par[1:2])),
                    rep(0, 12)), byrow = T, nrow = 4)}

  Pmat2 <- if(length(phi) > 1) {
    matrix(c(P11(phi[2]-phi[1], sum(par[4:5])),
             P12(phi[2]-phi[1], par[4], sum(par[4:5]), par[6]),
             P131(phi[2]-phi[1], sum(par[4:5]), par[5]),
             P132(phi[2]-phi[1], par[4], par[6], sum(par[4:5])), 0 ,
             P22(phi[2]-phi[1], par[6]), 0,
             P23(phi[2]-phi[1], par[6]),
             rep(0,8)), byrow = T, nrow = 4)}

  Pmat3 <- if(length(phi) > 2) {
    matrix(c(P11(t-phi[2], sum(par[7:8])),
             P12(t-phi[2], par[7], sum(par[7:8]), par[9]),
             P131(t-phi[2], sum(par[7:8]), par[8]),
             P132(t-phi[2], par[7], par[9], sum(par[7:8])), 0 ,
             P22(t-phi[2], par[9]), 0,
             P23(t-phi[2], par[9]),
             rep(0,8)), byrow = T, nrow = 4)}

  ret <- list(Pmat1 = Pmat1, Pmat2 = Pmat2, Pmat3 = Pmat3)
  ret <- ret[lapply(ret, is.null) == F]
  ret
}
```

Function to obtain the $\mathbf{f_j(t_j)}$ matrix for time point t from a `icu.mod` object. Needs to be multiplied with the appropriate **P**-matrices to obtain the complete density.

```
get.fmat <- function(t,
                     icu.mod){
  phi <- icu.mod$phi; par <- icu.mod$est[,1]

  f131 <- function(t, l13, l1.) l13 * exp(-l1.*t)
  f132 <- function(t, l12, l23, l1.){
    ((l12*l23)/(l1. - l23)*(exp(-l23*t) - exp(-l1.*t)))}
  f23 <- function(t, l23) l23 * exp(-l23*t)

  fmat1 <- matrix(c(0,0,f131(t, par[2], sum(par[1:2])), f132(t, par[1], par[3],
                        sum(par[1:2])), rep(0, 12)), byrow =T, nrow = 4 )

  fmat2 <- if(phi[1] != 'none'){
    matrix(c(0,0, f131(t-phi[1], par[5], sum(par[4:5])),
             f132(t-phi[1], par[4], par[6],sum(par[4:5])), rep(0,3), f23(t-phi[1],
                        par[6]), rep(0,8)), byrow =T, nrow = 4 )}

  fmat3 <- if(length(phi) > 1){
    matrix(c(0,0, f131(t-phi[2], par[8], sum(par[7:8])), f132(t-phi[2], par[7],
                        par[9], sum(par[7:8])), rep(0,3), f23(t-phi[2], par[9]), rep(0,8)),
           byrow =T, nrow = 4 )}

  fmat4 <- if(length(phi) > 2){
    matrix(c(0,0, f131(t-phi[3], par[11], sum(par[10:11])), f132(t-phi[3], par[10],
                 par[12], sum(par[10:11])), rep(0,3), f23(t-phi[3], par[12]), rep(0,8)),
           byrow =T, nrow = 4 )}
  ret <- list(fmat1 = fmat1, fmat2 = fmat2, fmat3 = fmat3, fmat4 = fmat4)
  ret <- ret[lapply(ret, is.null) == F]
  ret
}
```

Obtain matrix $\mathbf{P_j(t_j)}$ from a `icu.mod` for time t

```
get.Pmat.t <- function(icu.mod, t){

  phi <- icu.mod$phi
  phi <- if(identical(phi, 'none')) NULL else phi
  par <- icu.mod$est[,1]
  P11 <- function(t, l1.) exp(-l1.*t)
  P12 <- function(t, l12, l1., l23){
    (l12/(l1. - l23)*(exp(-l23*t) - exp(-l1.*t)))}
  P22 <- function(t, l23) exp(-l23*t)
  P132 <- function(t, l12, l23, l1.){
    ( (l12 * l23) / (l1. - l23) ) *
      ( ( (1-exp(-l23*t)) /l23) - ( (1-exp(-l1.*t)) / l1.) )}
  P131 <- function(t, l1., l13) (l13/l1.) * (1 - exp(-l1.*t))
```

```
 P23 <- function(t, l23) 1 - exp(-l23*t)

 Pmat1 <- matrix(c(P11(t, sum(par[1:2])),
           P12(t, par[1], sum(par[1:2]), par[3]),
           P131(t, sum(par[1:2]), par[2]),
           P132(t, par[1], par[3], sum(par[1:2])),
           rep(0, 12)), byrow = T, nrow = 4)

 Pmat2 <- suppressWarnings(
   matrix(c(P11(t-phi[1], sum(par[4:5])),
           P12(t-phi[1], par[4], sum(par[4:5]), par[6]),
           P131(t-phi[1], sum(par[4:5]), par[5]),
           P132(t-phi[1], par[4], par[6], sum(par[4:5])), 0 ,
           P22(t-phi[1], par[6]), 0,
           P23(t-phi[1], par[6]),
           rep(0,8)), byrow = T, nrow = 4))

 Pmat3 <- suppressWarnings(
   matrix(c(P11(t-phi[2], sum(par[7:8])),
           P12(t-phi[2], par[7], sum(par[7:8]), par[9]),
           P131(t-phi[2], sum(par[7:8]), par[8]),
           P132(t-phi[2], par[7], par[9], sum(par[7:8])), 0 ,
           P22(t-phi[2], par[9]), 0,
           P23(t-phi[2], par[9]),
           rep(0,8)), byrow = T, nrow = 4))

 Pmat4 <- suppressWarnings(
   matrix(c(P11(t-phi[3], sum(par[10:11])),
           P12(t-phi[3], par[10], sum(par[10:11]), par[12]),
           P131(t-phi[3], sum(par[10:11]), par[11]),
           P132(t-phi[3], par[10], par[1], sum(par[10:11])), 0 ,
           P22(t-phi[3], par[12]), 0,
           P23(t-phi[3], par[12]),
           rep(0,8)), byrow = T, nrow = 4))

 phi <- c(phi, pmax(max(icu.mod$time, t)))

 ret <- if(is.null(phi)){
   Pmat1
 } else if(t <= phi[1]){
   Pmat1
 } else if(t <= phi[2]){
   Pmat2
 } else if(t <= phi[3]){
   Pmat3
 } else{
   Pmat4
 }
ret
```

```
}
```

Wrapper to the matrix functions to obtain the probabilities of getting discharged at *t* on a grid of time points running from *t*1 untill *tlast* with *steps*. It returns an object of class `icu.prob` which will produce the theoretical density estimates. The function `plot.prob.icu` produces the theoretical density plot using `ggplot2` to which we can add the empirical density manually to obtain the full diagnostics plot.

```
prob.icu <- function(icu.mod, t1, tlast, step){
  Pmat <- get.Pmat(icu.mod)

  phi <- icu.mod$phi; par <- icu.mod$est[,1]

  select.fmat <- function(t){
    fmat <- get.fmat(t, icu.mod)
    dens <- if (phi[1] == 'none' || t <= phi[1] ) {fmat[1]
    } else if (length(phi) > 1 && t > phi[1] && t <= phi[2] ||
               length(phi) == 1 && class(phi) == 'numeric' && t > phi) {fmat[2]
    } else if (length(phi) == 3 && t > phi[2] && t <= phi[3] ||
                 length(phi) == 2 && t > phi[2]){fmat[3]
    } else fmat[4]
  dens}

  all.t <- seq(t1, tlast, step)
  fmat <- sapply(all.t, select.fmat)

  fin.fmat <- mapply(
  function(n, m){if(n == 'fmat1'){m
  } else if(n == 'fmat2'){Pmat[[1]] %*% m
  } else if(n == 'fmat3'){Pmat[[1]] %*% Pmat[[2]] %*% m
  } else if(n == 'fmat4'){Pmat[[1]] %*% Pmat[[2]] %*% Pmat[[3]] %*% m}
  }, names(fmat), fmat, SIMPLIFY =F)

  dens.mat <- data.frame(t = all.t,
                         Uninf = sapply(fin.fmat, function(x) x[1,3]),
                         Infec = sapply(fin.fmat, function(x) x[1,4]))
  ret <- list(density = dens.mat)
  class(ret) <- 'icu.prob'
  ret
}

print.icu.prob <- function(obj) print(obj$density)

plot.icu.prob <- function(p.icu){
  library(ggplot2); library(reshape2)
  pl.dat <- melt(p.icu$density, 't')
  colnames(pl.dat) [2:3] <- c('Infection', 'prob')
  ggplot(pl.dat, aes(t, prob, col = Infection)) + geom_line() +
    xlab('Time') + ylab('Density') + ggtitle('Density based on model fit')
}
```

The function `LRT.icu` performs a likelihood ratio test, the smaller model needs to be fed as the first argument.

```
LRT.icu <- function(mod1, mod2){
  df <- nrow(mod2$est) - nrow(mod1$est)
  stat <- round(-2*mod1$LL + 2*mod2$LL, 3)
  p.value <- round(1 - pchisq(stat, df), 4)
  cat('Stat       DF      P-value', '\n',stat, '    ',df, '     ', p.value)
}
```

Finally the function to predict the state of a patient after t days, based on the model fit.

```
predict.icu <- function(icu.mod,
                        t){
  int.of.t <- sapply(t, function(x) sum (x > icu.mod$phi) + 1)
  Pmat <- get.Pmat(icu.mod)
  Pmat.t <- get.Pmat.t(icu.mod, t)
  Pmat.mult <- if(int.of.t > 1) Pmat[1:(int.of.t-1)] else NULL
  Pmat.mult[[length(Pmat.mult) + 1]] <- get.Pmat.t(icu.mod, t)
  Pmat.star <- Reduce('%*%', Pmat.mult, accumulate = T)

  #P11 and P12, survival untill t given in the P(t)
  ret <- round (unlist(tail(Pmat.star,1)) [c(1,5)], 4)

  #P131 and P132, distribution is the sum of all distributions
  ret[3:4] <- round(colSums(do.call('rbind', Pmat.star)) [3:4], 4)
  names(ret) <- c('ICU.Uninf', 'ICU.Inf', 'Disch.Uninf', 'Disch.Inf')
  ret
}
```

B Sampling Distributions of Endpoint-Only Parameters

The next graphics are density plots of the sampling distributions of the parameters obtained, obtained from the simulation study in section 5.

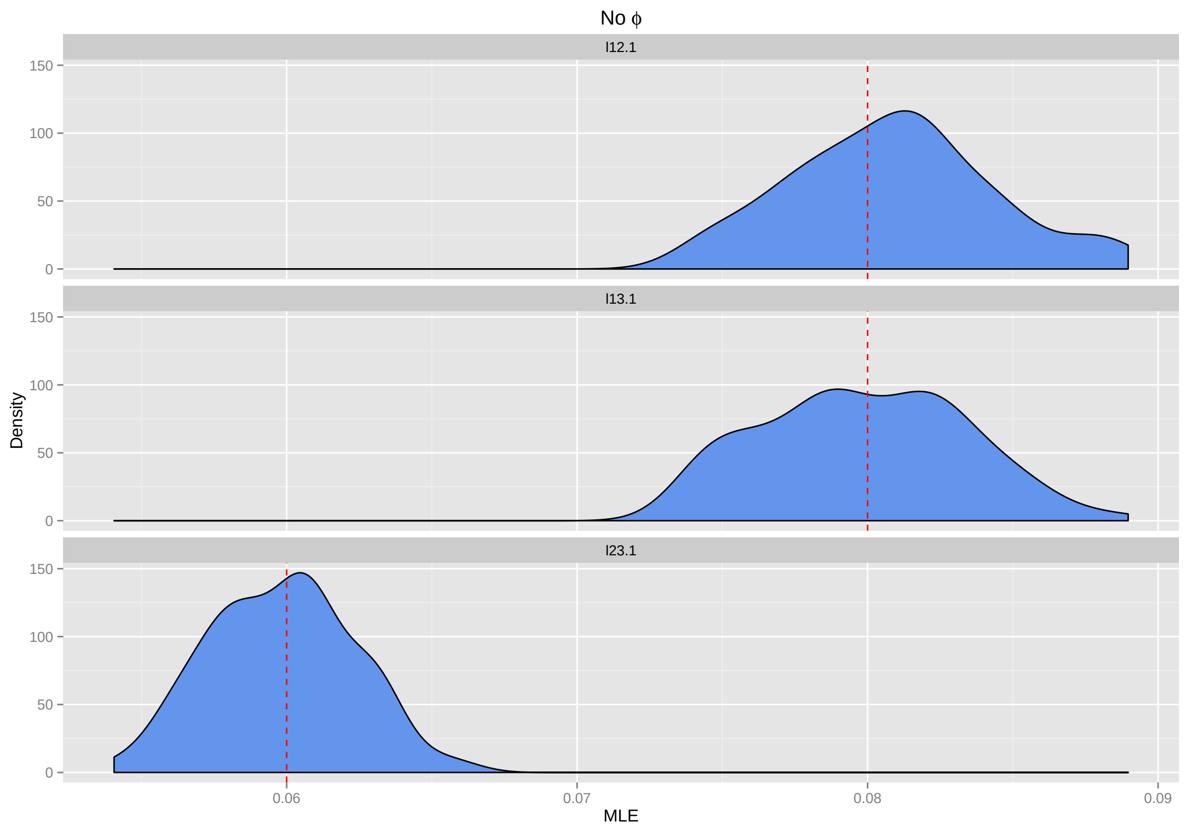

Figure 18: Sampling distributions of the three parameters of the model without split points

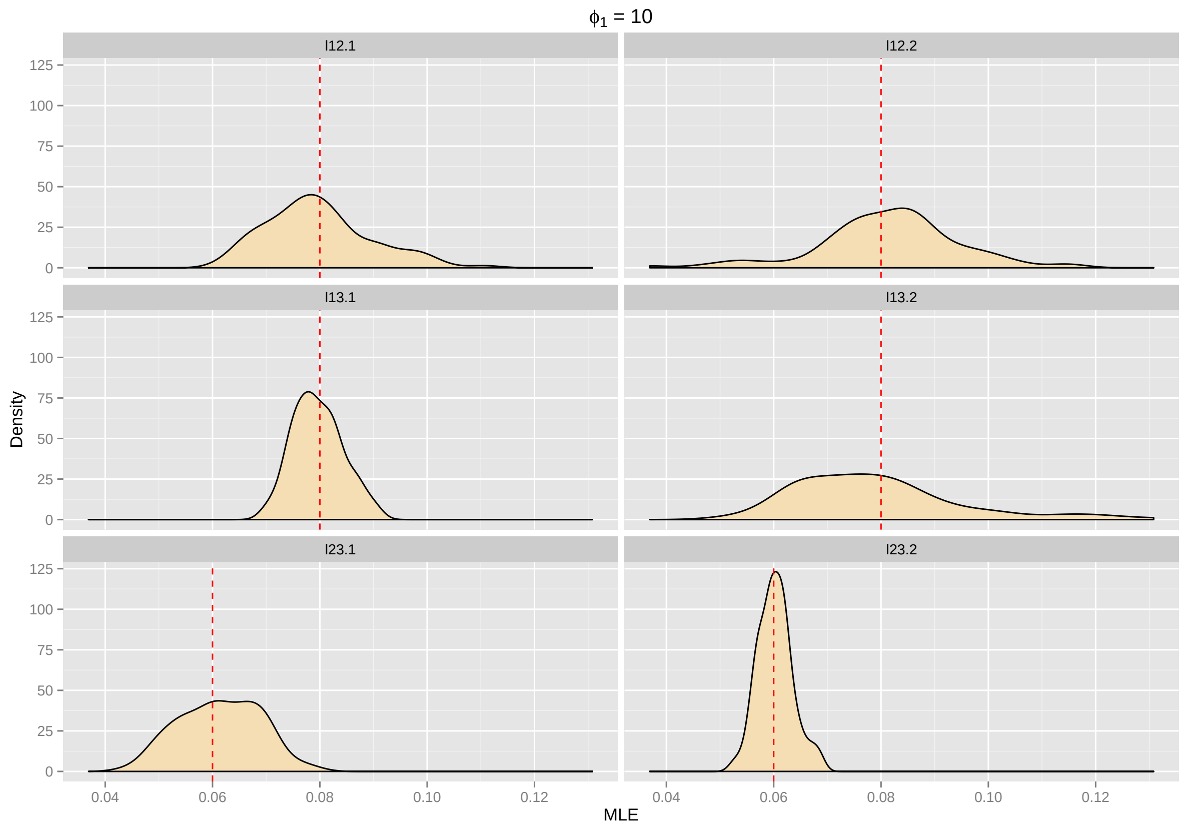

Figure 19: Sampling distributions of the three parameters of the model with split points at 10.

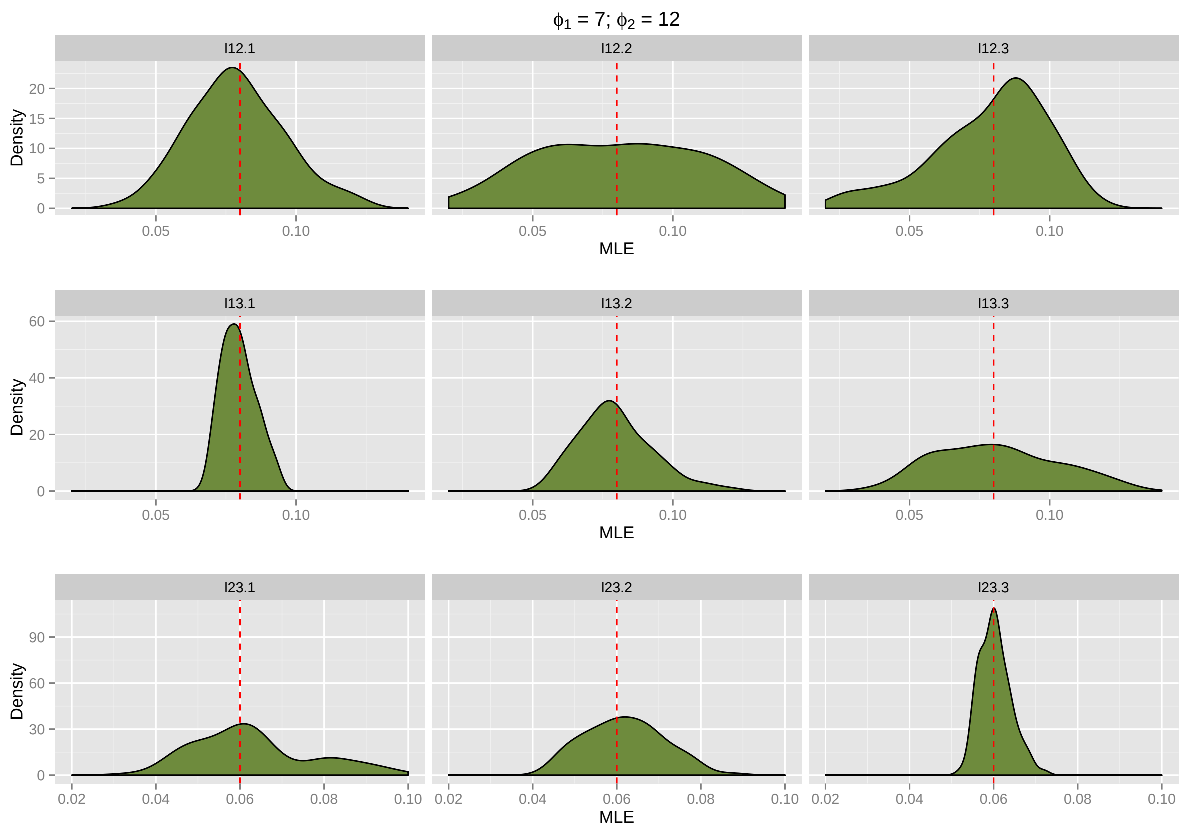

Figure 20: Sampling distributions of the three parameters of the model with split points at 7 and 12.

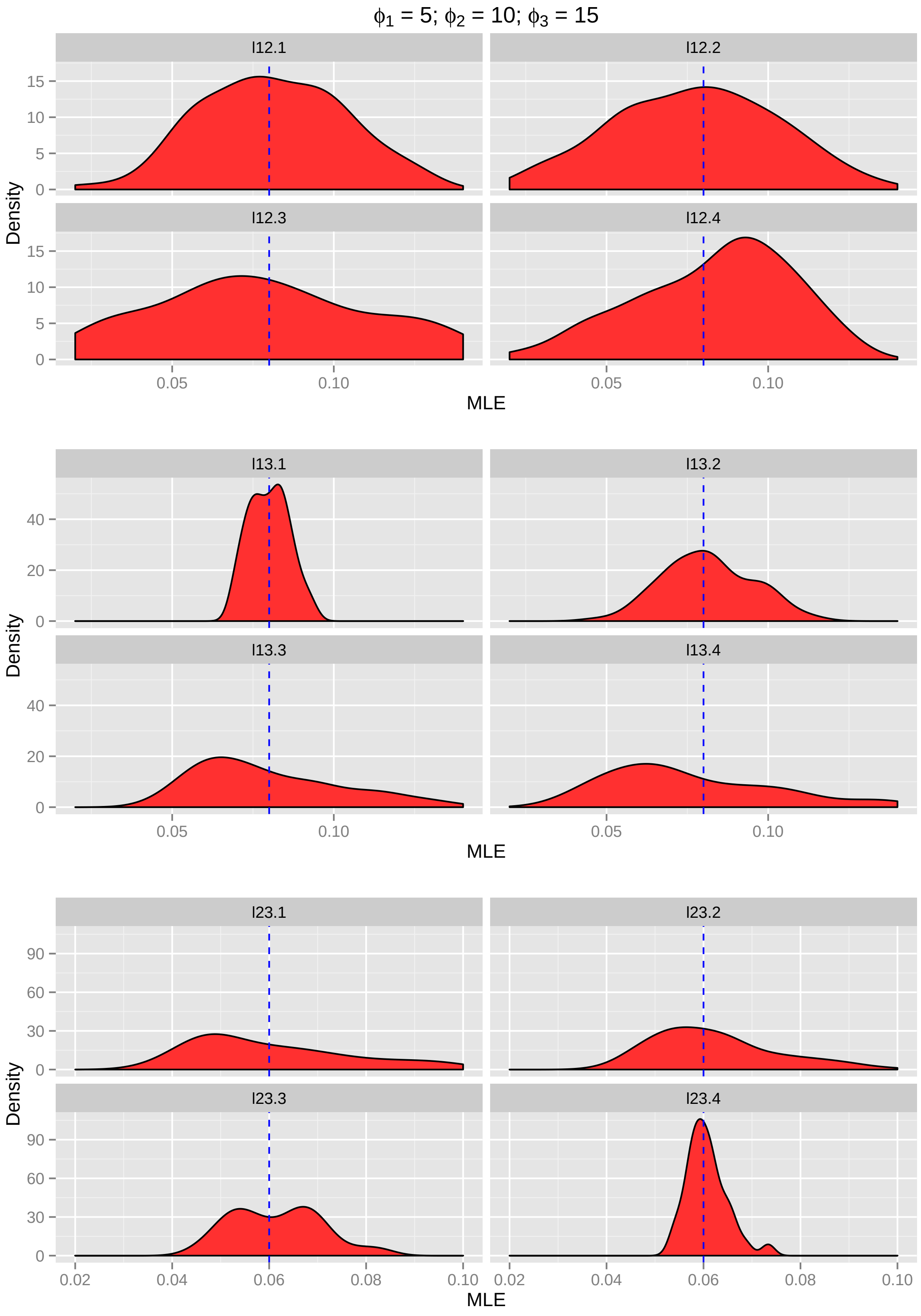

Figure 21: Sampling distributions of the three parameters of the model with split points at 5, 10, and 15.

C Example Session Data Analysis of the `kmi.pneu` Data Set

This is the code used to analyze the first data set.

```
require(kmi)
require(survival)
require(ggplot2)
require(xtable)

data(icu.pneu)
ip <- icu.pneu

head(ip)
ip <- ip[c('start', 'stop', 'pneu')]
summary(ip)


# We see one outlier, we remove it. Change factor level names
sort(ip$stop)
ip <- ip[ip$stop != 460, ]
ip$pneu <- factor(ip$pneu, labels = c('Uninfected', 'Infected'))

# Inspection of data
table(ip$pneu)
tapply(ip$stop, ip$pneu, mean)

# Graphical inspection of the data
ggsurv(survfit(Surv(stop) ~ pneu, data = ip), plot.cens = F,
       surv.col = c('darkorange', 'darkblue'), CI = T)

ggplot(ip, aes(stop)) +
  geom_histogram(aes(fill = pneu), binwidth = 1) +
  facet_wrap(~pneu) +
  xlab('Discharge time') +
  ylab('') +
  scale_fill_manual(values = c('darkblue','darkorange' ))

# Model estimation for endpoint only
pneu.no.phi <- icu.pwc(ip, 'stop', 'pneu', 'Infected')
xtable(pneu.no.phi$est, digits = 5)

plot(prob.icu(pneu.no.phi, 0, 100 , .1)) +
  stat_density(data = ip, aes(x = stop, y = ..count../1421,
               group = factor(pneu)), col = 'red',
               position = 'identity', geom = 'line', lty = 2) +
  xlim(c(0,50))

pneu.one.phi <- icu.pwc(ip, 'stop', 'pneu', 'Infected', 15)
pneu.one.phi <- fix.par.fix(pneu.one.phi)
```

```
plot(prob.icu(pneu.one.phi, 0, 50 , .1)) +
  stat_density(data = ip, aes(x = stop, y = ..count../1421,
               group = factor(pneu)), col = 'red',
               position = 'identity', geom = 'line', lty = 2) +
  xlim(c(0,50))
table(subset(ip, pneu == 'Infected', stop) < 5)
xtable(pneu.one.phi$est, digits = 5)

LRT.icu(pneu.no.phi, pneu.one.phi)

# two split points, without fixed parameters
pneu.two.phi.1 <- icu.pwc(ip, "stop", "pneu", "Infected", c(5,16), fix.par = "no")

# two split points with lambda 23 being fixed
pneu.two.phi.2 <- icu.pwc(ip, "stop", "pneu", "Infected", c(5,16), fix.par = "l23")

# LRT between the two models
LRT.icu(pneu.two.phi.2, pneu.two.phi.1)

pneu.two.phi.2 <- fix.par.fix(pneu.two.phi.2)
plot(prob.icu(pneu.two.phi.2, 0, 50, .1)) +
  stat_density(data = ip, aes(x = stop, y = ..count../1421,
                              group = factor(pneu)), col = 'red',
               position = 'identity', geom = 'line', lty = 2) +
  xlim(c(0,50))

xtable(pneu.two.phi.2$est, digits = 5)

# two split points with hazard ratio being fixed
pneu.two.phi.3 <- icu.pwc(ip, "stop", "pneu", "Infected", c(5,16),
                          fix.par = "prop.haz")
pneu.two.phi.3 <- fix.par.fix(pneu.two.phi.3)
plot(prob.icu(pneu.two.phi.3, 0, 50, .1)) +
  stat_density(data = ip, aes(x = stop, y = ..count../1421,
                              group = factor(pneu)), col = 'red',
               position = 'identity', geom = 'line', lty = 2) +
  xlim(c(0,50))


# estimate from panel data
pd <- make.panel.dat(ip[,c(2,1,3)], c(5,16))
pan.two.phi <- icu.pwc.pan(pd, 'Discharge', 'Infection', 'Infected', c(5,16))

plot(prob.icu(pan.two.phi, 0, 50, .1)) +
  stat_density(data = ip, aes(x = stop, y = ..count../1421,
                              group = factor(pneu)), col = 'red',
               position = 'identity', geom = 'line', lty = 2) +
  xlim(c(0,50))
```

D The ggsurv function

R objects of the class `survfit` can easily be plotted using the standard `plot.survfit` function. These plots are functional but not very attractive to look at. The R package `ggplot2` created by Hadley Wickham allows to construct plots from the ground up, creating better looking plots than the standard R plots. Using `ggplot2` the function `ggsurv` creates better looking survival plots, that are also easier to read because of the built in grid lines of `ggplot2`. The function is just as easy to use as the `plot.survfit` function. In the next sections you will find how to use `ggsurv`, some examples of use, and the code to implement the function.

D.1 Documentation

The full function looks like

```
ggsurv <- function(s,
                   CI = 'def',
                   plot.cens = T,
                   surv.col = 'gg.def',
                   cens.col = 'red',
                   lty.est = 1,
                   lty.ci = 2,
                   cens.shape = 3,
                   back.white = F,
                   xlab = 'Time',
                   ylab = 'Survival',
                   main = '')
```

The arguments are
`s` An object of class `survfit` as created by the `survfit` function from the `survival` package.

`CI = 'def'` Plot the 95% confidence interval of the survival estimate? Defaults to `T` for only one stratum, and `F` for multiple strata.

`plot.cens = T` Mark the censored observations?

`surv.col = 'gg.def'` Color of the survival estimate. Defaults to black for one stratum, and to the default `ggplot` colors for multiple strata. Length of vector with color names should be either 1 or equal to the number of strata.

`cens.col = 'red'` Color of the points that mark censored observations.

`lty.est = 1` Linetype of the survival curve(s). Vector length should be either 1 or equal to the number of strata.

`lty.ci = 2` Linetype of the bounds of the 95% CI.

`cens.shape = 3` Shape of the points that mark censored observations.

`back.white = F` If `TRUE` the background will not be the default grey of `ggplot` but will be white with borders around the plot.

`xlab = 'Time'` The label printed on the x-axis.

`ylab = 'Survival'` The label printed on the y-axis.

`main = ''` The plot label.

D.2 Examples

Using the `lung` data set from the `survival` package. First we make survival plots of all the patients, not creating any strata. Figure 22 is created by

```
data(lung)
s <- survfit(Surv(time, status) ~ 1, data = lung)
ggsurv(s)
```

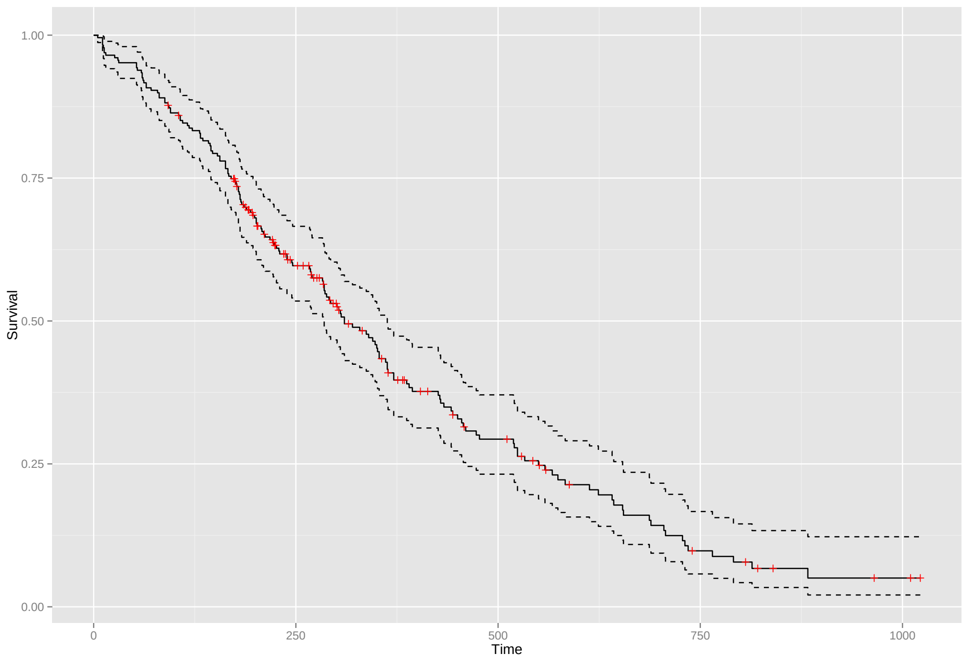

Figure 22: Default plot for a `survfit` object with one stratum only.

Adjusting the parameters of the plot is very similar to the `plot.survfit` function in `R`. This is how we create 23

```
ggsurv(s, surv.col = 'blue', cens.col = 'orange', lty.ci = 3,
       cens.shape = 8, back.white = T)
```

The `lung` data set contains the covariate *Sex*, allowing us to see whether survival is different for men and women. We first refit the survival object.

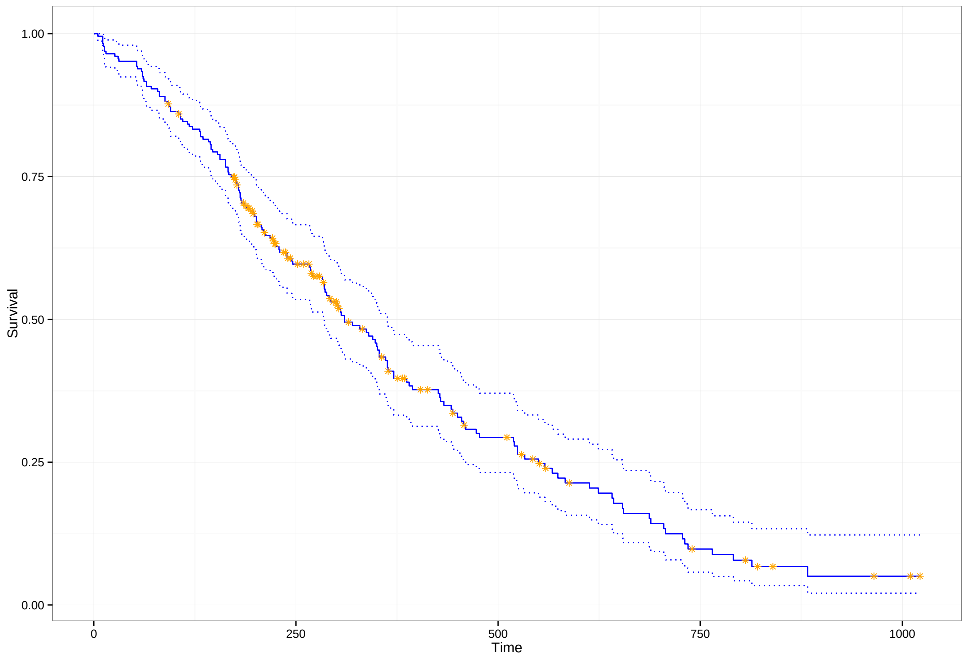

Figure 23: Customized plot, still single stratum.

```
lung$sex <- factor(lung$sex, labels = c( 'Male', 'Female'))
s.sex <- survfit(Surv(time, status) ~ sex, data = lung)
```

To create 24 we just enter the object into the `ggsurv` function, which will automatically recognize that it contains mulitple strata. By default it will no longer plot the confidence bounds with multiple strata.

```
ggsurv(s.sex)
```

The colors of the two survival lines are the first two colors `ggplot2` uses by default. If nothing is specified, these colors will be used. Note that the legend is created automatically. We can of course manually adjust the colors, and also change the other parameters just as with the single stratum plot. The following code produces 25

```
ggsurv(s.sex, surv.col = c('orange', 'purple'), CI = T, back.white = T,
        cens.col = 'black', cens.shape = 1)
```

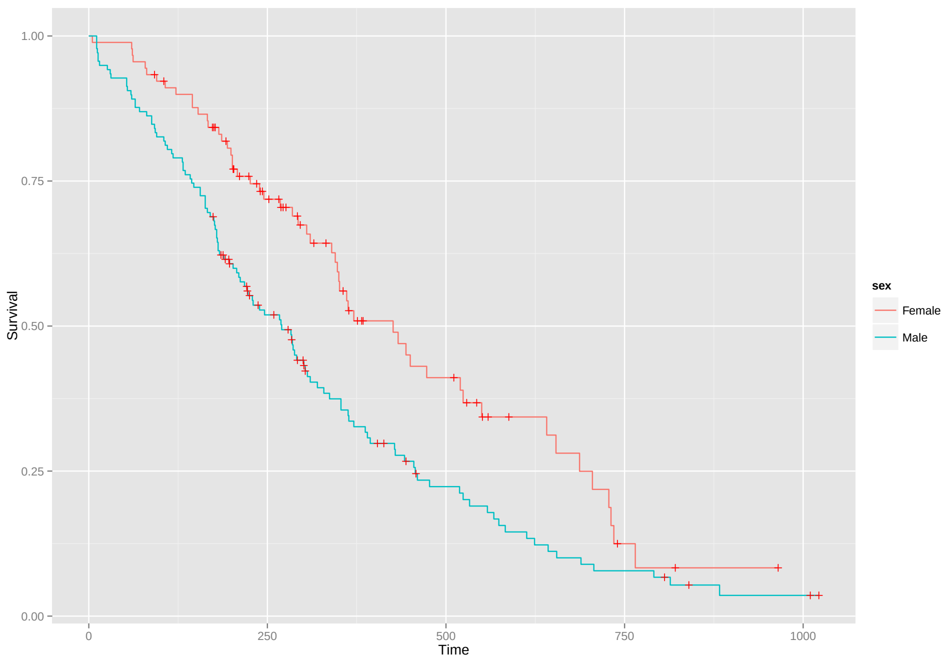

Figure 24: Default plot of a survival object with multiple strata.

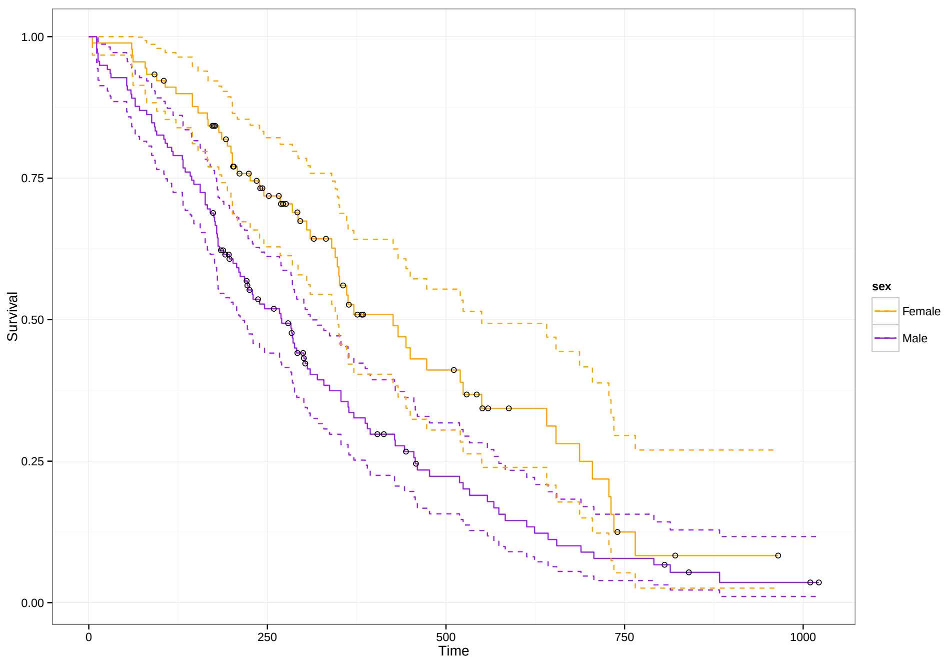

Figure 25: Modified plot of a survival object with multiple strata.

D.3 Modifying Plots after `ggsurv`

The produced plots are `ggplot` objects. A third advantage of `ggplot2` over ordinary R plotting, besides its better looking results and its flexibility, is that we can create plots in an iterative fashion by adding layers. This means that you can still modify the plot after using the `ggsurv` function. This time using the `kidney` data set.

```
data(kidney)
s.kid <- survfit(Surv(time, status) ~ disease, data = kidney)
pl.kid <- ggsurv(s.kid, plot.cens = F)
pl.kid
```

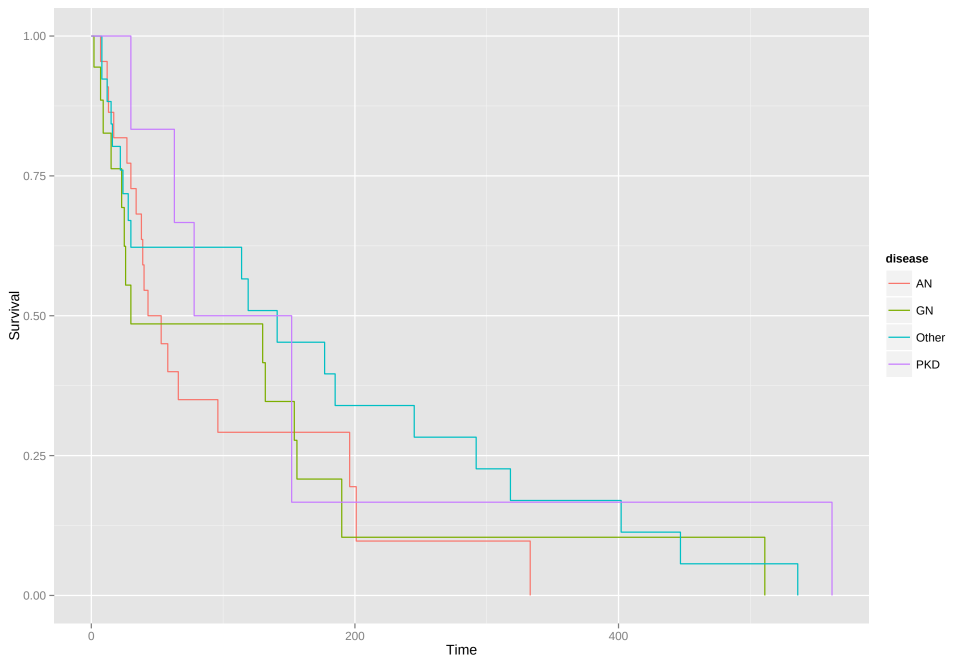

Figure 26: Survival plot of kidney data set.

Different from previous plots we have saved the plot in an object, just evaluating the object in the console automatically produces the plot, which is shown in Figure 26. The object creation allows us to add layers after the function. Maybe we would like to have a closer look at the survival in the first 80 days, we can use the `ggplot2` functions `xlim` and `ylim` to zoom and produce the plot in 27.

```
(pl.kid2 <- pl.kid + xlim(c(0, 80)) + ylim(c(0.45, 1)))
```

We can use the new object to modify the plot even further if we would like to. Instead of the legend we might want to add the group names to the plot directly. We save the colors in a vector, than add the text and finally remove the legend to produce 28.

```
col <- scales::hue_pal(h = c(0, 360) + 15, c = 100, l = 65, h.start = 0,
                       direction = 1)(4)
pl.kid2 + annotate("text", label = c('AN', 'GN', 'Other', 'PKD'),
  x = c(50, 20, 50, 71), y = c(.47, .55, .67, .8), size = 5, colour = col) +
  guides(color = F, linetype = F)
```

Inside the `ggsurv` function a double legend is created, since we can both assign different colours and different linetypes to the groups. They are automatically combined by the function, but in order to fully remove the legend both need to be switched off in the final command.

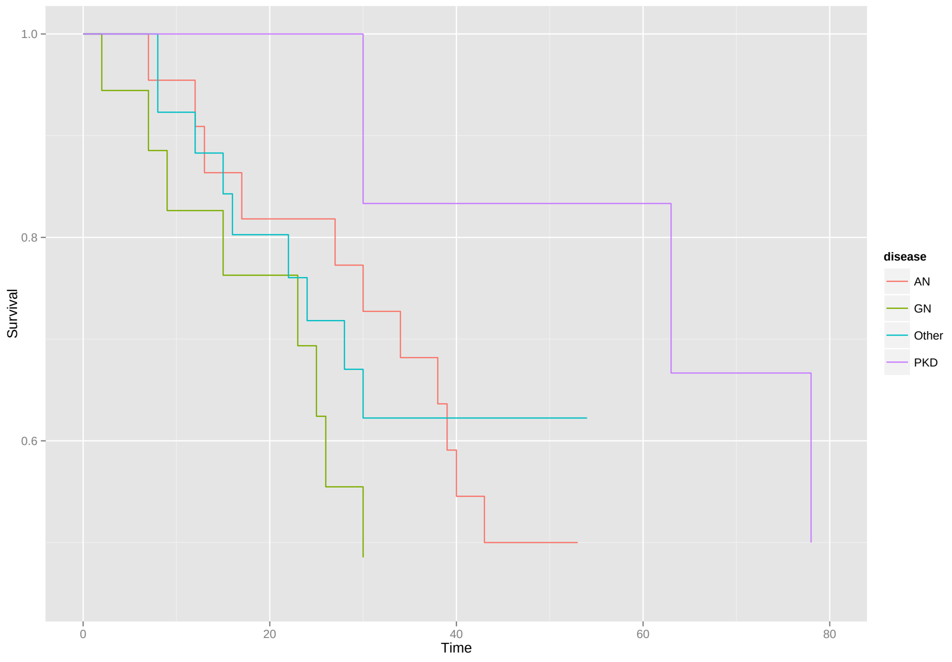

Figure 27: Figure 5, but now restricted to the first 80 days.

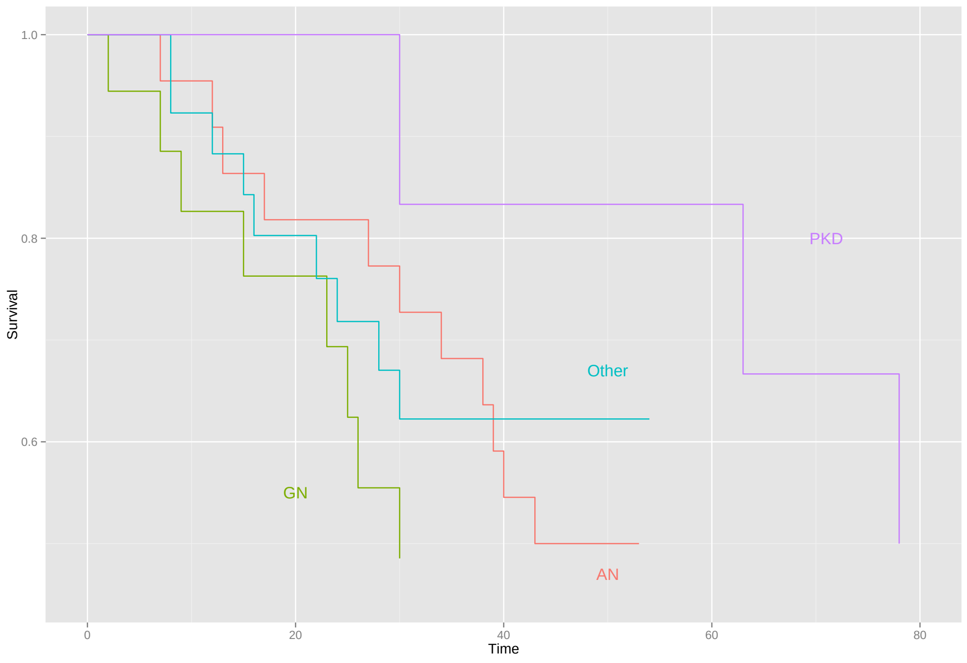

Figure 28: Removing the legend, add group names to the plot instead.

D.4 Function code

```
ggsurv <- function(s,
                   CI = 'def',
                   plot.cens = T,
                   surv.col = 'gg.def',
                   cens.col = 'red',
                   lty.est = 1,
                   lty.ci = 2,
                   cens.shape = 3,
                   back.white = F,
                   xlab = 'Time',
                   ylab = 'Survival',
                   main = ''){

  library(ggplot2)
  strata <- ifelse(is.null(s$strata) ==T, 1, length(s$strata))
  stopifnot(length(surv.col) == 1 | length(surv.col) == strata)
  stopifnot(length(lty.est) == 1 | length(lty.est) == strata)

  ggsurv.s <- function(s, CI = 'def', plot.cens = T, surv.col = 'gg.def',
                       cens.col = 'red', lty.est = 1, lty.ci = 2,
                       cens.shape = 3, back.white = F, xlab = 'Time',
                       ylab = 'Survival', main = ''){

    dat <- data.frame(time = c(0, s$time),
                      surv = c(1, s$surv),
                      up = c(1, s$upper),
                      low = c(1, s$lower),
                      cens = c(0, s$n.censor))
    dat.cens <- subset(dat, cens != 0)

    col <- ifelse(surv.col == 'gg.def', 'black', surv.col)

    pl <- ggplot(dat, aes(x = time, y = surv)) +
      xlab(xlab) + ylab(ylab) + ggtitle(main) +
      geom_step(col = col, lty = lty.est)

    pl <- if(CI == T | CI == 'def') {
      pl + geom_step(aes(y = up), color = col, lty = lty.ci) +
        geom_step(aes(y = low), color = col, lty = lty.ci)
    } else (pl)

    pl <- if(plot.cens == T) {pl + geom_point(data = dat.cens,
                                   aes(y = surv), shape = cens.shape,
                                  col = cens.col)
    } else if (plot.cens == T & nrow(dat.cens) == 0){
      stop('There are no censored observations')
    } else(pl)

    pl <- if(back.white == T) {pl + theme_bw()
    } else (pl)
    pl
```

```
 }
s <- S1
 ggsurv.m <- function(s, CI = 'def', plot.cens = T, surv.col = 'gg.def',
                      cens.col = 'red', lty.est = 1, lty.ci = 2,
                      cens.shape = 3, back.white = F, xlab = 'Time',
                      ylab = 'Survival', main = '') {
   n <- s$strata

   groups <- factor(unlist(strsplit(names
                                  (s$strata), '='))[seq(2, 2*strata, by = 2)])
   gr.name <-  unlist(strsplit(names(s$strata), '='))[1]
   gr.df <- vector('list', strata)
   ind <- vector('list', strata)
   n.ind <- c(0,n); n.ind <- cumsum(n.ind)
   for(i in 1:strata) ind[[i]] <- (n.ind[i]+1):n.ind[i+1]

   for(i in 1:strata){
     gr.df[[i]] <- data.frame(
       time = c(0, s$time[ ind[[i]] ]),
       surv = c(1, s$surv[ ind[[i]] ]),
       up = c(1, s$upper[ ind[[i]] ]),
       low = c(1, s$lower[ ind[[i]] ]),
       cens = c(0, s$n.censor[ ind[[i]] ]),
       group = rep(groups[i], n[i] + 1))
   }

   dat <- do.call(rbind, gr.df)
   dat.cens <- subset(dat, cens != 0)

   pl <- ggplot(dat, aes(x = time, y = surv, group = group)) +
     xlab(xlab) + ylab(ylab) + ggtitle(main) +
     geom_step(aes(col = group, lty = group))

   col <- if(length(surv.col == 1)){
     scale_colour_manual(name = gr.name, values = rep(surv.col, strata))
   } else{
     scale_colour_manual(name = gr.name, values = surv.col)
   }

   pl <- if(surv.col[1] != 'gg.def'){
     pl + col
   } else {pl + scale_colour_discrete(name = gr.name)}

   line <- if(length(lty.est) == 1){
     scale_linetype_manual(name = gr.name, values = rep(lty.est, strata))
   } else {scale_linetype_manual(name = gr.name, values = lty.est)}

   pl <- pl + line
```

```
    pl <- if(CI == T) {
      if(length(surv.col) > 1 && length(lty.est) > 1){
        stop('Either surv.col or lty.est should be of length 1 in order
             to plot 95% CI with multiple strata')
      }else if((length(surv.col) > 1 | surv.col == 'gg.def')[1]){
        pl + geom_step(aes(y = up, color = group), lty = lty.ci) +
          geom_step(aes(y = low, color = group), lty = lty.ci)
      } else{pl +  geom_step(aes(y = up, lty = group), col = surv.col) +
              geom_step(aes(y = low,lty = group), col = surv.col)}
    } else {pl}

    pl <- if(plot.cens == T & nrow(dat.cens) > 0) {
      pl + geom_point(data = dat.cens, aes(y =  surv)
                      , shape = cens.shape, col = cens.col)
    } else if (plot.cens == T & nrow(dat.cens) == 0){
      stop('There are no censored observations')
    } else(pl)

    pl <- if(back.white == T) {pl + theme_bw()
    } else (pl)
    pl
  }
  pl <- if(strata == 1) {ggsurv.s(s, CI , plot.cens, surv.col ,
                                  cens.col, lty.est, lty.ci,
                                  cens.shape, back.white, xlab,
                                  ylab, main)
  } else {ggsurv.m(s, CI, plot.cens, surv.col ,
                   cens.col, lty.est, lty.ci,
                   cens.shape, back.white, xlab,
                   ylab, main)}
  pl
}
```

The function will be in the `GGally` package on short notice.