A comparison of methods to estimate time-dependent correlated gamma frailty models

Author: Frank W.N. Boesten

Thesis Advisor: Dr. M. Fiocco (MI Leiden)

Thesis Advisor: Dr. H. Putter (LUMC)

July 2016
Abstract

Frailty models are widely used to account for unobserved heterogeneity by including a random term – the frailty – which is assumed to multiply the hazard of a subject (individual frailty) or the hazards of all subjects in a cluster (shared frailty). Especially the gamma distribution is popular for both discrete and continuous frailty models. Full likelihood inference involving distributions in which high-dimensional dependencies are present are rather difficult to deal with and therefore it is common to approximate likelihoods by using univariate or bivariate marginal distributions. Composite likelihood methods reduce the computational complexity of the full likelihood. So far, only composite likelihood procedures have been developed for discrete time-dependent autoregressive correlated gamma frailty models.

In this thesis we investigate the possibility of applying full-likelihood methods to an autoregressive correlated gamma frailty model. Building from previous work, two new gamma frailty models are proposed. Feasibility of different methods is researched; computation time is the biggest obstacle. A combination of the EM-algorithm and profile likelihood method is proposed to estimate the full likelihood. A method to estimate the standard error is described. The performance of the proposed methods is investigated by performing a simulation study.

The models proposed in this thesis are applied to a data set from a multicenter clinical trial provided by The European Organization for Research and Treatment of Cancer (EORTC trial 10854). The goal of the research was to study the effect of one course of perioperative chemotherapy given directly after surgery, compared to surgery alone for women with stage I or II invasive breast cancer. R-software environment is used to write code to estimate all models developed in this thesis. Codes and functions are written in a general form and can be applied to other data sets of similar structure. All R-code can be found in the appendix.
I would like to express my sincere gratitude to my two supervisors dr. M. Fiocco and prof. dr. H. Putter. I am grateful for their help and patience in improving my English scientific writing skills. I especially enjoyed the discussions on the direction of the research, for which they were always willing to schedule an appointment.

I would like to thank my professors from the Applied Mathematics master track for their excellent education. Furthermore, I am thankful for the flexibility of the Mathematical Institute of Leiden University in allowing me to develop myself outside the scope of mathematics.
4.4.1 R-code ........................................ 62
4.4.2 Results ....................................... 63
4.5 Conclusions ..................................... 66

5 Flexible Two-stage .................................. 68
  5.1 Introduction .................................... 68
  5.2 Algebraic Expressions ................................
      5.2.1 Marginal Distribution ......................... 69
      5.2.2 Joint Distribution ............................ 70
  5.3 R-code ........................................ 72
      5.3.1 Simulation of Data .......................... 73
      5.3.2 Likelihood Pairs of Observations ............ 75
      5.3.3 Estimation .................................. 77
      5.3.4 Standard Error ............................. 79
  5.4 Simulation Study ................................ 80
      5.4.1 Goal ..................................... 80
      5.4.2 R-code ................................... 81
      5.4.3 Results ................................... 81
  5.5 Conclusion ..................................... 83

6 Application to Data Set ............................ 85
  6.1 Introduction Data Set ............................ 85
      6.1.1 Breast Cancer ............................... 85
      6.1.2 Data Characteristics ......................... 86
      6.1.3 Exclusion of Patients from Analysis .......... 87
      6.1.4 Analysis .................................. 87
  6.2 Data Reconstruction .............................. 87
  6.3 Model Estimation ................................ 90
      6.3.1 EM ........................................ 91
      6.3.2 Two-stage .................................. 93
      6.3.3 Flexible Two-stage ......................... 96
  6.4 Conclusion ..................................... 99

7 Discussion .......................................... 100
  7.1 Full Likelihood .................................. 100
  7.2 Comparison .................................... 101
  7.3 Two-Stage Method ............................... 101
  7.4 Further Research ................................ 101

A R-code EM ......................................... 104
  A.1 Main R-file .................................... 104
  A.2 Single Simulation ............................... 108
  A.3 Simulation of Data ............................. 109
  A.4 EM-algorithm .................................. 109
  A.5 Two-stage Procedure ........................... 114
  A.6 Standard Error ................................ 116

B R-code Flexible Two-stage ......................... 118
  B.1 Main R-file .................................... 118
  B.2 Single Simulation ............................... 120
  B.3 Simulation of Data ............................. 121
  B.4 Estimation of Parameters ...................... 123
# List of Figures

1.1 A Typical Statisticians Journey ............................................. 7
3.1 Model 1 ................................................................. 24
3.2 Model 1 Collapsed ...................................................... 27
3.3 Model 1 in Venn Diagram ............................................... 29
3.4 Model 2 ................................................................. 30
3.5 Model 2 Construction .................................................... 30
3.6 Model 3 ................................................................. 34
6.1 Survival Curves Hospitals .................................................. 88
6.2 Survival Curve EM ...................................................... 92
6.3 Comparison Perioperative and Control EM ......................... 92
6.4 Hazard Ratio’s EM ....................................................... 93
6.5 Survival Curve Two-stage ............................................... 94
6.6 Comparison Perioperative and Control Two-stage ............... 95
6.7 Hazard Ratio’s Two-stage ............................................... 96
6.8 Survival Curve Flexible Two-stage .................................. 98
6.9 Comparison Perioperative and Control Flexible Two-stage .. 98
6.10 Hazard Ratio’s Flexible Two-stage ................................ 99
List of Tables

1.1 Made Up Data on Breast Cancer ........................................... 3
3.1 Summary of Models ............................................................ 37
4.1 Number of Summands EM ...................................................... 48
4.2 Long Format ........................................................................ 49
4.3 Wide Format ....................................................................... 49
4.4 Simulation Studies EM ......................................................... 63
4.5 Results Simulation Study EM Question 1 ................................ 64
4.6 Results Simulation Study EM Question 2 ................................ 65
4.7 Results Simulation Study EM Question 3 ................................ 65
4.8 Results Simulation Study EM Question 4 ................................ 66
4.9 Results Simulation Study EM Question 5 ................................ 66
5.1 Simulation Studies Flexible Two-Stage ................................. 81
5.2 Results Simulation Study Flexible Two-stage Question 1 ........ 82
5.3 Results Simulation Study Flexible Two-stage Question 2 ........ 82
5.4 Results Simulation Study Flexible Two-stage Question 3 ........ 83
6.1 Hospitals Participating EORTC Trial 10854 .......................... 86
6.2 Characteristics of Centers ..................................................... 87
6.3 Reconstructed Data EM Perioperative .................................... 91
6.4 Reconstructed Data EM Control .......................................... 91
6.5 Results EM ......................................................................... 91
6.6 Reconstructed Data Two-stage Perioperative ......................... 93
6.7 Reconstructed Data Two-stage Control .................................. 93
6.8 Results Two-stage ............................................................... 94
6.9 Reconstructed Data Flexible Two-stage Perioperative ............ 96
6.10 Reconstructed Data Flexible Two-stage Control .................... 97
6.11 Results Flexible Two-stage .................................................. 97
Chapter 1

Introduction

In Section 1.1 a general introduction into the subject of this thesis is given. This section is intended for readers without or with little knowledge of statistics, and can be skipped by readers with experience in this field. In Section 1.2 available literature on the subject is discussed. In Section 1.3 the thesis’s contribution is illustrated. In Section 1.4 the structure of the thesis is presented.

1.1 General Introduction

"Why are statisticians still necessary? Isn’t everything discovered already?"

— Frequently asked question

When statisticians explain what they do, they often are asked this question. It is understandable that people are inclined to think that there is nothing new to discover anymore. First of all, statistics courses only focus on explaining existing statistical methods. Secondly, the methods taught in these courses are often very general. Therefore, it seems that there is no need for new statistical methodology, as it seems that there is already a method for every situation.

They could not be more wrong in believing that the field of statistics is ‘done’. What is the motivation for statisticians to develop new models? It all starts with a researcher who calls for the help of a statistician. The statistician then goes through the following steps:

1. Reformulate the research question into a statistical framework;
2. Find an appropriate statistical model;
3. Execute statistical algorithm;
4. Communicate results to researcher.

A statistical question is a question concerning statistics which can be answered by collecting data. A statistic is a single measure of some attribute of a sample. A statistical model is a set of assumptions about the way that data is generated. A statistical algorithm is a computation method used to calculate the value of the statistic based on the data.
For example, a researcher might want to investigate who is more popular amongst adolescents: Justin Bieber or Taylor Swift. Assume the researcher asked 100 adolescents, and suppose that 59 replied that Justin Bieber is more popular than Taylor Swift. The researcher wants to know whether he or she can now conclude that Justin Bieber is more popular. The statistician then might transform this question to the following statistical question: “Is the probability for replying ‘Bieber’ to be the more popular than Lady Gaga significantly different from $p = 0.5$?”

In the second step, the statistician might decide to make use of a standard Neyman-Pearson hypothesis test. If the statistician wants to use this hypothesis test, he has to make some assumptions. For example, he needs to assume that the 100 interviewed adolescents the researcher are representative for the entire population of adolescents. In the third step, the researcher could make use of standard software programs (for example SPSS) to calculate the P-value. In the fourth step, the statistician would explain how the researcher should interpret the P-value.

A statistician develops a new statistical method when he finds out that he cannot find a method to answer the question of the researcher. This might be because there is no statistical model to answer the question, or it might be that a models exist, but that the assumptions that have to be made in order to apply the method, cannot be made in the context of the research question. In the example above, the researcher would have to be able to explain why the 100 adolescents are representative for the entire population in order to apply the Neyman-Pearson hypothesis test. If this cannot be assumed, another method has to be found. If it does not exist, a new method has to be developed.

The research question that motivated this thesis came from the field of medicine. A medical researcher asked an apparently simple question: “How big is the variability between hospitals where breast cancer is treated?” In particular, the researcher was interested in studying the time to relapse (i.e. the cancer comes back). Note that we do not want to rank hospitals, but want to determine to what degree there is variability between the hospitals. When naively approaching this question, one could believe that this question can be answered by using simple statistical methods which are already known to us for a century. An example of time to relapse for two hospitals is shown in Table 1.1. In this case, we could calculate the mean time to recurrence for both hospitals. If we make the following assumption: “The patient groups in both hospitals are similar”, we can apply the student’s t-test (first described in 1908). The outcome of this test is a t-statistic, which tells us whether the difference between the observed mean time to recurrence is statistically significant. If the difference is significant, we can use this difference as a measure of the variability between hospitals.

However this naive approach is not correct. First of all, as we can see in the table, some of the data is quite old. Some patients were treated for breast cancer 15 years ago. Maybe hospitals have changed treatments since then, or other hospital personnel is giving the treatment. We thus want to be

---

1This example is meant as an illustration how a statistician works. This does not mean that this is the way a statistician should answer this research question; in the past decades, the P-value has been extensively criticized.
able to draw conclusions based on more recent data. If we set the range of the study to, for example, 5 years, it may happen that some patients have not experienced relapse by the end of the study. Moreover, some patients might die before relapse. Breast cancer is a disease that can be treated today, and patients often do not experience relapse. Excluding these two types of patients from the data set would severely bias our analysis, because they often are a large proportion of the population.

A third problem with this naive approach, is that types of patients that the two hospitals treat might be different. There are many different types of breast cancer. Patients’ characteristics such as tumor size and nodal status must be taken into account when performing the analysis. We cannot assume that hospitals treat the same types of patients. For example, in the Netherlands, the most difficult patients are often treated in University Hospitals. Therefore, when hospitals are ranked, University Hospitals often do not seem to perform well because of this reason. If we do not account for difference between patients across hospitals, a large part of the observed variability might just be due to these differences.

Patients might differ in disease, but also in demographic factors. For example, age is known to be an important factor in predicting relapse. If we do not take this demographic factor into account, we will also not be able to make a fair comparison. In general, there are more younger people in the cities than in the country side. If we thus not take age into account, we are not able to fairly compare hospitals in cities with hospitals in rural areas.

We can thus see that our earlier named method from 1908 is not applicable anymore. Applying it would be a grave mistake, for three reasons:

1. By the end of study, some patients have not experienced relapse;
2. Some patients will die before relapse;
3. Patients are different.

Therefore, after the second world war, statisticians started developing methods to deal with these kind of situations, giving rise to the field of "Survival
Analysis. The first method to deal with the first two issues, was introduced by Kaplan and Meier (1958) who proposed a method to estimate the survival for the group of interest. Cox’s Proportional Hazard Model (1972) can be used to incorporate patient’s characteristics in the analysis. Vaupel et al. (1979) extended Cox’s model to be able to estimate the variability (or heterogeneity) between clusters (hospitals in our case) of individuals. With each step taken, statisticians attempt to make their statistical model as realistic as possible. When models become more realistic, they also become more complex. As a rule of thumb, the more general (and thus unrealistic) a statistical model is, the earlier it was discovered. Only the more general (and thus older) methods are taught in introductory courses in statistics. Student then think that those methods can be used on any situation. This might be an explanation about the often asked question stated at the beginning of this chapter.

What is the contribution to the field of Survival Analysis of this thesis? To answer this question, we need to dive more deeply into the details of the existing literature concerning heterogeneity between clusters of individuals in survival data. As the answer to this question is too technical to discuss in this section, which is meant as a general introduction, it is discussed further in Sections 1.2 and 1.3.

We will now explain what kind of problems arise when a statistician tries to develop a new statistical model. This thesis illustrates what kind of problems a modern statistician encounters, it is thus a good example of a typical modern day statistician’s journey.

The first step in a typical statistical journey is to build a statistical model. When building a statistical model, the statistician has two goals. First of all, he must attempt to build a mathematical model which approximates reality as closely as possible. This model has to be build around the statistical question that is posed by the researcher in the field (an oncologist in our case). Almost always, additional assumptions have to be made in order to build a statistical model. Ideally, these assumptions are not too restrictive to make the model as realistic as possible. The more restrictive they are, the less situations can be applied to use the model. Often, assumptions concerning the distribution of the data have to be made. It is not unusual that, the statistician needs to write mathematical proofs to show that the model has the desired properties.

The second goal is to provide a statistical algorithm for the statistics of interest. There are two things the statistician has to do. Firstly, he has to find algebraic expressions for the statistical parameters. Usually, these expressions are cumbersome or difficult to evaluate. Often, the problem is that integrals cannot be evaluated algebraically. Numerical approximation can be used, but if these are computationally extensive, this might slow the calculating procedure too much for the statistical method to be feasible. Also, the statistician has to prove mathematically that the statistical algorithm works.

Building the statistical model is often the most difficult part of the statistical journey. There are often many ways to build the statistical model, each with advantages and disadvantages. As discussed in Section 1.2 there are at
least five ways of incorporating heterogeneity into a Survival Model. In this thesis we focus on a particular way of doing this: frailty models. But as we will see, there are many different ways of building a frailty model. At least seven decisions have to be made on the structure of the model. Also, when the model is build, there are a lot of options to build a statistical algorithm. In our case, there are at least four types of statistical algorithms that can be applied to frailty models. Applying such a method is a time consuming process, as it involves evaluating a lot of expressions algebraically.

After building a statistical model, the statistician has to write a computer program to execute the algorithm, as most modern statistical algorithms are far too complex to execute with a normal calculator. The most common used programming language amongst statisticians is R (R Core Team, 2014), as it is specifically designed for statistics. One of the advantages of R is that everyone can build a package that contains functions which can be used freely. Using packages often reduces the amount of time a statistician has to invest in programming, but there are often still obstacles to overcome. Sometimes, packages are not available and specific R code has to be written. Also, using a computer can sometimes give numerical problems. Rounding errors can occur for example. Another problem is computation time. Often a statistical algorithm works, but it takes too long to execute to be of practical value to researchers.

After building the model and programming it in R, it is necessary to test whether we did not make any mistakes in the first two steps. The derivation of the algebraic expressions are often cumbersome, and R-code often consists of hundreds of lines of code. At this stage mistakes can easily be made. It may also happen that the method developed is not robust. As stated earlier, often assumptions on the distribution of the data need to be made. In our case, we assume that the data is distributed through Poisson and Gamma distributions. In real world data, it will almost never happen that the data is perfectly distributed through the desired distributions. Also, it is often difficult to test whether the data is distributed the way we assumed it to be. A statistical algorithm is called robust when it performs well on a wide range of probability distributions.

We test whether the statistical algorithm works by doing simulation studies. As noted before, the goal of every statistical algorithm is to estimate parameters. When a researcher simulates data, the real distribution of the data is known. As a consequence, the true parameters are known. We then apply the statistical algorithm on the data and evaluate how far off the estimates are from the true parameters. This provides a measure of how the proposed method works. Several different scenarios are simulated to study how the algorithm works. Doing simulations is often computationally extensive. Statisticians generally repeat the simulation on a certain configuration at least 500 times.

The communication part can also take a lot of time of a statistician’s job. Of the nine months spent on this thesis, we spent at least 30% of the time on writing this thesis and writing presentations to communicate our findings. The first step in making the statistical method available for researchers, is to write a paper (or thesis) which contains all technical details about the
method and its performance. However, implementing a method from a paper (or thesis) is a time consuming process, as the statistician has to then write R-code. As this can take up to a couple of weeks, a method often will not be implemented if it is only documented on a paper. Therefore, the second step in making a statistical method available is to write a R-package (or a package in another programming language) so that other people can apply the specific method. As working with R requires knowledge of programming and advanced knowledge of statistics, it is often still not assessable for non-statisticians. If you want to make an method available for a more general public, one can choose to write packages for programs such as SPSS.

Communicate results to the researcher can be difficult. To illustrate how a statistical algorithm works and how to interpret results is very demanding. As modern day statistical methods are often complex, this is not an easy task. Often non-statisticians find it difficult to understand even the most basic statistical methods.

Wulff et al. (1987) conducted a research on Danish doctors by letting them fill in a short multiple choice test which tested basic statistical knowledge. They concluded from this research that “that the statistical knowledge of most doctors is so limited that they cannot be expected to draw the right conclusions from statistical analyses published in papers in medical journals”. For example, only 18% of the respondents were able to answer a question concerning P-values correctly. As almost every paper in medicine contains a P-value, this is highly concerning. A lot of research has been done in the past decades to assess how often mistakes are made in the statistical analysis in papers. Many papers have been written on this subject. For a recent example, see Nieuwenhuis et al. (2011). They conducted a research on the statistical analysis in 513 published papers in five top-ranking journals (Science, Nature, Nature Neuroscience, Neuron and The Journal of Neuroscience). They focused on mistakes made in statistical test which test the difference between two experimental effects. They found that approximately in half of the papers, mistakes were made in the statistical analysis. These two examples illustrate nicely that communicating a statistical method to non-statisticians is not an easy task.

A modern day statistician’s journey thus consists of four steps: building a statistical model, programming it in R, carrying out simulation studies, and communicating our method to other people (statisticians, and sometimes non-statisticians). This process is summarized in Figure 1.1. All four steps are performed in this thesis. The only step not required in this thesis, is the communication of results to clinicians since this thesis is meant for statisticians.

1.2 Literature Review

Standard models in Survival Analysis assume homogeneity: all individuals are subject to the same hazard $\mu(t)$. However, often it cannot be assumed that the hazard rate is the same for all individuals. Therefore, most models in Survival Analysis allow the introduction of covariates. Each individual $i$
is associated with a vector \( W_i(t) = (W_1(t), \ldots, W_p(t)) \), where \( p \) is the number of covariates and \( t \) is the time point. In most cases, the covariates are assumed to be time-independent. The hazard rate (in the time independent case) is then given by:

\[
\mu_i(t|W_i) = \mu_0(t) \cdot c(\beta^t W_i)
\]

where \( \mu_0(t) \) is an arbitrary baseline hazard rate, \( \beta = (\beta_1, \ldots, \beta_p)^t \) is a parametric vector and \( c(\beta^t W_i(t)) \) is a known function. However, even when covariates are included, we cannot always assume that the covariates explain all the differences in hazard rate between individuals or clusters of individuals. Variance that cannot be explained by incorporating covariates into the model is called unobserved heterogeneity.\(^2\) If we do not take into account this unobserved heterogeneity, life expectancy and potential gains in life expectancy from health and safety interventions can be wrongly estimated (Vaupel et al., 1979).

One way of introducing heterogeneity is by multiplying the hazard rate of individuals with a random factor. When random terms are incorporated into a survival model, the hazard function for an individual \( i \) at time \( t \) is thus multiplied by a random term \( Z_i \):

\[
\mu_i(t|W_i, Z_i) = Z_i \cdot \mu_0(t) \cdot c(\beta^t W_i).
\]

\(^2\)Models to incorporate unobserved heterogeneity are also used outside of the field of Survival Analysis. They can be used in any field of application of event history analysis: life sciences, demography social sciences, economy etc. (Putter and van Houwelingen, 2015)
The random term can be individual-specific or group-specific. The name “frailty” was first introduced by Vaupel et al. (1979) to describe these random factors. The frailty term for an individual or cluster of individuals tells how ‘frail’ the specific individual or cluster of individuals is. The higher the frailty, the frailer the individual or group of individuals. Since then, many papers have been published on this topic. When a researcher incorporates frailty terms into a survival model, decisions have to be made on the following topics:

1. How are the hazard function or the survival function modeled? Parametric or non-parametric, piecewise constant? etc.
2. Are covariates included? If yes, how?
3. How are the frailty terms distributed?
4. Are the frailty terms time-dependent?
5. Is the frailty variance constant? If not, how is this incorporated in the model?
6. Is correlation incorporated into the frailty terms? If so, what is the correlation structure?
7. What kind of statistical algorithm is used to estimate the parameters?

As the research and application of frailty models progresses, undoubtedly other topics will rise as well. For example, frailty models can be combined with competing risk models.

Developing computationally feasible statistical algorithms for frailty models is not an easy task for two reasons. First, the frailty terms are not observed directly from the data. Therefore, frailty terms have to be estimated within the statistical algorithm alongside the model parameters. Second, the expressions for the full likelihood functions of frailty survival models are often quite cumbersome. In many cases, standard software to fit frailty models is missing. Therefore, if a model is changed on one of these topics, a lot of effort has to be put in developing a computationally feasible new statistical algorithm to estimate the parameters of the model. More often than not, a slightly changed model is worth a new publication. One can thus imagine that the literature available on the subject of frailty models is vast and wide.

Often, the frailty terms are assumed to be gamma-distributed. Gamma frailty-models fit well into the proportional hazard model. Therefore, it is easy to do algebraic manipulations. This makes estimation of the model parameters easier. However, other frailty distributions are also available. See for example Hougaard (2000) chapters 7 through 11 for an introduction to different types of frailty distributions and advantages and disadvantages.

Abbring and van den Berg (2007) showed that under some regularity conditions, the frailty distribution among the survivors at a specific time converges to a gamma distribution. This supports the use of gamma frailty models in the field of Survival Analysis.

How to include covariates and how to model the hazard function are topics which are not specific to frailty survival models. However, the decision on
how to model these can have big impact on computational feasibility. The hazard function can be parametric or non-parametric. Although it is most natural to assume that the hazard rate is not piecewise constant, piecewise constant hazard functions are developed (for applications, see for example [Paik et al. (1994) or [Wintrebert et al. (2004)]. Two reasons motivated this research: first, these models are often more simple than continuous time hazard models. The second motivation comes from the meta-analysis for survival data. Here, a set of survival curves addressing the same research question coming from published studies are available. The aim is to pool the curves together to estimate overall survival. By choosing time intervals, from each survival curve it is possible to estimate how many events and how many censored observations took place to estimate the hazard rate per time interval. (See for example [Fiocco et al. (2009)]).

Both discrete and continuous models are thus of interest to statisticians. Covariates are often incorporated into frailty models by using Cox’s proportional hazards model [Cox 1972].

The most simple frailty models are not time-dependent, have constant variance and correlation is not incorporated (see [Vaupel et al.]). Many time-varying frailty models have been developed. Among these are frailty models based on stochastic processes (see for example [Aalen (2008), chapter 11], diffusion processes (see for example [Aalen and Gjessing (2004)]), and Lévy-type processes (for an application, see for example [Gjessing et al. (2003)]). For an overview, see [Wienke (2010)]. [Unkel et al. (2014)] proposed a model where also the variance of the frailty terms is time-varying.

Often, related to the concept of the time-dependent frailties, there is the correlation aspect. If the frailty terms are time-dependent, correlation between the frailty terms can be introduced. Correlation is incorporated to make models more natural. For example, if group-specific frailty factors are used, it is assumed that all the unobserved heterogeneity can be explained by the differences between groups. By incorporating correlation into the model, this assumption is relaxed ([Auget et al. 2007]). See for example [Yashin and Iachine (1995), Petersen (1998) and Wienke et al. (2003) and see [Wienke (2010)] for examples of correlated frailty models for different types of frailty distributions.

As stated before, the statistical algorithms for frailty models are often computationally extensive. Among the methods used are the Expectation-Maximization algorithm (see Section 2.4.5 for an introduction), first introduced by [Dempster et al. (1977)] (for an application, see for example [Petersen et al. (1996)]) and the Monte Carlo Markov Chain algorithm. MCMC is used since late 1940s and started to become popular in early 1990s ([Robert and Casella, 2012]) (for an application, see for example [Jones (2004)]). As these are two computationally extensive methods that are not feasible in every situation, also composite likelihood procedures are used. These procedures do not use the full likelihood to build a statistical algorithm, but use the composition of individual component log likelihoods. The name ‘composite likelihood’ was first introduced by [Lindsay (1988)]. See [Varin (2008)] for an overview of applications of composite likelihood in Survival Analysis.
Chapter 1. Introduction

There are also other ways of incorporating unobserved heterogeneity into a Survival Model. Among these are the fixed effects model, the stratified model, the copula model and the marginal model. For an overview, see Duchateau and Janssen (2008).

In this thesis, we continue the work of Henderson and Shimakura (2003) and Fiocco et al. (2009b). The former proposed a Poisson-gamma model to account for between-subjects heterogeneity and within-subjects serial correlation occurring in longitudinal count data. The event counts are conditionally independent Poisson random variables given the value $Z$ of a gamma-distributed frailty term. They used a particular relation between the multivariate normal and multivariate gamma distributions to construct this process. The gamma frailty process has mean 1 with unknown variance $\xi$ and correlation structure $\text{Corr}(Z(s), Z(t)) = \rho^{|t-s|}$. The novelty in this paper was the inclusion of an autoregressive structure in the Poisson-Gamma model. However, the full likelihood was so complicated that it became unmanageable (Varin, 2008). Therefore, they proposed a Newton-Raphson composite likelihood procedure based on all pairs of time points to estimate the model parameters.

There are two disadvantages in this model. First, the statistical algorithm is based on composite likelihood, which is suboptimal. Second, when counts are high, rounding errors can occur.

Fiocco et al. proposed a new multivariate gamma distribution. The gamma process retains the desired mean, variance and correlation structure, but has explicit finite-dimensional distributions. The gamma process is constructed by using independent additive components with common components between the frailty terms to produce the desired correlation structure. Although an explicit expression for the full likelihood of the model, it is intractable. Therefore, the authors applied a composite likelihood procedure on the marginal and bivariate distributions. As this is still a high-dimensional maximization problem, they proposed a 2-stage procedure, where the marginal distributions are used to estimate all parameters except the frailty correlation, and a second stage where the pairs of observations are used to estimate only the correlation. The authors applied the longitudinal count data frame to discrete survival data. This is a common approach in Survival Analysis. The event counts are assumed to be Poisson-distributed with rate parameter depending on the hazard rate and the number at risk per time interval. In a big simulation study (Fiocco et al., 2012) the performance of the composite likelihood methodology of Fiocco et al. and Henderson and Shimakura have been compared. The Poisson correlated Gamma frailty model applied to a meta analysis with a single arm per study, has been extended to a meta analysis for studies with two arms (Fiocco et al., 2009a) and to a meta-regression (Fiocco et al., 2012). In addition, in the last paper the robustness of the Poisson correlated Gamma frailty model has also been investigated.

The models of Henderson & Shimakura and Fiocco et al. have been generalized to continuous time (see Putter and van Houwelingen (2015)). The authors construct the frailty terms by using a compound birth-death process and the parameters of the model are estimated by using the expectation-maximization algorithm on the full likelihood.
One could wonder why we would want to develop a model based on the full likelihood for the discrete case, while a method for the continuous case is already proposed. As described before, discrete models are of interest for meta-analysis studies where one does not have access to the original data sets.

1.3 Aims of Thesis

In this thesis, we continue the work of Fiocco et al. The goal of this thesis is to assess whether we can further improve it. First, we will assess whether it is possible to use full-likelihood instead of a composite likelihood algorithm to estimate the model parameters. Then, we want to test how well the composite likelihood works compared to a full-likelihood. Finally, we propose a flexible two-stage algorithm.

1.4 Structure of Thesis

In Chapter 2, some basic concepts in Survival Analysis and types of statistical algorithms are illustrated. This chapter is intended for readers with little knowledge of Survival Analysis. In Chapter 3, we investigate whether the full likelihood function can be made more tractable by simplifying the multivariate gamma distribution proposed by Fiocco et al. (2009a). Two new distributions are proposed. In Chapter 4, the EM-algorithm – a full likelihood method – is applied. The R-code for the implementation of the EM-algorithm is discussed and a simulation study is carried out to compare the performance of the two methods. In Chapter 5, a flexible two-stage composite likelihood procedure is proposed. In Chapter 6, the methodology developed in the previous two chapters is applied to a data set. In Chapter 7, possibilities for further research are outlined. The statistical analysis is performed in the R-software environment. All R-code written for this thesis can be found in the appendix.
Chapter 2

Basics of Frailty Survival Models

In this chapter, the basics of Frailty Survival Models are introduced. In Section 2.1, general concepts of survival analysis are discussed. In Section 2.2, it is explained how covariates can be included in the analysis of survival data. In Section 2.3, it is discussed how frailty terms can be included as well. In Section 2.4, it is discussed how frailty survival models can be estimated.

As the field of Survival Analysis is vast and wide, we only focus on concepts used in this thesis. For a more thorough introduction of the field of Survival Analysis, see for example Klein and Moeschberger (2003).

2.1 Basic Functions in Survival Analysis

The goal of Survival Analysis is to estimate the time to an event of interest. Let $T$ be non-negative random variable denoting the time to an event of interest. $T$ can represent the time of interest of any phenomenon. Examples are: time to death after treatment, onset of disease, moment an adolescent leaves the home of his parents, time to divorce from first marriage, or moment at which a television breaks down after first usage. As the examples illustrate, Survival Analysis can be applied in a wide range of sciences, among which are: Medical, Demography, Economy or Industry.

$T$ is often called the \textit{survival time}, because in medicine, $T$ usually denotes the time a patient dies or the onset of a disease. $T$ then thus indicates how long a patient ‘survived’ without experiencing the event.

2.1.1 The Survival Function

In statistics, the \textit{Cumulative Distribution Function} (CDF) is used to describe the distribution of a random variable. It is defined as $F(t) = P(T < t) = \int_{s=0}^{t} f(s) \, ds$ (where $f(s)$ denotes the \textit{Probability Density Function}, PDF). It can be interpreted as the probability that $T$ will take a value less than or equal to $t$. The \textit{Survival Function} is defined as:

$$S(t) = 1 - F(t) = P(T \geq t) = \int_{s=t}^{\infty} f(s) \, ds.$$  \hfill (2.1)
Chapter 2. Basics of Frailty Survival Models

The Survival Function represents the probability that the event of interest has not occurred until time \( t \). The survival curve is monotone and non-increasing. The probability that the event did not take place at time \( t = 0 \) is 1 and goes to 0 as \( t \) goes to infinity: \( \lim_{t \to \infty} S(t) = 0 \).

### 2.1.2 The Hazard Function

The hazard function gives the rate at which an individual, who has survived to time \( t \), will experience the event in the next instant of time. It is defined as:

\[
\mu(t) = \lim_{\Delta \to 0} \frac{P(t \leq T < t + \Delta | T \geq t)}{\Delta}.
\]  

(2.2)

If \( T \) is a continuous random variable, this simplifies to:

\[
\mu(t) = \frac{f(t)}{S(t)}.
\]  

(2.3)

The hazard function describes the underlying failure mechanism of the event of interest. The function can take many different shapes. For example, if the hazard rate is high at the beginning and then decreases, this means that the event of interest is most likely to occur close to \( t = 0 \), and less likely to occur later on.

Sometimes, it is preferred to study the cumulative hazard function, which is defined as:

\[
M(t) = \int_{s=0}^{t} \mu(s) \, ds.
\]  

(2.4)

### 2.1.3 Transforming Basic Functions

The PDF, CDF, survival function, hazard function and cumulative hazards function are all functions which describe the distribution of the event of interest \( T \). All these functions can be transformed into each other. When \( T \) is continuous, the following holds:

\[
\begin{align*}
f(t) &= -\frac{dS(t)}{dt} \\
h(t) &= -\frac{d\ln[S(t)]}{dt} \\
H(t) &= -\ln[S(t)] \\
S(t) &= \exp[-H(t)] = \exp[-\int_{s=0}^{t} h(s) \, ds]
\end{align*}
\]  

(2.5)

Similar transformations hold for the discrete case.
2.2 Covariates

Often, researchers are not only interested in analyzing the distribution of $T$ (through the hazard function or survival function), but also in factors – often called covariates – associated with the distribution of the event of interest.

Covariates are often included in the survival model by employing Cox’s Proportional Hazards Model (Cox, 1972). The idea behind the model is that the hazard rate is not the same for every individual, but depends on the covariate values of an individual. Suppose that there are $n$ individuals and for each individual $i$ there is a vector $W_i = (W_{i1}, \ldots, W_{ip})$, where $p$ represents the number of covariates. The hazard rate for an individual $i$ is given by:

$$
\mu_i(t|W_i) = \mu_0(t) c(\beta^t W_i) 
$$

where $\mu_0(t)$ is an arbitrary baseline hazard rate which is the same for every individual, $\beta = (\beta_1, \ldots, \beta_p)$ is a parametric vector and $c(\beta^t W_i)$ is a known function. One of the most common models for $c(\beta^t W_i)$ is as follows:

$$
c(\beta^t W_i) = \exp(\beta^t W_i) = \exp(p \sum_{k=1}^p \beta_k W_{ik}).
$$

The model is called a ‘proportional hazards model’, because the hazard rates of two individuals are proportional to each other. Let $W_1$ and $W_2$ be the covariates corresponding to two individuals. The hazard ratio is given as:

$$
\frac{\mu(t|W_1)}{\mu(t|W_2)} = \exp\left[p \sum_{k=1}^p \beta_k (W_{1k} - W_{2k})\right]
$$

which is a constant. This is also called the proportional hazards assumption which implies that the survival times of individuals are independent of each other, which is not always true.

The proportional hazards model can be extended to incorporate time-dependent covariates. Suppose that for each individual $i$ there is a vector $Z_i(t) = (Z_{i1}(t), \ldots, Z_{iq}(t))$ where $q$ represents the number of time-varying covariates. The hazard rate for an individual $i$ is as follows:

$$
\mu_i(t|W_i, Z_i(t)) = \mu_0(t) \exp\left(p \sum_{k=1}^p \beta_k W_{ik} + \sum_{r=1}^q \beta'_r Z_{ir}(t)\right)
$$

where $\beta' = (\beta'_1, \ldots, \beta'_q)$ is a parametric vector. The ratio of two individuals with covariate values $(W_1, Z_1(t))$ and $(W_2, Z_2(t))$ is:
\[
\frac{\mu(t|W_1, Z_1(t))}{\mu(t|W_2, Z_2(t))} = \exp \left[ \sum_{k=1}^{p} \beta_k(W_{1k} - W_{2k}) + \sum_{r=1}^{q} \beta'_r(Z_{1r}(t) - Z_{2r}(t)) \right].
\] (2.10)

In Section 2.3 an extension of Cox’s proportional hazards model which handles dependence in survival data is discussed.

### 2.3 Frailty Models

In the medical field, the term frailty is often used to indicate that frail people have an increased risk for complication and mortality. This is also referred to as heterogeneity, which indicates variation in treatment outcome between patients.

Part of the heterogeneity can be explained by incorporating covariates into the model. However, not all variation can be explained by observed heterogeneity, as it is impossible to observe every variable influencing the event of interest. Variance that cannot be explained by covariates is called unobserved heterogeneity. If we do not take into account this unobserved heterogeneity, life expectancy and the impact of covariates can be wrongly estimated (Vaupel et al., 1979).

Currently, a popular way to account for unobserved heterogeneity, is the frailty model. A frailty is an unobservable random component which influences survival of individuals. The variance of this random component is a measure for unobserved heterogeneity. Unobserved heterogeneity can be incorporated on both individual and group-level.

In most frailty models, this frailty term acts multiplicatively on the hazard rate of the subgroup members. The hazard rate corresponding to subject \(i\) with frailty \(Z_i\) is defined as:

\[
\mu(t|X, Z_i) = Z_i \mu_0(t) \exp[\beta'X_i],
\] (2.11)

where \(Z_i\) is the individual- or group-specific frailty term. The frailty term for an individual or cluster of individuals describes how ‘frail’ the specific individual or cluster of individuals is. The higher the frailty, the frailer the individual or group of individuals.

To avoid unidentifiability issues between the baseline hazard rate \(\mu_0\) and the frailty terms \(Z\), restrictions must be put on the distribution of the frailty terms. Therefore, we assume that the frailty terms are independently drawn from a random variable \(Z\) with \(\mathbb{E}[Z] = 1\). Also, assumptions on the distribution of \(Z\) have to be made. As discussed in Section 1.3 many types of frailty distributions exist. The frailty variance is of interest, as it provides a measure for the heterogeneity between individuals or clusters of individuals. In this thesis, the notation \(\text{Var}(Z) = \xi = \theta^{-1}\) is used.

The frailty term can also be assumed to be time dependent: \(Z(t)\). Time varying frailty frailty models are of interest for two reasons. First, the assumption that the frailty term is constant over time is restrictive. Second,
Chapter 2. Basics of Frailty Survival Models

Time-dependent frailty terms allow for correlation between frailty of individuals or clusters of individuals between time points.

Time varying frailty models where correlation is incorporated between the frailty terms are a generalization of time-constant frailty models. When the correlation between the frailty terms is one, then the frailty terms are constant over time.

2.4 Estimation

When time to event and the covariate values of individuals are observed, the hazard function \( \mu_0(x) \), covariate vector \( \beta \) and the frailty parameters can be estimated. Most statistical algorithms are based on the likelihood function introduced in Section 2.4.1. In survival analysis, computing the likelihood function is often complicated by censoring, which is discussed in Section 2.4.2. In Section 2.4.3 we introduce the model used in this thesis. In Sections 2.4.4, 2.4.5 and 2.4.6 three likelihood based statistical algorithms are introduced.

2.4.1 Likelihood

Let \( x \) be a set of outcomes as discussed in Section 2.1. A statistical model describes the data-generating process of the outcomes. The model describes the probability distributions generating the data. Let \( \theta \) denote the set of parameter values which describe these distributions. The likelihood of \( \theta \), given the outcomes \( x \) is defined as the probability of those observed outcomes given the parameter values:

\[
L(\theta|x) = P(x|\theta).
\]  

Based on the data \( x \), estimates of \( \theta \) can be found by maximizing the likelihood function. The statistical algorithm tries to find the values of \( \theta \) which maximize the likelihood function. This type of parameter estimation is called maximum likelihood estimation (MLE).

In practice, it is often more convenient to maximize the log likelihood, defined as

\[
\ell(\theta|x) = \log L(\theta|x).
\]  

Because the logarithm is a monotonically increasing function, the logarithm of a function achieves its maximum value at the same point as the function itself.

Suppose that not all the data can be directly observed. In that case, the data \( x \) can be split into two parts: \( x = (x_{\text{obs}}, x_{\text{mis}}) \), where \( x_{\text{obs}} \) denotes the observed data and \( x_{\text{mis}} \) denotes the missing data. In such cases, it is not possible to maximize \( L(\theta|x_{\text{obs}}, x_{\text{mis}}) \). We maximize the observed likelihood function, which is defined as:
Chapter 2. Basics of Frailty Survival Models

\[ L_{\text{obs}}(\theta|x_{\text{obs}}) = \int_{x_{\text{mis}}} L(\theta|x_{\text{obs}}, x_{\text{mis}}) dx_{\text{mis}}. \]  

(2.14)

The corresponding observed log likelihood function is denoted as:

\[ \ell_{\text{obs}}(\theta|x_{\text{obs}}) = \log(L_{\text{obs}}(\theta|x_{\text{obs}})). \]  

(2.15)

The full likelihood \( L(\theta|x_{\text{obs}}, x_{\text{mis}}) \) is often referred to as the complete data likelihood function when unobserved data is present. The following notation is often used when expressions become cumbersome:

\[ L_{\text{obs}}(\theta) = L_{\text{obs}}(\theta|x_{\text{obs}}) \]  

(2.16)

\[ \ell_{\text{obs}}(\theta) = \ell_{\text{obs}}(\theta|x_{\text{obs}}). \]  

(2.17)

2.4.2 Censoring

Constructing the likelihood function of a survival model is often complicated by censoring or truncation. These two problems distinguish survival analysis from other fields of statistics. When the time to event is not directly observed, the event is called censored. Generally speaking, there are three types of censoring: right-censoring, left-censoring and interval censoring.

- **When an event is right-censored**, it is known that the event has not occurred before a certain time point, but we do not know when the event has taken place after that time point. Let \( T \) denote the event of interest and \( C_l \) the time of censoring. Define an indicator variable \( \delta = \mathbb{1}_{X \leq C_l} \). Instead of observing \( T \), we thus observe \((X, \delta)\), where \( X = \min(T, C_l) \). The contribution to the likelihood from this censored observation is given by \( P(T > X) \).

For example, in the data set discussed in Chapter 6, the recurrence of cancer after treatment is one of the events of interest. If a patient dies before recurrence, or a patient is lost to follow-up, it is only known recurrence did not take place until the last follow-up time.

- **Left-censoring** occurs when an event is known to have occurred before a certain time point, but we do not know exactly when. Let \( C_l \) denote the time of censoring and define the indicator variable \( \delta = \mathbb{1}_{X \geq C_l} \). Instead of observing \( T \), we thus observe \((X, \delta)\) where \( X = \max(T, C_l) \). The contribution to the likelihood from this censored observation is given by \( P(T < X) \).

For example, when a patient is asked when certain symptoms started, a patient might not be able to tell exactly when the symptoms started, but he might say that the symptoms started before a certain time point.

- **Interval censoring** occurs when an event is known to have occurred within a certain time interval. Let \( L \) and \( R \) denote the left and right
time point. Instead of observing $T$, we thus observe $(L, R]$. The contribution to the likelihood from this censored observation is $P(L < T \leq R)$. This type of censoring often occurs when periodic inspections are performed to check whether the time of event took place.

For example, when women have been treated for breast cancer, they are often routinely checked whether the cancer has returned. If the cancer is diagnosed at one of these checkups, the relapse took place between the two time points.

2.4.3 Longitudinal Data Frame

Many models to construct the likelihood function for censored time to event data exist. See for example Klein and Moeschberger (2003) for an introduction to this subject. We restrict ourselves to discussing the model used in this thesis, namely the Longitudinal Count Data Frame, which is largely used also outside the field of Survival Analysis. In Chapter 6 an application to survival data is discussed.

The longitudinal count data frame is used when observations are spread over specific periods of time. Let $W$ denote the end of the study. We divide the interval $[0, W]$ into $T$ intervals. Let $t_1, \ldots, t_T$ denote the end points of these intervals. Let $N$ denote the number of clusters of individuals (also called the number of units). For each unit $i$ ($i = 1, \ldots, N$) and for each interval $j$ ($j = 1, \ldots, T$), a number of counts $y_{ij}$ is observed.

It is often assumed that the counts are Poisson distributed:

$$Y_{ij} \overset{d}{=} P_{\text{pois}}(\mu_{ij}), \quad (2.18)$$

where $\mu_{ij}$ is the hazard rate for interval $i$ and unit $j$. If we include a frailty component $z_{ij}$ to account for unit heterogeneity, we obtain:

$$Y_{ij} | Z_{ij} \overset{d}{=} P_{\text{pois}}(z_{ij}\mu_{ij}). \quad (2.19)$$

Thus, conditional on the unobserved frailty terms, the counts are independently Poisson distributed random variables. Marginally, each $Z_{ij}$ is assumed to be gamma distributed with both shape and rate parameter equal to $\theta$:

$$Z_{ij} \overset{d}{=} \Gamma(\theta, \theta). \quad (2.20)$$

By constructing the frailty terms in this way, we have $E[Z_{ij}] = 1$ and $\text{Var}[Z_{ij}] = \zeta = \theta^{-1}$.

To account for correlation within units between time points, first-order autoregressive correlation is assumed between time intervals $l$ and $k$: $\text{corr}(Z_{il}, Z_{ik}) = \rho^{\mid l-k\mid}$.

To be able to evaluate the complete likelihood function for this model, we apply the law of conditional probability:
\begin{align*}
\mathcal{L}(\mu, \xi, \rho | Y, Z) &= P(Y | Z, \mu, \xi, \rho) \times P(Z | \mu, \xi, \rho) \\
&= P(Z | \xi, \rho) \times \prod_{j=1}^{T} \prod_{i=1}^{N} P_{\text{pois}}(Z_{ij} \mu_{ij}),
\end{align*}

where \((Y_{ij}), (Z_{ij})\) and \((\mu_{ij})\) denote the matrix of counts, the frailty components and the hazard rate respectively. \(Z\) follows a multivariate gamma distribution. Thus, if an expression for the multivariate gamma distribution with marginal distributions \(\Gamma(\theta, \theta)\) and correlation structure \(\text{corr}(Z_{ij}, Z_{ik}) = \rho^{\lvert j-k \rvert}\) is obtained, the likelihood for the model can be evaluated. In Chapter 3 different ways of describing the distribution of \(Z\) are discussed. Since \(Z\) is unobserved, we cannot maximize the complete data likelihood. Therefore, to find estimates \((\hat{\mu}, \hat{\xi}, \hat{\rho})\) for the parameters, the observed data likelihood has to be maximized:

\begin{align*}
\mathcal{L}_{\text{obs}}(\mu, \xi, \rho | Y) &= \int_{Z} P(Z | \xi, \rho) \times \prod_{j=1}^{T} \prod_{i=1}^{N} P_{\text{pois}}(Z_{ij} \mu_{ij}) dZ. 
\end{align*}

Cox’s proportional hazards model can be included in this framework by making the hazard rate for a unit \(i\) and time point \(j\) depend on covariates:

\[\mu_{ij} = \mu_{ij} \times \exp(\beta W_{ij}),\]

where \(\beta = (\beta_1, \ldots, \beta_p)\) is a parametric vector, \(p\) is the number of covariates, and \(W_{ij}\) is a vector which contains information on the covariate values of the hazard rate in unit \(i\) and time point \(j\). An extra set of parameters \(\beta\) has to be estimated.

### 2.4.4 Profile Likelihood

Suppose that the set of parameters \(\theta\) of a statistical model can be partitioned in two sets: \(\theta = (\phi, \eta)\). When we use the maximum likelihood method to estimate the parameters, the goal is to find

\[
(\hat{\phi}, \hat{\eta}) = \arg \max_{\phi, \eta} \mathcal{L}(\phi, \eta).
\]

However, when the set of parameters \((\phi, \eta)\) is large, this might be difficult. In such cases, the profile likelihood function can be used. The idea is as follows: suppose that \(\phi\) is known, then we can find an estimate for \(\eta\) by evaluating:

\[
\hat{\eta}(\phi) = \arg \max_{\eta} \mathcal{L}(\phi, \eta).
\]

When \(\phi\) is unknown, we can proceed as follows: for each \(\phi\), \(\hat{\eta}(\phi)\) can be evaluated. Estimates for both \(\phi\) and \(\eta\) can be found by maximizing the profile likelihood function

\[
\mathcal{L}_{\text{prof}}(\phi) = \mathcal{L}(\phi, \hat{\eta}(\phi))
\]
Section 2.4.5 EM-algorithm

The expectation maximization algorithm (EM-algorithm) [Dempster et al. 1977] switches iteratively between an expectation step (E-step) followed by a maximization step (M-step). It is used when part of the data is not observed, and the observed likelihood function is more difficult to evaluate than the complete data likelihood function.

Like in Section 2.4.1, let \( x_{\text{obs}} \) denote the set of observed data, \( x_{\text{mis}} \) the unobserved data, and \( \theta \) the set of parameters to be estimated.

Let \( \hat{\theta}^{(0)} \) be an initial value for \( \theta \). In the first step (expectation step or E-step) the following expression is evaluated:

\[
Q(\theta, \hat{\theta}^{(0)}) = E_{x_{\text{mis}}|x_{\text{obs}}, \hat{\theta}^{(0)}}[\ell(\theta|x_{\text{obs}}, x_{\text{mis}})].
\]  

(2.27)

In the second step (maximization step or M-step), \( Q \) is maximized with respect to \( \theta \) over all possible \( \theta \in \Theta \) to find a value \( \theta^{(1)} \) such that:

\[
Q(\theta^{(1)}, \hat{\theta}^{(0)}) \geq Q(\theta, \theta^{(0)})
\]  

(2.28)

for all \( \theta \in \Theta \). The E-step and M-step are then carried out iteratively. Let \( k \) denote the iteration number. The algorithm thus iterates between the following two steps:

**E-step** :

\[
Q(\theta, \hat{\theta}^{(k)}) = E_{x_{\text{mis}}|x_{\text{obs}}, \hat{\theta}^{(k)}}[\ell(\theta|x_{\text{obs}}, x_{\text{mis}})]
\]  

(2.29)

**M-step** :

\[
\hat{\theta}^{(k+1)} = \arg \max_\theta Q(\theta, \theta^{(k)}).
\]

The iteration over these steps is continued until the difference between to consecutive estimates of the parameters is sufficiently small:

\[
\left\| \hat{\theta}^{(n+1)} - \hat{\theta}^{(n)} \right\| < \epsilon
\]

where \( \epsilon \) denotes a small quantity. Another option is to iterate until the difference between two consecutive values of the observed data likelihood is sufficiently small:

\[
|l_{\text{obs}}(\hat{\theta}^{(k+1)}) - l_{\text{obs}}(\hat{\theta}^{(k)})| < \epsilon.
\]

The EM-algorithm does not maximize the complete data likelihood \( L(\theta|x_{\text{obs}}, x_{\text{mis}}) \), but instead maximizes the observed data likelihood \( L(\theta|x_{\text{obs}}) \) by using the
complete data likelihood. In each iteration step, the observed data likelihood becomes smaller, but it is not a guarantee that it converges to a local maximum. Therefore, running the EM-algorithm with multiple starting values can be useful (Do and Batzoglou 2008).

2.4.6 Composite Likelihood

Composite likelihood methods can be applied when the likelihood function is difficult to evaluate or it is not possible to find a closed expression for it. These issues often arise when \( \theta \) is high-dimensional. Instead of maximizing the likelihood function, another function similar to the likelihood function is maximized: the composite likelihood function. If the composite likelihood function bears enough resemblance to the original likelihood function, the estimates obtained by maximizing the composite likelihood function can still be consistent, but less efficient than MLE-estimates (Cox and Reid 2004). The original likelihood function is often called the full likelihood function. Technical details about the composite likelihood can be found in Lindsay (1988) or Varin (2008).

To illustrate this method, we discuss a simple example (Lindsay, 1988). Suppose that there are two observations \( y_1 \) and \( y_2 \) which are marginally standard normal, and that they have correlation \( \rho \) to be estimated. Let \( L(\theta|y_1, y_2) \) denote the full likelihood function, where \( \theta \) is a set of parameters describing the multivariate normal distribution. Instead of maximizing the full likelihood, the following function can be maximized:

\[
L_1(\theta|y_1, y_2) = f(y_2|y_1) \ast f(y_1|y_2) \\
L_2(\theta|y_1, y_2) = f(y_1) \ast f(y_2).
\]

The law of conditional probability is tells us that \( L(\theta|y_1, y_2) = f(y_2|y_1, \theta) \ast f(y_1|\theta) \). We see that \( L_1 \) and \( L_2 \) are similar to \( L \), but they are not the same.

Fiocco et al. (2009b) used a composite likelihood procedure to estimate the model described in Section 2.4.3. The procedure consists of two steps. In the first step, the parameters \( \mu \) and \( \xi \) are estimated by maximizing a composite likelihood function based on the marginal densities. In the second step, \( \rho \) is estimated using a composite likelihood function based on the pairwise likelihood of all variables and by plugging in the estimates from the first step.
Chapter 3

Correlated Gamma Frailty Processes

The goal of this chapter is to construct a correlated gamma frailty process \((Z_1, \ldots, Z_T)\) associated with time points \((t_1, \ldots, t_T)\), first-order autoregressive correlation \(\text{corr}(Z_{t_r}, Z_{t_s}) = \rho^{|t_r - t_s|}\), identical marginal distributions \(\Gamma(\alpha, \beta)\) and explicit finite-dimensional distributions. In Section 3.3 the available literature on this subject is discussed and it is explained why new construction methods are needed. In Section 3.2 the model of Fiocco et al. (2009b) is discussed. In Sections 3.3 and 3.4 two new methods are proposed. We will refer to these three models as models 1 through 3 respectively. In Section 3.5 the three methods are compared and it is discussed which one is to be preferred.

3.1 The Multivariate Gamma Distribution

Let \(Z = (Z_1, \ldots, Z_T)\) denote a vector of \(T\) gamma distributed variables with a given covariance matrix \(\Sigma\) and shape and rate vector \(\alpha = (\alpha_1, \ldots, \alpha_T)\), \(\beta = (\beta_1, \ldots, \beta_T)\) respectively. There is no closed form of the distribution of \(Z\). Only for specific configurations of \(\Sigma, \alpha\) and \(\beta\), expressions for the distribution of \(Z\) have been found. Kotz et al. (2004) give an overview of distributions for specific configurations of the parameters. See for example Furman (2007) for recent results. The first ones to describe a multivariate gamma distribution with the properties described in the introduction of this chapter, were Henderson and Shimakura (2003). However the distribution is not explicit finite-dimensional. Fiocco et al. (2009b) proposed a new correlated gamma frailty process which has the same properties, but with explicit finite-dimensional distributions. However, as is explained in Section 3.3, this model might be unnecessarily complex and restrictive.

The frailty process of Fiocco et al. is constructed from independently distributed gamma random variables \((X_1, \ldots, X_M)\) (with \(M\) possibly infinite) with identical rate parameter \(\beta\) but different shape parameters. The frailty terms \((Z_1, \ldots, Z_T)\) are constructed by making use of the infinite divisibility property of gamma random variables. This implies that each gamma distributed random variable can be written as the (possible infinite) sum of independently distributed gamma random variables. Let \(X \overset{d}{=} \Gamma(\alpha, \beta)\), \(\sum_{k=1}^{\infty} a_k = \alpha\) and \(X_k \overset{d}{=} \Gamma(a_k, \beta)\). We can write:
\[ \sum_{k=1}^{\infty} X_k \overset{d}{=} \Gamma\left(\sum_{k=1}^{\infty} \alpha_k, \beta\right) = \Gamma(\alpha, \beta). \] (3.1)

Define for each time point \( t \) in \((1, \ldots, T)\) an index set \( I_t \subset \{1, \ldots, M\} \). The frailty terms are then given by:

\[ Z_t = \sum_{k \in I_t} X_k. \]

The correlation between two frailty terms is then induced by the number of components the two have in common and the values of the shape parameters. The number of random variables \( M \), index sets \((I_1, \ldots, I_T)\) and shape coefficients \((\alpha_1, \ldots, \alpha_M)\) then have to be defined in such a way that the resulting frailty terms \((Z_1, \ldots, Z_T)\) have marginal distribution \( \Gamma(\alpha, \beta) \) and have correlation structure \( \text{corr}(Z_r, Z_s) = \rho^{|t_r - t_s|} \). Details about this construction are outlined in Sections 3.2, 3.3 and 3.4 which are organized as follows:

1. Define a set of gamma distributed random variables \( X_1, \ldots, X_M \) with shape \( \alpha_i \) for each \( i \in \{1, \ldots, M\} \) and an index set \( I_t \) for each \( t \in \{1, \ldots, T\} \).
2. Show that the resulting frailty terms are distributed as \( \Gamma(\alpha, \beta) \).
3. Show that the correlation structure is \( \text{corr}(Z_r, Z_s) = \rho^{|t_r - t_s|} \).

### 3.2 Model 1

Model 1 (Fiocco et al., 2009b) is constructed as follows:

\[ X_{ij} \overset{d}{=} \Gamma((1 - \rho)^2 \rho^{j-i} \alpha, \beta) \] (3.2)

where \( i, j \in \mathbb{Z} \) with \( i \leq j \). Denote by \( Z_t \) the frailty terms:

\[ Z_t = \sum_{i=-\infty}^{t} \sum_{j=t}^{\infty} X_{ij} \] (3.3)

for \( t \in \{1, \ldots, T\} \). See Figure 3.1 for a visualization. An extra assumption on the frailty terms of the process is required here: the frailty terms are all one unit of time apart from each other.

We will now prove that the resulting frailty terms are marginally distributed as \( \Gamma(\alpha, \beta) \).

**Theorem 3.2.1.** The frailty terms \( Z_1, \ldots, Z_T \) as defined in Model 1 have marginal distribution \( \Gamma(\alpha, \beta) \).

**Proof.** Due to the infinite divisibility property of the Gamma distribution (3.1):
We need to prove the following:

\[ \sum_{i=-\infty}^{\infty} \sum_{j=t}^{\infty} \rho^{j-i} = \sum_{i=-\infty}^{\infty} \rho^{-i} \sum_{j=t}^{\infty} \rho^j = \frac{1}{(1-\rho)^2}. \]

We make use of some well known identities for finite and infinite geometric series:

\[ \sum_{j=t}^{\infty} \rho^j = \sum_{j=0}^{\infty} \rho^j - \sum_{j=0}^{t-1} \rho^j = \frac{1}{1-\rho} - \frac{1-\rho^t}{1-\rho} = \frac{\rho^t}{1-\rho}. \]
\[ \sum_{i=-\infty}^{t} \rho^{-i} = \sum_{i=-\infty}^{0} \rho^{-i} + \sum_{i=1}^{t} \rho^{-i} \]
\[ = \sum_{i=0}^{\infty} \rho^i + \sum_{i=0}^{t-1} (\rho^{-1})^{i+1} \]
\[ = \frac{1}{1-\rho} + \frac{1}{\rho} \sum_{i=0}^{t-1} (\rho^{-1})^i \]
\[ = \frac{1}{1-\rho} + \frac{1}{\rho} \frac{1-\rho^{-t}}{1-\rho^{-1}} \]
\[ = \frac{1}{1-\rho} - \frac{1-\rho^{-t}}{1-\rho} \]
\[ = \frac{\rho^{-t}}{1-\rho}. \]

Then:
\[ \sum_{i=-\infty}^{t} \sum_{j=1}^{\infty} (1-\rho)^2 \rho^{j-i} = (1-\rho)^2 \sum_{i=-\infty}^{t} \sum_{j=1}^{\infty} \rho^{j-i} \]
\[ = (1-\rho)^2 \sum_{i=-\infty}^{t} \rho^{-i} \sum_{j=t}^{\infty} \rho^i \]
\[ = (1-\rho)^2 \frac{\rho^{-t}}{1-\rho} \frac{\rho^t}{1-\rho} \]
\[ = 1. \]

This concludes the proof. 

We now show that the frailty terms have the desired correlation structure.

**Theorem 3.2.2.** The frailty terms \( Z_1, \ldots, Z_T \) of Model 1 have correlation structure \( \text{corr}(Z_s, Z_t) = \rho^{|s-t|} \) for all \( s, t \in \{1, \ldots, T\} \).

**Proof.** Let \( Z_{s \cap t} \) denote the sum of \( X_{ij} \)'s that \( Z_t \) and \( Z_s \) have in common. Let \( Z_{t \setminus s} \) denote the sum of the \( X_{ij} \)'s which are part of \( Z_t \) but not of \( Z_s \), and denote \( Z_{s \setminus t} \) equivalently for \( Z_s \). Figure 3.1 shows:

\[ Z_{s \cap t} = \sum_{i=-\infty}^{s} \sum_{j=t}^{\infty} X_{ij}. \]  \hspace{1cm} (3.4)

We will now prove that (3.4) is distributed as \( \Gamma(\alpha \rho^{|t-s|}, \beta) \). To prove this, we again use the infinite divisibility property (3.1):

\[ Z_{s \cap t} \sim \Gamma(\sum_{i=-\infty}^{s} \sum_{j=t}^{\infty} (1-\rho)^2 \rho^{j-i}, \alpha, \beta). \]

By using results from Theorem 3.2.1 follows:
Chapter 3. Correlated Gamma Frailty Processes

\[ \sum_{i=-\infty}^{s} \sum_{j=0}^{\infty} \rho^{j-i} = \sum_{i=-\infty}^{t} \sum_{j=0}^{\infty} \rho^{j-i} - \sum_{i=s+1}^{t} \sum_{j=0}^{\infty} \rho^{j-i} \]

\[ = \frac{1}{(1-\rho)^2} - \sum_{i=s+1}^{t} \rho^{-i} \sum_{j=0}^{\infty} \rho^{j} \]

\[ = \frac{1}{(1-\rho)^2} - \rho^{t} \frac{1}{1-\rho} \sum_{i=s+1}^{t} \rho^{-i} \sum_{j=0}^{\infty} \rho^{j} \]

\[ = \frac{1}{(1-\rho)^2} - \rho^{t-s} \frac{1}{1-\rho} \sum_{i=0}^{t-s-1} \rho^{-i} \]

\[ = \frac{1}{(1-\rho)^2} - \rho^{t-s} \frac{1}{1-\rho} \frac{1-(\rho^{-1})^{t-s}}{1-\rho^{-1}} \]

\[ = \frac{1}{(1-\rho)^2} + \frac{\rho^{t-s}-1}{(1-\rho)^2} \]

\[ = \frac{\rho^{t-s}}{(1-\rho)^2}. \]

This yields:

\[ Z_{s \cap t} \overset{d}{=} \Gamma \left( \sum_{i=-\infty}^{s} \sum_{j=0}^{\infty} (1-\rho)^2 \rho^{j-i} \alpha, \beta \right) \]

\[ = \Gamma \left( \frac{\rho^{t-s}}{(1-\rho)^2} (1-\rho)^2 \alpha, \beta \right) \]

\[ = \Gamma (\rho^{t-s} \alpha, \beta). \]

The two frailty terms \( Z_{\cap s} \) and \( Z_{s \setminus t} \) are independent by construction. The covariance of \( Z_s \) and \( Z_t \) is then:

\[ \text{Cov}(Z_s, Z_t) = \text{Cov}(Z_{s \cap t} + Z_{s \setminus t}, Z_{s \cap t} + Z_{t \setminus s}) \]

\[ = \text{Cov}(Z_{s \cap t}, Z_{s \cap t}) + \text{Cov}(Z_{s \cap t}, Z_{t \setminus s}) + \text{Cov}(Z_{s \setminus t}, Z_{s \cap t}) + \text{Cov}(Z_{s \setminus t}, Z_{t \setminus s}) \]

\[ = \text{Var}(Z_{s \cap t}). \]

Here, we used some elementary properties of the covariance. The variance of a \( \Gamma(\alpha, \beta) \) distribution (with shape-rate parametrization) is given by \( \frac{\alpha}{\beta^2} \).

The correlation of \( Z_s \) and \( Z_t \) is thus given by:

\[ \text{corr}(Z_s, Z_t) = \frac{\text{Var}(Z_{s \cap t})}{\sqrt{\text{Var}(Z_s) \text{Var}(Z_t)}} \]

\[ = \frac{\rho^{t-s} \alpha/\beta^2}{\sqrt{\alpha^2/\beta^4}} \]

\[ = \rho^{t-s}. \]

This concludes the proof.
The fact that our frailty terms are built from infinitely many gamma random variables poses a problem how to simulate data from this multivariate distribution. Fiocco et al. (2009b) proposed to collapse the terms. If the number of frailty terms $T$ is fixed, we can define:

$$X_{i+} = \sum_{j=T+1}^{\infty} X_{ij} \overset{d}{=} \Gamma(\alpha(1-\rho)^{T+1-i}, \beta) \quad i = 1, \ldots, T$$

$$X_{+j} = \sum_{i=-\infty}^{0} X_{ij} \overset{d}{=} \Gamma(\alpha(1-\rho)^{i}, \beta) \quad i = 1, \ldots, T$$

$$X_{++} = \sum_{i=-\infty}^{0} \sum_{j=T+1}^{\infty} X_{ij} \overset{d}{=} \Gamma(\alpha \rho^{T+1}, \beta)$$

and $Z_t$ for $t \in \{1, \ldots, T\}$ is given by:

$$Z_t = \sum_{i=1}^{t} X_{i+} + \sum_{j=t}^{T} X_{+j} + X_{++} + \sum_{i=1}^{T} \sum_{j=1}^{t} X_{ij}.$$  

See Figure 3.2 for a visualization of the frailty terms. For $T$ frailty terms, this means that we need to simulate $T$ variables for the $X_{i+}$'s, $T$ variables for the $X_{+j}$'s, one variable for $X_{++}$ and $\frac{1}{2}(T+1) \times T$ for the residual variables. The total number of random variables to be simulated is $\frac{1}{2}T^2 + \frac{5}{2}T + 1$.

### 3.3 Model 2

The model described in Section 3.2 can be slightly improved. Firstly, it is unnecessary to build the frailty terms from infinitely many independent random variables. In addition, the model assumes that the distance between...
the time points \((t_1, \ldots, t_T)\) corresponding to the frailty terms is equal. This assumption is not very restrictive, but nevertheless it might be unnecessary. The following model – which we will call Model 2 – solves these two issues.

Define the independent gamma random variables as follows:

\[
X_{ij} \overset{\text{d}}{=} \Gamma((\rho^{ij} - \rho^{i-1,j} - \rho^{i,j+1} + \rho^{i-1,j+1})\alpha, \beta) \tag{3.5}
\]

for \(1 \leq i, j \leq T\) and \(j \geq i\). The number of gamma random variables this model needs is equal to \(\frac{1}{2}T(T + 1)\).

Let \(\rho_{ij} = |t_j - t_i|\). The term \(\rho_{ij}\) is defined to be zero when either \(i\) or \(j\) is outside the set \(\{1, \ldots, T\}\). An equivalent definition for (3.5) is:

\[
X_{ij} \overset{\text{d}}{=} \Gamma(\rho_{ij}^* (1 - \rho^\Delta_i)(1 - \rho^\Delta_j), \alpha, \beta) \tag{3.6}
\]

where \(\Delta_i = \rho_{i,i}^*\). From (3.6) it is clear that Model 2 is a generalization of Model 1; when \(\Delta_i = 1\) for all \(i = (2, \ldots, T)\), definition (3.6) is equal to (3.2).

Define for \(t \in \{1, \ldots, T\}\) the frailty terms as follows:

\[
Z_t = \sum_{j=t}^{T} \sum_{i=1}^{t} X_{ij}. \tag{3.7}
\]

The idea behind the model is based on Venn Diagrams [Venn, 1880]. A Venn Diagram is a diagram which shows all possible overlapping segments between a finite amount of sets. The frailty terms in the model in Section 3.2 are constructed from building blocks (independent gamma random variables \(X_1, \ldots, X_M\)), where some of the building blocks are used in multiple frailty terms \(Z_1, \ldots, Z_T\). We can associate this construction to Venn Diagrams.

In Figure 3.3, a Venn Diagram is shown for three sets/frailty terms. Each circle represents a frailty term. We want the frailty terms to have the same shape, therefore the surfaces of the circles are depicted to be equal. Each frailty term is correlated with all the other frailty terms, hence the overlapping areas.

If we apply the model introduced in Section 3.2 to the Venn Diagram for three time points, we get the following configuration:

\[
\begin{align*}
A_1 &= \{X_{11}, X_{1+1}\} \\
A_2 &= \{X_{22}\} \\
A_3 &= \{X_{3+}, X_{33}\} \\
A_{12} &= \{X_{12}, X_{2+}\} \\
A_{23} &= \{X_{23}, X_{2+}\} \\
A_{13} &= \emptyset \\
A_{123} &= \{X_{++}, X_{1+}, X_{3+}, X_{13}\}.
\end{align*}
\tag{3.8}
\]

We see that the model of Section 3.2 defines an unnecessary amount of independent random variables, since it is sufficient to define a single gamma
random variable per area. We also see that the area $A_{13}$ is empty. This implies that it is unnecessary to define an area depicting the overlap between two frailty terms which are more than one time point apart. The definition $\text{corr}(Z_r, Z_s) = \rho^{|t_r - t_s|}$ implies that the correlation between two frailty terms which are more than one time point apart is always smaller than the correlation between two frailty terms which are in between. This suggests that the correlation can be described only by terms which are close.

Can we construct another type of diagram which does not have these unnecessary areas? Can we define independent gamma random variables in such a way that each area only contains one variable? The answer can be seen in Figure 3.4. Each rectangle depicts a frailty term and for each square we define a gamma random variable $X_{ij}$ with $1 \leq i, j \leq T$ and $j \geq i$.

The question is now how to define the value of the shape parameter of the gamma random variables in such a way that the correlation structure holds and the frailty terms are marginally distributed as $\Gamma(\alpha, \beta)$.

In 3.5 the correlation that each rectangle is supposed to induce is depicted in the lower right corner. This implies that the correlation that each square is supposed to induce is $\rho^{|i-j|} - \rho^{i-1,j} - \rho^{i,j+1} + \rho^{i-1,j+1}$ for $1 \leq i, j \leq T$ and $j \geq i$. The term $\rho^{|i-j|}$ is defined to be zero when either $i$ or $j$ is outside the set $\{1, \ldots, T\}$.

The first thing we now need to show, is that the $X_{ij}$'s are well defined. The Gamma Distribution is only defined for positive values of the shape parameter, and it is not defined for the value zero. For convenience however, we will define the Gamma Distribution with value of the shape parameter equal to zero, to be the zero distribution. As $\rho^{|i-j|} \leq 1$ for all $i = (1, \ldots, T + 1)$, definition (3.6) shows that the shape parameter is larger or equal to zero, and thus we can conclude that they are indeed well defined.
Figure 3.4: Four frailty terms depicted as rectangles.

Figure 3.5: Frailty terms $Z_1$, $Z_2$, $Z_3$ and $Z_4$ depicted as rectangles. The correlation a rectangle is supposed to induce is depicted in the lower right corner of each rectangle. For the correlation ($\rho_{23}$) the rectangle is colored to show what we mean by 'the correlation a rectangle is supposed to induce'.
Chapter 3. Correlated Gamma Frailty Processes

Since the $X_{ij}$’s are well defined, we now have to show that the multivariate gamma distribution has the desired marginal distributions and correlation structure. This is proven in Theorem 3.3.2 and Theorem 3.3.3 respectively. A lemma used in the proofs of both theorems is given below.

**Lemma 3.3.1.** Let $s, t \in \{1, \ldots, T\}$ with $s < t$. Then:

$$\sum_{j=t}^{T} \sum_{i=1}^{s} X_{ij} \overset{d}{=} \Gamma(\rho^{s,t} \alpha, \beta).$$

**Proof.** By using the infinite divisibility property (3.1):

$$\sum_{j=t}^{T} \sum_{i=1}^{s} X_{ij} \overset{d}{=} \Gamma((\sum_{j=t}^{s} \rho^{i,j} - \rho^{i-1,j} - \rho^{i,j+1} + \rho^{i-1,j+1}) \alpha, \beta).$$

It suffices to prove that:

$$\sum_{j=t}^{s} \sum_{i=1}^{s} \rho^{i,j} - \rho^{i-1,j} - \rho^{i,j+1} + \rho^{i-1,j+1} = \rho^{s,t}.$$

Rewrite the left-hand side as:

$$= \sum_{j=t}^{T} \sum_{i=1}^{s} \rho^{i,j} - \sum_{j=t}^{s-1} \sum_{i=1}^{s} \rho^{i,j} - \sum_{j=t}^{T} \sum_{i=1}^{s} \rho^{i,j+1} + \sum_{j=t}^{T} \sum_{i=1}^{s} \rho^{i-1,j+1}$$

$$= \sum_{j=t}^{T} \sum_{i=1}^{s} \rho^{i,j} - \sum_{j=t}^{s-1} \sum_{i=1}^{s} \rho^{i,j} - \sum_{j=t+1}^{T} \sum_{i=1}^{s} \rho^{i,j} + \sum_{j=t+1}^{T} \sum_{i=1}^{s} \rho^{i,j+1}$$

$$= \sum_{j=t}^{T} \sum_{i=1}^{s} \rho^{i,j} - \sum_{j=t+1}^{T} \sum_{i=1}^{s} \rho^{i,j}$$

$$= \rho^{s,t}.$$

This concludes the proof. □

**Theorem 3.3.2.** The frailty terms $Z_{1}, \ldots, Z_{T}$ as defined by Model 2 have marginal distribution $\Gamma(\alpha, \beta)$.

**Proof.** Let $t \in \{1, \ldots, T\}$. Then $Z_{t}$ is defined as

$$Z_{t} = \sum_{j=t}^{T} \sum_{i=1}^{t} X_{ij}.$$

According to Lemma 3.3.1, $Z_{t} \overset{d}{=} \Gamma(\rho^{t,t} \alpha, \beta)$. As $\rho^{t,t} = 1$, this concludes the proof. □
Chapter 3. Correlated Gamma Frailty Processes

Theorem 3.3.3. The frailty terms $Z_1, \ldots, Z_T$ as defined in Model 2 have correlation structure $\text{corr}(Z_s, Z_t) = \rho^{t,s}$ for all $t, s \in \{1, \ldots, T\}$.

Proof. Let $s, t \in \{1, \ldots, T\}$ with $s < t$. Let $Z_{s \cap t}$ denote the sum of the $X_{ij}$'s with common terms in $Z_t$ and $Z_s$. Let $Z_{t \setminus s}$ denote the sum of the $X_{ij}$'s which are part of $Z_t$ but not of $Z_s$, and in the same way define $Z_{s \setminus t}$. In Figure 3.4 the three frailty terms are shown:

$$Z_{s \cap t} = \sum_{j=1}^{T} \sum_{i=1}^{s} X_{ij}$$

$$Z_{t \setminus s} = \sum_{j=1}^{t-1} \sum_{i=1}^{s} X_{ij} \quad (3.9)$$

$$Z_{s \setminus t} = \sum_{j=1}^{T} \sum_{i=s+1}^{t} X_{ij}.$$

According to Lemma 3.3.1 $Z_{s \cap t} \overset{d}{=} \Gamma(\rho^{s,t}, \alpha, \beta)$. We will now derive the distribution of $Z_{t \setminus s}$ and $Z_{s \setminus t}$. By applying the infinite divisibility property (3.1):

$$\sum_{j=s}^{t-1} \sum_{i=1}^{s} X_{ij} \overset{d}{=} \Gamma((\sum_{j=s}^{t-1} \sum_{i=1}^{s} \rho^{i-j} - \rho^{i-1,j} - \rho^{i,j+1} + \rho^{i-1,j+1})\alpha, \beta).$$

The sum can be rewritten as:

$$= \sum_{j=s}^{t-1} \sum_{i=1}^{s} \rho^{i-j} - \sum_{j=s}^{t-1} \sum_{i=1}^{s} \rho^{i-1,j} + \sum_{j=s}^{t-1} \sum_{i=1}^{s} \rho^{i,j+1}$$

$$= \sum_{j=s}^{t-1} \sum_{i=1}^{s} \rho^{i-j} - \sum_{j=s}^{t-1} \sum_{i=1}^{s} \rho^{i-1,j} - \sum_{j=s+1}^{t} \sum_{i=1}^{s} \rho^{i,j} + \sum_{j=s+1}^{t} \sum_{i=1}^{s} \rho^{i,j} - \sum_{j=s+1}^{t} \sum_{i=1}^{s} \rho^{i,j} - \sum_{j=s+1}^{t} \sum_{i=1}^{s} \rho^{i,j}$$

$$= \sum_{j=s}^{t-1} \rho^{i-j} - \sum_{j=s+1}^{t} \rho^{i-j}$$

$$= \rho^{s,t} - \rho^{s,t}$$

$$= 1 - \rho^{s,t}.$$

This shows that $Z_{s \setminus t} \overset{d}{=} \Gamma((1 - \rho^{s,t})\alpha, \beta)$. In the same way, we can prove that $Z_{t \setminus s} \overset{d}{=} \Gamma((1 - \rho^{s,t})\alpha, \beta)$. The situation is similar to the one in Theorem 3.2.2. Since $Z_{s \setminus t}$, $Z_{t \setminus s}$ and $Z_{s \cap t}$ are independent by construction, $\text{Cov}(Z_s, Z_t) = \text{Var}(Z_{s \cap t})$. By using some elementary properties of the gamma distribution, follows:
Chapter 3. Correlated Gamma Frailty Processes

\[
\text{corr}(Z_s, Z_t) = \frac{\text{Var}(Z_{s,t})}{\sqrt{\text{Var}(Z_s)\text{Var}(Z_t)}} = \frac{\rho^{t_s,t_s}/\beta^2}{\sqrt{\alpha^2/\beta^4}} = \rho^{t_s},
\]

This concludes the proof. \qed

### 3.4 Model 3

So far, we constructed a multivariate gamma distribution by using the infinite divisibility property of gamma random variables. This produced two models where the number of gamma random variables necessary were quadratic as a function of the number of time points \(T\). This leads to the following question: is it possible to construct a model where only a linear amount of independent random variables is required to construct the multivariate gamma distribution?

Define

\[
X_i \overset{d}{=} \Gamma(\alpha, \beta) \quad i = 1
\]

\[
X_i \overset{d}{=} \Gamma([1 - \rho^{t_i-t_i-1}]\alpha, \beta) \quad i \in \{2, \ldots, T\}
\]

\[
B_i \overset{d}{=} B(\rho^{t_i+1-t_i}\alpha, (1 - \rho^{t_i+1-t_i})\alpha) \quad i \in \{1, \ldots, T - 1\}.
\]

The number of independent gamma and beta random variables the model needs is equal to \(2T - 1\). Define the frailty terms as follows

\[
Z_k = X_1 \quad k = 1
\]

\[
Z_k = X_k + B_{k-1} * Z_{k-1} \quad k \in \{2, \ldots, T\}
\]

\[
= X_k + \sum_{i=1}^{k-1} (\prod_{m=k} B_m) X_i.
\]

Figure 3.6 illustrates this model. Under conditions specified in Lemma 3.4.1, the product of a beta and a gamma distributed variable is again distributed as a gamma distribution. We begin by assigning \(X_1 \sim \Gamma(\alpha, \beta)\) to \(Z_1\), which has the desired marginal distribution. Then we multiply \(X_1\) with \(B_1\), and assign the resulting gamma distributed random variable \(B_1X_1\) to \(Z_2\). If we assign the right shape parameters to the beta distribution, this term will produce the desired correlation. We then construct a new gamma distribution \(X_2\) and add that to \(B_1X_1\) to produce the desired marginal distribution. We continue this process until all frailty terms \(Z_1, \ldots, Z_T\) are defined.
Lemma 3.4.1. If
\[ X \overset{d}{=} \Gamma(\alpha^*, \beta^*) \]
\[ B \overset{d}{=} B(c\alpha^*, (1-c)\alpha^*) \]
with \(0 < c < 1\), then
\[ B \ast X \overset{d}{=} \Gamma(\rho |t_{m+1} - t_m|, \alpha) \]


We will now use this lemma to prove that the frailty terms have the desired marginal distributions and correlation structure.

Theorem 3.4.2. For each \(k \in \{1, \ldots, T\}\) the frailty terms as defined by the beta-gamma model have marginal distribution \(Z_k \overset{d}{=} \Gamma(\alpha, \beta)\).

Proof. We will use a proof by induction.

Base case
Let \(k = 1\). Then \(Z_1 = X_1\) and thus we know that \(Z_1 \overset{d}{=} \Gamma(\alpha, \beta)\).

The inductive hypothesis
Suppose that \(Z_k \overset{d}{=} \Gamma(\alpha, \beta)\) for all \(k\) up to some \(m \in \{1, \ldots, T - 1\}\).

The inductive step
Let \(k = m + 1\). Then we know that \(Z_{m+1} = X_{m+1} + B_m \ast Z_m\). By applying Lemma 3.4.1,
\[ B_m \ast Z_m \overset{d}{=} \Gamma(\rho |t_{m+1} - t_m|, \alpha, \beta) \]
\[ X_{m+1} \overset{d}{=} \Gamma(1 - \rho |t_{m+1} - t_m|, \alpha, \beta) \].
By applying the infinite divisibility property of the gamma random variable \( Z_{m+1} \overset{d}{=} \Gamma(\alpha, \beta) \).

Based on induction, the theorem holds for all \( k \in \{1, \ldots, T\} \).

\[ \text{Lemma 3.4.3.} \quad \text{The correlation structure for } m \in \{1, \ldots, T-1\} \text{ and the frailty terms } Z_1, \ldots, Z_T \text{ as defined by Model 3 is as follows:} \]

\[ \text{corr}(Z_m, Z_{m+1}) = \rho^{m+1},m. \]

\[ \text{Proof.} \quad \text{The correlation between } Z_m \text{ and } Z_{m+1} \text{ is as follows:} \]

\[ \text{corr}(Z_n, Z_m) = \frac{\text{Cov}(Z_n, Z_m)}{\sqrt{\text{Var}(Z_n) \text{Var}(Z_m)}}. \]

By lemma 3.4.1 \( \text{Var}(Z_{n+1}) = \text{Var}(Z_n) = \frac{\alpha}{\beta^2} \). The denominator is thus \( \frac{\alpha}{\beta^2} \).

The definition of the covariance is given as follows:

\[ \text{Cov}(Z_{m+1}, Z_m) = \mathbb{E}[Z_{m+1}Z_m] - \mathbb{E}[Z_{m+1}]\mathbb{E}[Z_m]. \]

Recall that \( \mathbb{E}[Z_{m+1}] = \mathbb{E}[Z_m] = \frac{\alpha}{\beta} \), thus \( \mathbb{E}[Z_{m+1}]\mathbb{E}[Z_m] = \frac{\alpha^2}{\beta^2} \). The term \( \mathbb{E}[Z_{m+1}Z_m] \) is given as follows:

\[ \mathbb{E}[Z_{m+1}Z_m] = \mathbb{E}[Z_m(B_mZ_m + X_{m+1})] = \mathbb{E}[B_m] \mathbb{E}[Z^2_m] + \mathbb{E}[X_{m+1}] \mathbb{E}[Z_m] = \rho^{m+1,m} \frac{\alpha^2}{\beta^2} + \frac{\alpha^2}{\beta^2}(1 - \rho^{m+1,m}) \]

\[ = \frac{\rho^{m+1,m} \alpha + \alpha^2}{\beta^2}. \]

Finally the correlation is given as:

\[ \text{corr}(Z_{m+1}, Z_m) = \frac{\frac{\rho^{m+1,m} \alpha + \alpha^2}{\beta^2} - \frac{\alpha^2}{\beta^2}}{\alpha/\beta^2} = \rho^{m+1,m}. \]

This concludes the proof.

\[ \text{Lemma 3.4.4.} \quad \text{If } \text{corr}(Z_n, Z_m) = \rho^{n,m} \text{ with } Z_n, Z_m \overset{d}{=} \Gamma(\alpha, \beta), \text{ the following identity holds:} \]

\[ \mathbb{E}[Z_nZ_m] = \frac{\alpha \rho^{n,m} + \alpha^2}{\beta^2}. \]

\[ \text{Proof.} \quad \text{By using the definition of the correlation and covariance, we can write:} \]
\[ \rho^{n,m} = \text{corr}(Z_m, Z_n) = \frac{\text{Cov}(Z_m, Z_n)}{\sqrt{\text{Var}(Z_m) \text{Var}(Z_n)}} = \frac{\mathbb{E}[Z_n Z_m] - \mathbb{E}[Z_n] \mathbb{E}[Z_m]}{\sqrt{\alpha / \beta^2} \ast \alpha / \beta^2} = \frac{\mathbb{E}[Z_n Z_m] - \frac{\alpha^2}{\beta^2}}{\alpha / \beta^2} \cdot (3.11) \]

By isolating the term \( \mathbb{E}[Z_n Z_m] \), the desired result follows. This concludes the proof. \( \square \)

**Theorem 3.4.5.** The frailty terms \( Z_1, \ldots, Z_T \) as defined by Model 3 have correlation structure

\[ \text{corr}(Z_s, Z_t) = \rho^{s,t} \]

for every \( s, t \in \{1, \ldots, T\} \).

**Proof.** This proof is based on induction.

*Rephrasing of theorem*

For each \( k \in \{2, \ldots, T\} \) the following correlation structure holds for all \( i, j \leq k \): \( \text{corr}(Z_i, Z_j) = \rho^{i,j} \).

*Base case*

Let \( k = 2 \). Based on Lemma 3.4.3 \( \text{corr}(Z_1, Z_2) = \rho^{1,2} \).

*The inductive hypothesis*

Suppose that the theorem holds for all \( k \) up to some \( m \in \{3, \ldots, T - 1\} \).

*The inductive step*

Let \( k = m + 1 \). We need to prove that for all \( i, j \leq m + 1 \) the correlation structure holds. By the induction hypothesis, we know that the correlation structure holds for all \( i, j \leq m \). Lemma 3.4.3 guarantees that for the terms \( Z_m \) and \( Z_{m+1} \) the same correlation structure. The only thing left to prove is that the same correlation structure holds for \( j = m + 1 \) and \( i < m \).

By the same reasoning as in Lemma 3.4.3 we can write:

\[ \text{corr}(Z_i, Z_{m+1}) = \frac{\text{Cov}(Z_i, Z_{m+1})}{\sqrt{\text{Var}(Z_i) \text{Var}(Z_{m+1})}} = \frac{\mathbb{E}[Z_i Z_{m+1}] - \mathbb{E}[Z_i] \mathbb{E}[Z_{m+1}]}{\sqrt{\alpha / \beta^2} \ast \alpha / \beta^2} = \frac{\mathbb{E}[Z_i Z_{m+1}] - \frac{\alpha^2}{\beta^2}}{\alpha / \beta^2} \cdot (3.11) \]

We can rewrite \( \mathbb{E}[Z_i Z_{m+1}] \) as:
Chapter 3. Correlated Gamma Frailty Processes

\[ \mathbb{E}[Z_{m+1}, Z_i] = \mathbb{E}[(X_{m+1} + B_m Z_m) Z_i] \]
\[ = \mathbb{E}[X_{m+1}] \mathbb{E}[Z_i] + \mathbb{E}[B_m] \mathbb{E}[Z_m Z_i] \]
\[ = \frac{\alpha^2}{\beta^2} (1 - \rho^{m+1,m}) + \rho^{m+1,m} \frac{\alpha \rho^{m,i} + \alpha^2}{\beta^2} \]
\[ = \frac{\alpha \rho^{m+1,i} + \alpha^2}{\beta^2}. \]

Here, we applied Lemma 3.4.4 on \( \mathbb{E}[Z_m Z_i] \). By substituting this expression in (3.11), we get the desired result.

Based on induction, the theorem holds for all \( k \in \{1, \ldots, T\} \).

\[ \square \]

3.5 Comparison Between Models

Results of the previous three sections are summarized in Table 3.1. We can conclude that Model 2 is more general than Model 1: it is not necessary to assume equal distance between time points. Table 3.1 suggests that Model 2 uses less random variables to construct the frailty process than Model 1. This is actually not true. If the gamma random variables of Model 1 are collapsed in the right way, it would also result in a model which uses \( \frac{1}{2} T^2 + \frac{5}{2} T + 1 \) random variables to construct the process. We chose to describe Model 1 in the form in which it was published in Fiocco et al. (2009a). Model 2 is thus actually a generalization of Model 1.

It is difficult to compare Model 2 and Model 3. The latter needs only a linear amount of random variables, but on the other hand, some of the components have a beta-distribution. We will see later in chapter 7 that this will pose a problem some applications.

<table>
<thead>
<tr>
<th>Model</th>
<th>Equal distance between time points assumed?</th>
<th>Number of random variables used to construct the frailty terms as a function of the number of time points T</th>
<th>Type of distribution of the random variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 1</td>
<td>yes</td>
<td>( \frac{1}{2} T^2 + \frac{5}{2} T + 1 )</td>
<td>Gamma</td>
</tr>
<tr>
<td>Model 2</td>
<td>no</td>
<td>( \frac{1}{2} T^2 + \frac{5}{2} T )</td>
<td>Gamma</td>
</tr>
<tr>
<td>Model 3</td>
<td>no</td>
<td>( 2T - 1 )</td>
<td>Beta and Gamma</td>
</tr>
</tbody>
</table>

Table 3.1: Summary of models discussed
Chapter 4

EM Algorithm

The goal of this chapter is to investigate whether it is computationally feasible to apply a combination of the profile likelihood method (Section 2.4.4) and the EM-algorithm (Section 2.4.5) to estimate the parameters of the model described in Section 2.4.3. Model 2 (Section 3.3) is used to model the frailty process. In Section 4.1, it is explained how the combination of these two methods works. In Section 4.2, algebraic expressions to execute the algorithm are derived. In Section 4.3, the R-code to implement the algorithm for two time points is discussed. We start with two time points ($T = 2$), because the EM-algorithm is generally known to be computationally demanding, and we want to assess computation time of the algorithm before putting effort into generalizing it for higher time points. In Section 4.4, a simulation study is carried out to assess performance of the algorithm compared to the method of Fiocco et al. (2009b). In Section 4.5, conclusions are drawn on the performance of the algorithm.

4.1 Introduction

The notation used in this section is introduced in Section 2.4.3. The goal is to maximize the observed data likelihood $\ell_{\text{obs}}(\mu, \xi, \rho | Y)$ (see (2.22)) in order to find the estimates $(\hat{\mu}, \hat{\xi}, \hat{\rho})$ based on the data $Y$. However, as this is a complex function to maximize, we use the profile likelihood method (Section 2.4.4). The parameters are partitioned into two sets: $(\mu)$ and $(\xi, \rho)$. We profile out the set $(\mu)$ and maximize:

$$\ell_{\text{prof}}(\rho, \xi | Y) = \ell_{\text{obs}}(\xi, \rho, \hat{\mu}(\xi, \rho) | Y).$$

(4.1)

In the subscript, it is suppressed that this is actually the observed profile likelihood function. The function $\hat{\mu}(\xi, \rho)$ is analytically intractable. Therefore, we make use of the following procedure based on Nielsen et al. (1992). To begin, we fix $\rho$ and $\xi$. The goal is to evaluate:

$$\hat{\mu}(\xi, \rho) = \arg \max_\mu \ell_{\text{obs}}(\mu, \xi, \rho | Y).$$

(4.2)

However, as the observed data likelihood function is still difficult to evaluate (even after partitioning the parameter set in two), we apply the EM-algorithm to evaluate (4.2), as the complete data likelihood $\ell(\mu, \xi, \rho | Y, Z)$ is
Chapter 4. EM Algorithm

39

easier to deal with. Let $k$ denote the iteration step of the EM-algorithm. We specify that the EM-algorithm terminates when the difference between two consecutive estimates for $\mu$ is less than some predetermined constant $\epsilon_{\mu}$:

$$\sum_{m=1}^{T} |\mu_m^{(k+1)} - \mu_m^{(k)}| \leq \epsilon_{\mu}$$

and the difference between two consecutive values of the observed log likelihood is less than some predetermined constant $\epsilon_{\text{lik}}$:

$$|\ell_{\text{obs}}(\rho, \xi, \hat{\mu}^{(k+1)}) - \ell_{\text{obs}}(\rho, \xi, \hat{\mu}^{(k)})| \leq \epsilon_{\text{lik}}. \tag{4.4}$$

The EM-algorithm needs initial estimates for $\mu$. The closer these initial estimates are to the optimal value, the faster the EM-algorithm converges. The following initial values for $\mu$ are chosen:

$$\mu_i^{(0)} = \frac{1}{n} \sum_{j=1}^{n} Y_{ij} \quad i = (1, \ldots, T), \tag{4.5}$$

which is an unbiased estimate (see Lemma 5.2.1). Based on $\hat{\mu}(\rho, \xi)$ (output of EM-algorithm), the profile log likelihood can be computed:

$$\ell_{\text{prof}}(\rho, \xi) = \ell_{\text{obs}}(\rho, \xi, \hat{\mu}(\rho, \xi)). \tag{4.6}$$

This procedure is repeated for new values of $(\rho, \xi)$ until maximum likelihood estimators $(\hat{\rho}, \hat{\xi})$ are found which maximize $\ell_{\text{prof}}(\rho, \xi|Y)$. This can be done in $\mathbb{R}$ with standard optimization functions. When the EM-algorithm is not executed for the first time, the initial estimates for $\mu$ are chosen not by evaluating (4.5), but by taking $\hat{\mu}(\rho, \xi)$ of the previous values of $(\rho, \xi)$. This proved more efficient in practice than starting over with (4.5) every time the EM-algorithm is executed.

To summarize, we use the following procedure:

**Input:**
- A $(N \times T)$ matrix $Y$
- Two parameter $\epsilon_{\mu}$ and $\epsilon_{\text{lik}}$ indicating when the algorithm has to terminate
- Fixed values for $\xi$ and $\rho$

**Step 1:** Calculate $\hat{\mu}(\rho, \xi)$ by iterating the next two steps over $k$, starting at $k = 0$, until both (4.3) and (4.4) hold:

a) Evaluate $\hat{Z}^{(k)} = \mathbb{E}[Z|Y, \xi, \rho, \hat{\mu}^{(k)}]$ (E-step)

b) Evaluate $\hat{\mu}^{(k+1)} = \arg\max_{\mu} \mathcal{L}(\mu, \xi, \rho|Y, \hat{Z}^{(k)})$ (M-step)

**Step 2:** Calculate $\ell_{\text{prof}}(\rho, \xi) = \ell_{\text{obs}}(\rho, \xi, \hat{\mu}(\rho, \xi))$
Step 3: Choose new values for $\rho$ and $\xi$ and repeat steps 1 through 2 until values are found which maximize $\ell_{\text{prof}}(\rho, \xi)$

Output: Maximum likelihood estimates for $\mu$, $\xi$, $\rho$ and $Z$

Standard errors for $(\mu, \rho, \theta)$ can be estimated by using a parametric bootstrap approach. Let $(\hat{\mu}, \hat{\theta}, \hat{\rho})$ be the estimated parameters from the original data. For one bootstrap data set, we proceed as follows:

1. Generate $(z_{it}^*)$ with $i = (1, \ldots, N)$ and $t = (1, \ldots, T)$ from a multivariate gamma distribution with parameters $\theta = \hat{\theta}$ and $\rho = \hat{\rho}$.

2. Generate $(y_{it}^*)$ with $i = (1, \ldots, N)$ and $t = (1, \ldots, T)$ from $y_{it}^* \sim \text{Pois}(\hat{\mu}_{it}z_{it}^*)$.

3. Estimate $(\mu, \theta, \rho)$ from the bootstrap data set $y^*$ by using the EM-algorithm, obtaining $(\hat{\mu}^*, \hat{\theta}^*, \hat{\rho}^*)$.

Let $N_{\text{boot}}$ be the number of bootstrap simulations giving the estimates $(\hat{\mu}^*, \hat{\theta}^*, \hat{\rho}^*)$ for each bootstrap iteration. These estimates can then be used to obtain standard errors or confidence intervals using standard procedures (Wehrens et al., 2000).

4.2 Algebraic Expressions

The goal of this section is to derive the algebraic expressions for the functions described in Section 4.1. Algebraic expressions for the complete data likelihood, observed data likelihood, and the expected value of the frailty terms conditional on the parameters are derived in Sections 4.2.1, 4.2.2, and 4.2.3 respectively.

The algebraic expressions are quite cumbersome; therefore we introduce some notation. In Section 3.3, the gamma random variables have two subscripts: $X_{ij}$ for $1 \leq i, j \leq T$ and $j \geq i$. We will denote these random variables by $(X_{11}, \ldots, X_M)$, where $M = \frac{1}{2}T(T+1)$. Also, we will use the notation $\Gamma(\alpha_m, \beta)$ for $m = (1, \ldots, M)$ to denote the distribution of these variables instead of using (3.5). We will see that it is not necessary to specify beforehand how the $X_{ij}$’s correspond to the $X_m$’s. We will use the index set $I_t$ for $t \in \{1, \ldots, T\}$ to denote those random variables which are part of $Z_t$. Thus $Z_t = \sum_{k \in I_t} X_k$. We also postpone substituting $\alpha = \beta = \xi^{-1}$.

Furthermore, in Sections 4.2.1 and 4.2.2 we will derive the contribution to the likelihood of a single cluster $(y_{11}, \ldots, y_{1T}, x_{11}, \ldots, x_M)$ instead of deriving the full likelihood. The full likelihood can then be obtained by taking the product of the likelihoods of all clusters. This enables us to suppress the index indicating the cluster.

Moreover, we do not derive expressions for the special cases:

1. $\rho = 0$: there is no correlation between the frailty terms;

2. $\rho = 1$: the frailty terms are constant over time;
3. $\theta = \infty$: the frailty terms are all constant and equal to 1.

In the first and second case, the model reduces to a frailty model where the frailty terms are not correlated. In the third case, the model reduces to a model in which no frailty terms are present. In the numerical maximization procedure, we do not allow these three values to be chosen by the maximizer. This has little effect on the performance of the method in practical applications. For example, it does not matter whether we know that $\rho$ is exactly zero, it is enough to know that $\rho$ is very close to zero.

We use of the following theorem and lemma:

**Theorem 4.2.1 (Multinomial Theorem).** For any positive integer $m$, non-negative integer $n$ and real numbers $(x_1, \ldots, x_m)$:

$$(x_1 + \cdots + x_m)^n = \sum_{k_1 + \cdots + k_m = n} \binom{n}{k_1, \ldots, k_m} \prod_{1 \leq t \leq m} x_t^{k_t}. $$

The sum is taken over all combinations of non-negative integer indexes $k_1$ through $k_m$ such that the sum of all $k_i$ is $n$. The term between the parentheses is called the multinomial coefficient and is defined as:

$$\binom{n}{k_1, \ldots, k_m} = \frac{n!}{k_1! \cdots k_m!}.$$

**Lemma 4.2.2.** If $\gamma, \delta > 0$:

$$\int_0^\infty x^{\gamma-1} e^{-x\delta} = \frac{\Gamma(\gamma)}{\delta^\gamma}.$$ 

**Proof.** The probability density function of a gamma distributed random variable with shape $\gamma$ and rate $\delta$ is:

$$f(x; \gamma, \delta) = \frac{\delta^\gamma x^{\gamma-1} e^{-x\gamma}}{\Gamma(\gamma)}.$$

As $f$ is a probability density function, the integral of $f$ over the entire domain $(0, \infty)$ of $x$ is one. Therefore:

$$\int_0^\infty \frac{\delta^\gamma x^{\gamma-1} e^{-x\gamma}}{\Gamma(\gamma)} = 1 $$

$$\delta^\gamma \int_0^\infty x^{\gamma-1} e^{-x\gamma} = 1 $$

$$\int_0^\infty x^{\gamma-1} e^{-x\gamma} = \frac{\Gamma(\gamma)}{\delta^\gamma}. $$

$\square$
4.2.1 Complete Likelihood

The goal of this section is to find an expression for the contribution to the complete data likelihood \( \mathcal{L}(\mu, \rho, \xi | Y, Z) \) from a single cluster:

\[
\mathcal{L}(\mu, \rho, \xi | y_1, \ldots, y_T, x_1, \ldots, x_M)
\]

which we will denote by \( \mathcal{L} \). The result is Theorem 4.2.3.

**Theorem 4.2.3.** The contribution to the complete data likelihood from a single cluster of data \((y_1, \ldots, y_T, x_1, \ldots, x_M)\) is:

\[
\mathcal{L} = \prod_{t=1}^{T} e^{-\mu_t z_t} \frac{(\mu_t z_t)^y_t}{y_t!} \prod_{m=1}^{M} \beta_m x_m^{\alpha_m - 1} e^{-x_m \beta} \frac{1}{\Gamma(\alpha_m)}. \tag{4.7}
\]

**Proof.** The event counts \((Y_1, \ldots, Y_N)\) are, conditional on the frailty terms, distributed as independent Poisson random variables. Also, the gamma random variables \((X_1, \ldots, X_M)\) are independent by construction (Section 3.3). When the expression for the complete data likelihood (2.21) is combined with the expression for the distribution of the gamma random variables (3.5), we obtain:

\[
\mathcal{L} = \prod_{t=1}^{T} P(y_t | \sum_{k \in I_t} x_k) \prod_{m=1}^{M} P(x_m) \\
= \prod_{t=1}^{T} P(y_t | z_t) \prod_{m=1}^{M} P(x_m) \\
= \prod_{t=1}^{T} P_{\text{pois}}(y_t; \mu_t z_t) \prod_{m=1}^{M} \Gamma(x_m; \alpha_m, \beta) \\
= \prod_{t=1}^{T} e^{-\mu_t z_t} \frac{(\mu_t z_t)^y_t}{y_t!} \prod_{m=1}^{M} \beta_m x_m^{\alpha_m - 1} e^{-x_m \beta} \frac{1}{\Gamma(\alpha_m)}. \tag{4.8}
\]

The corresponding log complete likelihood is:

\[
\ell = \sum_{t=1}^{T} -\mu_t z_t + y_t \log(\mu_t z_t) - \log(y_t!) \\
+ \sum_{m=1}^{M} \alpha_m \log(\beta) + (\alpha_m - 1) \log(x_m) - x_m \beta - \log(\Gamma(\alpha_m)). \tag{4.9}
\]

4.2.2 Observed Likelihood

The goal of this section is to find an expression for the contribution to the observed data likelihood \( \mathcal{L}_{\text{obs}}(\mu, \xi, \rho | Y) \) of a single cluster:

\[
\mathcal{L}_{\text{obs}}(\mu, \xi, \rho | y_1, \ldots, y_T)
\]
which we will denote by $L_{\text{obs}}$. The result is Theorem 4.2.4. The idea is that we can find the observed likelihood by integrating out the gamma random variables:

$$L_{\text{obs}} = \int_{x_1=0}^{\infty} \cdots \int_{x_M=0}^{\infty} L(\mu, \xi, \rho | y_1, \ldots, y_T, x_1, \ldots, x_M) \, df(x_1), \ldots, df(x_M).$$  \hspace{1cm} (4.10)

**Theorem 4.2.4.** The contribution to the observed data likelihood from a single cluster $(y_1, \ldots, y_T)$ is:

$$L_{\text{obs}} = \sum_{1 \leq j \leq N_r} \int_{k_{j,l'}, y_{l'}} \prod_{t=1}^{T} \left[ \left( \frac{y_t}{\mu^t_{y_t}} \right) \cdot \frac{\Gamma(a_m + \sum_{l'\neq \nu} k_{l',r})}{\beta + \sum_{l'\neq \nu} \mu_{l'}} \right] \cdot \beta^a_m \left( \frac{\beta^a_m}{\Gamma(a_m)} \right).$$  \hspace{1cm} (4.11)

$n_t$ denotes the number of elements in $I_t$, thus $n_t = \#I_t$. $l_{j,t}$ denotes the $j$'th element of $I_t$. Furthermore:

$$\sum_{1 \leq j \leq N_r} \int_{k_{j,l'}, y_{l'}} \prod_{t=1}^{T} \left[ \left( \frac{y_t}{\mu^t_{y_t}} \right) \cdot \frac{\Gamma(a_m + \sum_{l'\neq \nu} k_{l',r})}{\beta + \sum_{l'\neq \nu} \mu_{l'}} \right] \cdot \beta^a_m \left( \frac{\beta^a_m}{\Gamma(a_m)} \right) = C.$$  \hspace{1cm} (4.12)

The first two product terms are constants when integrating out the $x_m$'s. Therefore, we substituted these in $C$. If we want to further isolate the $x_m$'s, we can use Theorem 4.2.1 to evaluate $z_t^{y_t}$:

$$\left( \sum_{k \in I_t} x_k \right)^{y_t} = \sum_{k_{l_{j,1}}, \ldots, k_{l_{j,t}}} \left( \sum_{k_{l_{j,1}}, \ldots, k_{l_{j,t}}} \prod_{t=1}^{T} \left[ \left( \frac{y_t}{\mu^t_{y_t}} \right) \cdot \frac{\Gamma(a_m + \sum_{l'\neq \nu} k_{l',r})}{\beta + \sum_{l'\neq \nu} \mu_{l'}} \right] \cdot \beta^a_m \left( \frac{\beta^a_m}{\Gamma(a_m)} \right) \right).$$  \hspace{1cm} (4.13)

If we substitute (4.13) expression into $\prod_{t=1}^{T} z_t^{y_t}$ and rearrange the terms, we obtain:

$$\prod_{t=1}^{T} z_t^{y_t} = \sum_{1 \leq j \leq N_r} \int_{k_{j,l'}, y_{l'}} \prod_{t=1}^{T} \left[ \left( \frac{y_t}{\mu^t_{y_t}} \right) \cdot \frac{\Gamma(a_m + \sum_{l'\neq \nu} k_{l',r})}{\beta + \sum_{l'\neq \nu} \mu_{l'}} \right] \cdot \beta^a_m \left( \frac{\beta^a_m}{\Gamma(a_m)} \right).$$  \hspace{1cm} (4.14)
And if we substitute (4.14) expression in (4.12):

\[ L = C \sum_{1 \leq i \leq n} k_{i_j} = y_{i_j} | r' \in T \prod_{t=1}^{T} \left[ \left( k_{l_1, \ldots, k_{l_t}} \right) \right] \]

\[ \times \prod_{m=1}^{M} \left[ e^{-x_m (\beta + \sum_{j=m}^{\infty} \mu_j)} x_m^{-\alpha_m - 1 + \sum_{j=m}^{\infty} k_{j_r}} \right]. \]  

(4.15)

The \( x_m \)'s are now isolated. Therefore, we can now apply Lemma 4.2.2 on (4.15):

\[ L_{\text{obs}} = \int_{x_1=0}^{\infty} \cdots \int_{x_M=0}^{\infty} L \]

\[ = C \sum_{1 \leq i \leq n} k_{i_j} = y_{i_j} | r' \in T \prod_{t=1}^{T} \left[ \left( k_{l_1, \ldots, k_{l_t}} \right) \right] \]

\[ \times \prod_{m=1}^{M} \left[ \Gamma(\alpha_m + \sum_{j=m}^{\infty} k_{j_r}) / (\beta + \sum_{j=m}^{\infty} \mu_j)^{\alpha_m + \sum_{j=m}^{\infty} k_{j_r}} \right]. \]  

(4.16)

By resubstituting \( C \) and rearranging the terms in (4.16), we get the desired result.

4.2.3 E-step and M-step

The goal of this section is to find the algebraic expressions necessary to execute the EM-algorithm. In this section, we assume that \( \rho \) and \( \xi \) are fixed. For the \( k \)th iteration step, the E-step is:

\[ Q(\mu, \hat{\mu}^{(k)}) = \mathbb{E}_{Z(\cdot) | Y, \hat{\mu}^{(k)}}[\ell_{\text{obs}}(\hat{\mu}^{(k)} | Y, Z)]. \]  

(4.17)

Let \( \mathbb{E}^{[\cdot, \cdots]} \) denote the operator \( \mathbb{E}_{Z(\cdot) | Y, \hat{\mu}^{(k)}}[\cdot, \cdots] \). The expected value is a linear operator. When we apply this property and combine expressions (4.9) and (4.17):

\[
Q(\mu, \hat{\mu}^{(k)}) = \sum_{t=1}^{T} -\hat{\mu}_t^{(k)} \mathbb{E}^{(k)}[z_t] + y_t \mathbb{E}^{(k)}[\log(\hat{\mu}_t^{(k)} z_t)] - \log(y_t!) \\
+ \sum_{m=1}^{M} \alpha_m \log(\beta) + (\alpha_m - 1) \mathbb{E}^{(k)}[\log(x_m)] - \mathbb{E}^{(k)}[x_m] \beta - \log(\Gamma(\alpha_m)) \\
= \sum_{t=1}^{T} -\hat{\mu}_t^{(k)} \mathbb{E}^{(k)}[z_t] + y_t \log(\hat{\mu}_t^{(k)}) + y_t \mathbb{E}^{(k)}[\log(z_t)] - \log(y_t!) \\
+ \sum_{m=1}^{M} \alpha_m \log(\beta) + (\alpha_m - 1) \mathbb{E}^{(k)}[\log(x_m)] - \mathbb{E}^{(k)}[x_m] \beta - \log(\Gamma(\alpha_m)).
\]  

(4.18)
And for the M-step, we need to evaluate:

\[
\hat{\mu}^{(k+1)} = \arg \max_\mu Q(\mu, \hat{\mu}^{(k)})
\]
\[
= \arg \max_\mu \sum_{t=1}^{T} -\mu_t \mathbb{E}^{(k)}[z_t] + y_t \log(\mu_t) + y_t \mathbb{E}^{(k)}[\log(z_t)] - \log(y_t!)
\]
\[
+ \sum_{m=1}^{M} \alpha_m \log(\beta) + (\alpha_m - 1) \mathbb{E}^{(k)}[\log(x_m)] - \mathbb{E}^{(k)}[x_m] \beta - \log(\Gamma(\alpha_m)).
\]

(4.19)

To evaluate expression (4.18), we thus need to find algebraic expressions \(\mathbb{E}^{(k)}[x_q]\) for \(q = (1, \ldots, M)\), and \(\mathbb{E}^{(k)}[z_t]\) and \(\mathbb{E}^{(k)}[\log(z_t)]\) \(t = (1, \ldots, T)\). From now on, we will suppress the superscript \((k)\). In Theorem 4.2.6, an algebraic expression for \(\mathbb{E}[x_q]\) for \(q = (1, \ldots, M)\) is given. An expression for \(\mathbb{E}[z_t]\) can then be found by applying the linearity of the expected value operator:

\[
\mathbb{E}[z_t] = \mathbb{E}[\sum_{k \in I_t} x_k] = \sum_{k \in I_t} \mathbb{E}[x_k] \quad t = (1, \ldots, T).
\]

(4.20)

However, it is not possible to find an algebraic expression for \(\mathbb{E}[\log(z_t)]\). This problem can be solved by deleting a number of terms in expression (4.20). Those expressions which do not depend on \(\mu\) can be deleted, because they are constant:

\[
\hat{\mu}^{(k+1)} = \arg \max_\mu \sum_{t=1}^{T} -\mu_t z_t^{(k+1)} + y_t \log(\mu_t).
\]

(4.21)

Note that this is one of the reasons we use a combination of the profile likelihood method and the EM-algorithm.

The idea behind the proof of Theorem 4.2.6 is that we first determine the density function of \(x_m\) given the counts and given the model parameters:

\[
f_{x_q}(x_q|y_1, \ldots, y_T) = \frac{P(y_1, \ldots, y_T, x_q)}{P(y_1, \ldots, y_T)}
\]

(4.22)

for \(q \in \{1, \ldots, M\}\). We then integrate out \(x_q\) to find the expected value. In Lemma 4.2.5, an expression for this density function is given. In the remainder of this section, the lemma and theorem are proven.

**Lemma 4.2.5.** For \(q \in \{1, \ldots, M\}\) and fixed \(\mu, \rho\) and \(\xi\), the conditional density function is:

\[\quad\]

\footnote{We suppressed the conditionality of the density functions on the parameters to make the expression less cumbersome.}
Chapter 4. EM Algorithm

\[ f(x_q | y_1, \ldots, y_T) = \left[ \sum_{1 \leq i \leq n_{q}} e^{-x_q (\beta + \sum_{j \neq q} \mu_j)} x_q^{-1 + \sum_{j \neq q} \kappa_{j,q}} \right]^{-1} \prod_{t=1}^{T} \left( \begin{array}{c} y_t \\ k_{t,1}, \ldots, k_{t,n} \end{array} \right) \]

Proof. Theorem 4.2.4 gives an expression for the denominator in Equation (4.22). Therefore, the only thing left to do is to find an expression for the numerator. This can be done by integrating \( x_{m} \) out of the complete likelihood function \( \mathcal{L}(\mu, \zeta, \rho | y_1, \ldots, y_T, x_1, \ldots, x_M) \) for all \( m \in \{1, \ldots, M\} \setminus \{q\} \). Equation 4.15 gives an expression for the likelihood function where the gamma random random variables are isolated. When Lemma 4.2.2 is applied:

\[ P(y_1, \ldots, y_T, x_q) = C * \sum_{1 \leq i \leq n_{q}} e^{-x_q (\beta + \sum_{j \neq q} \mu_j)} x_q^{-1 + \sum_{j \neq q} \kappa_{j,q}} \prod_{t=1}^{T} \left( \begin{array}{c} y_t \\ k_{t,1}, \ldots, k_{t,n} \end{array} \right) \prod_{1 \leq m \leq M | m \neq q} \left[ \frac{\Gamma(\alpha_m + \sum_{j \neq q} \kappa_{j,m})}{(\beta + \sum_{j \neq q} \mu_j)^{\alpha_m + \sum_{j \neq q} \kappa_{j,m}}} \right] \]

(4.24)

Recall that \( C = \prod_{t=1}^{T} \left[ \frac{\mu^y_t}{\nu^y_t!} \right] \prod_{m=1}^{M \setminus \{q\}} \left[ \frac{\nu^\mu_m}{\nu^\kappa_m!} \right] \). If we divide expression (4.24) by (4.22), we obtain the desired result. \( \square \)

Theorem 4.2.6. For \( q \in \{1, \ldots, M\} \) and fixed \( \mu, \rho \) and \( \zeta \), the conditional expectation of \( X_q \) given the count data \( (y_1, \ldots, y_T) \) and model parameters is:
Chapter 4. EM Algorithm

47

\[ E(X_q | y_1, \ldots, y_T) = \left[ \sum_{1 \leq j \leq nt} \prod_{t=1}^{T} \left( \begin{array}{c} y_t \\ k_{h_t,1}, \ldots, k_{h_t,L} \end{array} \right) \right] \]

\[ \times \Gamma(a_q + 1 + \sum_{l_{p,r}=q} k_{l_{p,r}}) / \left( \prod_{1 \leq m \leq M} \sum_{m \neq q} \left[ \sum_{n \leq 1} \prod_{t=1}^{T} \left( \begin{array}{c} y_t \\ k_{h_t,1}, \ldots, k_{h_t,L} \end{array} \right) \right] \right) \]

\[ \times \Gamma(a_m + \sum_{l_{p,r}=m} k_{l_{p,r}}) / \left( \prod_{1 \leq m \leq M} \sum_{m \neq q} \left[ \sum_{n \leq 1} \prod_{t=1}^{T} \left( \begin{array}{c} y_t \\ k_{h_t,1}, \ldots, k_{h_t,L} \end{array} \right) \right] \right) \]

Proof. We can find the expected value by applying the following formula:

\[ E(X_q | y_1, \ldots, y_T) = \int_{x_q=0}^{\infty} x_q \ast f_{x_q}(x_q | y_1, \ldots, y_T) \, dx_q. \quad (4.26) \]

The only term in \( f_{x_q}(x_q | y_1, \ldots, y_T) \) depending on \( x_q \) in Lemma 4.2.5 is:

\[ e^{-x_q \left( \beta + \sum_{l_{p,r}=q} \mu_r \right)} x_q^{a_q-1 + \sum_{l_{p,r}=q} k_{l_{p,r}}} \cdot \]

If we multiply this expression by \( x_q \) and apply Lemma 4.2.2, we obtain the desired result.

4.2.4 Number of Summands

The expressions for the E-step (Theorem 4.25) and the observed likelihood (Theorem 4.2.4) are cumbersome and might be computationally intractable. The main reason for this is that the number of summands increases rapidly with the number of time points \( T \) or the counts \( y \). The goal of this section is to assess how quickly the number of summands increases.

For simplicity, we assume that the model parameters \( \mu, \zeta \) and \( \rho \) are fixed, and that the counts are constant for all units and time points: \( y_{ij} = y \) for all \( i = (1, \ldots, T) \) and \( j = (1, \ldots, N) \). A formula for the number of summands is given in Lemma 4.2.7.

Lemma 4.2.7. For a positive integer \( T \) and positive integer \( y \), the number of summands in the sum

\[ \sum_{\sum_{i,j \leq n} k_{ij} = y} \prod_{i=1}^{T} \left( \begin{array}{c} y_t \\ k_{h_t,1}, \ldots, k_{h_t,L} \end{array} \right) \]

is given by:

\[ \sum_{\sum_{i,j \leq n} k_{ij} = y} \prod_{i=1}^{T} \left( \begin{array}{c} y_t \\ k_{h_t,1}, \ldots, k_{h_t,L} \end{array} \right) \]

\[ \times \Gamma(a_q + 1 + \sum_{l_{p,r}=q} k_{l_{p,r}}) / \left( \prod_{1 \leq m \leq M} \sum_{m \neq q} \left[ \sum_{n \leq 1} \prod_{t=1}^{T} \left( \begin{array}{c} y_t \\ k_{h_t,1}, \ldots, k_{h_t,L} \end{array} \right) \right] \right) \]

\[ \times \Gamma(a_m + \sum_{l_{p,r}=m} k_{l_{p,r}}) / \left( \prod_{1 \leq m \leq M} \sum_{m \neq q} \left[ \sum_{n \leq 1} \prod_{t=1}^{T} \left( \begin{array}{c} y_t \\ k_{h_t,1}, \ldots, k_{h_t,L} \end{array} \right) \right] \right) \]
is:

\[ N_{\text{sum}} = \prod_{t=1}^{T} \frac{(y + (T - t + 1)t + 1)(T - t + 1)t - 1}{(T - t + 1)t - 1}. \]

Proof. It is well known that the number of terms in a multinomial sum

\[ \sum_{k_1, \ldots, k_m = y} \]

is \((y + m - 1)\). Recall from Section 3.3 that for given \(T\), the number of gamma random variables \(X_1, \ldots, X_M\) is \(M = \frac{1}{2} T(T + 1)\). For \(t \in (1, \ldots, T)\), \(Z_t\) is defined as \(Z_t = \sum_{i=1}^{T} \sum_{j=1}^{T} X_{ij}\). This means that the frailty term \(Z_t\) consists of the following number of components:

\[ n_t = \#I_t = \sum_{j=t}^{T} \sum_{i=1}^{1} 1 = \sum_{j=t}^{T} t = (T - t + 1)t. \] (4.27)

Therefore, the number of summands is:

\[ N_{\text{sum}} = \prod_{t=1}^{T} \frac{(y + (T - t + 1)t + 1)(T - t + 1)t - 1}{(T - t + 1)t - 1}. \] (4.28)

This concludes the proof.

From Lemma 4.2.7, we conclude that the number of summands increases exponentially or factorial with the number of time points \(T\) and counts \(y\). This might not necessarily be a problem, because in applications, \(T\) and \(y\) do not have to be extremely large. Therefore, to get an idea of how high the number of the number of summands is for values of \(T\) and \(y\) commonly encountered in practice, see Table 4.1.

<table>
<thead>
<tr>
<th>Counts</th>
<th>(T = 2)</th>
<th>(T = 3)</th>
<th>(T = 5)</th>
<th>(T = 10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>121</td>
<td>1.24 \times 10^6</td>
<td>1.65 \times 10^{19}</td>
<td>2.55 \times 10^{24}</td>
</tr>
<tr>
<td>20</td>
<td>441</td>
<td>9.45 \times 10^7</td>
<td>2.76 \times 10^{26}</td>
<td>1.78 \times 10^{31}</td>
</tr>
<tr>
<td>50</td>
<td>2601</td>
<td>4.11 \times 10^{10}</td>
<td>1.34 \times 10^{37}</td>
<td>1.23 \times 10^{42}</td>
</tr>
</tbody>
</table>

Table 4.1: The number of summands for different values of \(y\) and \(T\).

From this table, we can conclude that the algebraic expressions for the E-step and observed likelihood are probably computationally intractable for \(T > 3\). Whether or not expressions are computationally tractable for \(T = 2\) or \(T = 3\) will have to be investigated by performing simulation studies.
4.3 **R-code**

In this section, we discuss the code which the authors of this thesis wrote for the implementation of the EM-algorithm in R for two time points \((T = 2)\). To see the complete R-code used, see Appendix A.4.

In Section 4.3.1, the format of the input data is discussed. In Sections 4.3.2, 4.3.3, and 4.3.4, the implementation for the E-step, M-step and observed log likelihood are respectively discussed. In the Section 4.3.5, the main function which covers the rest of the steps and calls to the three previously named functions is discussed.

### 4.3.1 Format of Data

The data is given to the main function in a long format. The idea behind the long format is that the data is put in a table with minimum number of columns, as opposed to the wide format, where it is the other way around. See for example Table 4.2 and Table 4.3. Both tables display the same data, but the former is displayed in a long format and the latter in a wide format.

<table>
<thead>
<tr>
<th>Time</th>
<th>Study</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

**Table 4.2**: An example of count data in a data long format for three clusters and two time points

<table>
<thead>
<tr>
<th>Study</th>
<th>Time Point 1</th>
<th>Time Point 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

**Table 4.3**: An example of count data in a wide format for three clusters and two time points

### 4.3.2 E-step

The goal of this section is to write a function `E_step` which executes the E-step of the algorithm based on the current estimates of \(\rho, \xi, \mu_1\) and \(\mu_2\). Recall that model 2 needs \(M = \frac{1}{2}T(T + 1)\) gamma random variables to construct the frailty terms. We thus need three gamma random variables: \(x_1, x_2\) and \(x_3\) to implement the EM-algorithm for two time points. In this situation, the formula’s from Theorem 4.2.6 simplify to:
Chapter 4. EM Algorithm

\[ \mathbb{E}_\gamma(x_1 | y_1, y_2) = \frac{\sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \binom{y_2}{k} \frac{\Gamma(k+l+\alpha_1+1)}{(\mu_1+\mu_2+\beta)^{k+l+\alpha_1+1}} \frac{\Gamma(y_1-k+\alpha_2)}{(\mu_1+\beta)^{y_1-k+\alpha_2}} \frac{\Gamma(y_2-l+\alpha_3)}{(\mu_2+\beta)^{y_2-l+\alpha_3}}}{\sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \binom{y_2}{k} \frac{\Gamma(k+l+1)}{(\mu_1+\mu_2+\beta)^{k+l+1}} \frac{\Gamma(y_1-k+1)}{(\mu_1+\beta)^{y_1-k+1}} \frac{\Gamma(y_2-l)}{(\mu_2+\beta)^{y_2-l}}} \]  
(4.29)

\[ \mathbb{E}_\delta(x_2 | y_1, y_2) = \frac{\sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \binom{y_2}{k} \frac{\Gamma(k+l+\alpha_1+1)}{(\mu_1+\mu_2+\beta)^{k+l+\alpha_1+1}} \frac{\Gamma(y_1-k+\alpha_2)}{(\mu_1+\beta)^{y_1-k+\alpha_2+1}} \frac{\Gamma(y_2-l+\alpha_3)}{(\mu_2+\beta)^{y_2-l+\alpha_3}}}{\sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \binom{y_2}{k} \frac{\Gamma(k+l+1)}{(\mu_1+\mu_2+\beta)^{k+l+1}} \frac{\Gamma(y_1-k+1)}{(\mu_1+\beta)^{y_1-k+1}} \frac{\Gamma(y_2-l+1)}{(\mu_2+\beta)^{y_2-l+1}}} \]  
(4.30)

\[ \mathbb{E}_\xi(x_3 | y_1, y_2) = \frac{\sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \binom{y_2}{k} \frac{\Gamma(k+l+\alpha_1+1)}{(\mu_1+\mu_2+\beta)^{k+l+\alpha_1+1}} \frac{\Gamma(y_1-k+\alpha_2)}{(\mu_1+\beta)^{y_1-k+\alpha_2}} \frac{\Gamma(y_2-l+\alpha_3+1)}{(\mu_2+\beta)^{y_2-l+\alpha_3+1}}}{\sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \binom{y_2}{k} \frac{\Gamma(k+l+1)}{(\mu_1+\mu_2+\beta)^{k+l+1}} \frac{\Gamma(y_1-k+1)}{(\mu_1+\beta)^{y_1-k+1}} \frac{\Gamma(y_2-l+1)}{(\mu_2+\beta)^{y_2-l+1}}} \]  
(4.31)

A problem with expressions [4.29], [4.30] and [4.31] is that the function \( \text{gamma}() \), does not handle big numbers well. When the number over which \( \text{gamma}() \) is evaluated is bigger than ±180, this function returns \( \text{Inf} \), which stands for ‘infinite’. To partially solve this problem, we can take the exp of the log of the values to be summed over. For example, expression [4.29] then becomes:

\[ \mathbb{E}_\gamma(x_1 | y_1, y_2) = \sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \exp \left[ \log \left( \frac{y_1}{k} \right) + \log \left( \frac{y_2}{l} \right) \right] + \log(\Gamma(k+l+\alpha_1+1)) - (k+l+\alpha_1+1) \log(\mu_1 + \mu_2 + \beta) + \log(\Gamma(y_1-k+\alpha_2)) - (y_1-k+\alpha_2) \log(\mu_1 + \beta) + \log(\Gamma(y_2-l+\alpha_3)) - (y_2-l+\alpha_3) \log(\mu_2 + \beta) \]

\[ / \sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \exp(\log \left( \frac{y_1}{k} \right) + \log \left( \frac{y_2}{l} \right)) + \log(\Gamma(k+l+1)) - (k+l+1) \log(\mu_1 + \mu_2 + \beta) + \log(\Gamma(y_1-k+1)) - (y_1-k+1) \log(\mu_1 + \beta) + \log(\Gamma(y_2-l)) - (y_2-l) \log(\mu_2 + \beta) \]  
(4.32)

However, this does not entirely solve the problem, as the function \( \exp \) also returns \( \text{Inf} \) when the number over which it is evaluated exceeds ±700. A solution is to divide both the denominator and the numerator by a constant \( C \). For example, expression [4.32] then becomes:
\[ \mathbb{E}_\gamma(x_1|y_1,y_2) = \sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \exp \left[ \log \left( \frac{y_1}{k} \right) + \log \left( \frac{y_2}{l} \right) \right. \\
\left. + \log(\Gamma(k+l+a_1+1)) - (k+l+a_1) \log(\mu_1 + \mu_2 + \beta) \right. \\
\left. + \log(\Gamma(y_1-k+a_2)) - (y_1-k+a_2) \log(\mu_1 + \beta) \right. \\
\left. + \log(\Gamma(y_2-l+a_3)) - (y_2-l+a_3) \log(\mu_2 + \beta) - \log(C) \right] \\
/ \sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \exp \left[ \log \left( \frac{y_1}{k} \right) + \log \left( \frac{y_2}{l} \right) \right. \\
\left. + \log(\Gamma(k+l+a_1)) - (k+l+a_1) \log(\mu_1 + \mu_2 + \beta) \right. \\
\left. + \log(\Gamma(y_1-k+a_2)) - (y_1-k+a_2) \log(\mu_1 + \beta) \right. \\
\left. + \log(\Gamma(y_2-l+a_3)) - (y_2-l+a_3) \log(\mu_2 + \beta) \right] . \]

We define \( C \) as the mean of the summands in the numerator:

\[
C = \frac{1}{y_1y_2} \sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \log \left( \frac{y_1}{k} \right) + \log \left( \frac{y_2}{l} \right) \]

\[
+ \log(\Gamma(k+l+a_1)) - (k+l+a_1) \log(\mu_1 + \mu_2 + \beta) \tag{4.34} \\
+ \log(\Gamma(y_1-k+a_2)) - (y_1-k+a_2) \log(\mu_1 + \beta) \\
+ \log(\Gamma(y_2-l+a_3)) - (y_2-l+a_3) \log(\mu_2 + \beta). \]

The R-code used to implement these formulas is given below. Note that we did not yet substitute the values \( a_1 = \frac{\rho}{\xi} \) and \( a_2 = a_3 = \frac{1-\rho}{\xi} \) (see Formula (3.5)) yet. The values of \( a_1, a_2 \) and \( a_3 \) thus have to be computed before calling the function E_step.

Note that the denominators in Formula’s (4.29), (4.30) and (4.31) are the same. This value is stored in the variable den, which stands for denominator. The numerators of these formula’s are stored in num_1, num_2 and num_3 respectively. \( a_1, a_2, a_3 \) and \( b \) stand for \( a_1, a_2, a_3 \) and \( \beta \) respectively. \( \mu_1 \) and \( \mu_2 \) stand for \( \mu_1 \) and \( \mu_2 \) and the expected values are stored in e_x_1, e_x_2 and e_x_3. All other variables are help variables which are used to make computation easier or to prevent expressions to be computed more often than necessary.

The function E_step also calculates the contribution to the observed log likelihood of the cluster of data. The reason for this is that the EM-algorithm needs to keep track of the observed log likelihood to know when the algorithm has to terminate. The formula’s for the observed likelihood and the expected values are similar. Therefore, it is efficient to calculate the observed log likelihood within the function E_step. This value is stored in loglik.

```R
E_step <- function(y_1, y_2, a_1, a_2, a_3, b, mu_1, mu_2) {
```
# Function to calculate expected value of x_1, x_2, x_3
# Input:
# The count data y_1, y_2 and current estimates of model parameters mu_1, 
# mu_2, rho, xi,
# a_1, a_2, a_3 depend on rho and xi (e.g. a_1 = rho/xi),
# Output:
# Expected values of x_1, x_2, x_3
# Variables to store expected values x_1, x_2, x_3 in
e_x_1 <- 0; e_x_2 <- 0; e_x_3 <- 0
# Variables to store denominator and numerator for x_1, x_2, x_3 in
den <- 0; num_1 <- 0; num_2 <- 0; num_3 <- 0
# In the following variables, matrices are stored which contain elements 
to be summed over to calculate the denominator and numerators
h_0 <- 0; h_1 <- 0; h_2 <- 0; h_3 <- 0
h_1_extra <- 0; h_2_extra <- 0; h_3_extra <- 0
# Help vectors to reduce computation time by preventing gamma and binomial 
coefficients to be computed more times than necessary
bin.1 <- lchoose(y_1, {0:y_1})
bin.2 <- lchoose(y_2, {0:y_2})
max.1 <- y_1+y_2+1
max.2 <- y_1 + 1
max.3 <- y_2 + 1
frac.a1 <- lgamma(a_1 + (0:max.1)) - (a_1 + (0:max.1))*log(mu_1+mu_2+b)
frac.a2 <- lgamma(a_2 + (0:max.2)) - (a_2 + (0:max.2))*log(mu_1+b)
frac.a3 <- lgamma(a_3 + (0:max.3)) - (a_3 + (0:max.3))*log(mu_2+b)
# Values to be summed over are calculated
h_0 <- outer(0:y_1, 0:y_2, FUN=function(r,c) bin.1[r+1] + bin.2[c+1])
h_1 <- outer(0:y_1, 0:y_2, FUN=function(r,c) frac.a1[r+c+1])
h_2 <- outer(0:y_1, 0:y_2, FUN=function(r,c) frac.a2[y_1-r+1])
h_3 <- outer(0:y_1, 0:y_2, FUN=function(r,c) frac.a3[y_2-c+1])
# Values to be summed over where there is an extra ‘+1’ in the terms
h_1_extra <- outer(0:y_1, 0:y_2, FUN=function(r,c) frac.a1[r+c+2])
h_2_extra <- outer(0:y_1, 0:y_2, FUN=function(r,c) frac.a2[y_1-r+2])
h_3_extra <- outer(0:y_1, 0:y_2, FUN=function(r,c) frac.a3[y_2-c+2])
# Calculate constant to subtract from both denominator and numerator 
# to make sure the function handles high counts well
C <- mean(h_0+h_1+h_2+h_3)
# Denominator and numerators of x_1, x_2, x_3 are calculated
den <- sum(exp(h_0+h_1+h_2+h_3-C))
um_1 <- sum(exp(h_0+h_1_extra+h_2+h_3-C))
um_2 <- sum(exp(h_0+h_1+h_2_extra+h_3-C))
um_3 <- sum(exp(h_0+h_1+h_2+h_3_extra-C))
# Calculating expected values
e_x_1 <- num_1 / den
e_x_2 <- num_2 / den
e_x_3 <- num_3 / den
# Calculating log observed data likelihood with new values of x_1, x_2, x_3 
# A call to ‘obs_log_lik’ would also be an option, but then part of the 
calculations 
# would be done twice
loglik <- (a_1+a_2+a_3)*log(b)-lgamma(a_1)-lgamma(a_2)-lgamma(a_3)+y_1*
log(mu_1)+y_2*log(mu_2)-lfactorial(y_1)-lfactorial(y_2)+C+log(den)
return(cbind(e_x_1, e_x_2, e_x_3, -loglik))
4.3.3 M-step

The goal of this section is to write a function \texttt{M\_step} which executes the M-step of the EM-algorithm based on the current estimates of \(\rho\), \(\xi\) and \(Z\). For \(T = 2\), expression (4.20) is:

\[
\hat{\mu}^{(k+1)} = \arg \max_{\mu_1, \mu_2} \sum_{i=1}^{N} -\mu_1 \hat{z}_{i1}^{(k+1)} + y_{i1} \log(\mu_1) - \mu_2 \hat{z}_{i2}^{(k+1)} + y_{i2} \log(\mu_2). \tag{4.35}
\]

Instead of evaluating (4.35), we evaluate:

\[
\arg \max_{\mu_1, \mu_2} \sum_{i=1}^{N} -\mu_1 \hat{z}_{i1}^{(k+1)} + y_{i1} \log(\mu_1) - \log(y_{i1}!) - \mu_2 \hat{z}_{i2}^{(k+1)} + y_{i2} \log(\mu_2) - \log(y_{i2}!). \tag{4.36}
\]

The reason for this is that we can now rewrite (4.36) as:

\[
\arg \max_{\mu_1, \mu_2} \sum_{i=1}^{N} \log[P_{\text{pois}}(y_{i1}; \mu_1 \hat{z}_{i1}^{(k+1)})] + \log[P_{\text{pois}}(y_{i2}; \mu_2 \hat{z}_{i2}^{(k+1)})]. \tag{4.37}
\]

The advantage of evaluating (4.37) in \(\mathbb{R}\) is that there are packages available to do this efficiently. The \texttt{glm}-function, which is available in standard versions of \(\mathbb{R}\), can do this for us. It is a function which can be used to fit generalized linear models. The frailty terms \(z_1\) and \(z_2\) are treated as \texttt{offset}, which means that these variables are assumed to be constant and work multiplicatively on \(\mu_1\) and \(\mu_2\).

The reason why evaluating (4.36) gives the same results as evaluating (4.35) is that the only thing we did is add constants. So even though \(\mathbb{E}[\log(x_{it})] \neq \log(\mathbb{E}[x_{it}])\) for \(t = (1, 2)\) and \(i = (1, \ldots, N)\), we can add the terms \(y_{it} \log(x_{it})\) to expression (4.35) without changing the outcome, as \(x_{it}\) can be considered a constant when evaluating the expression.

The function \texttt{M\_step} first calculates the values of \(z_1\) and \(z_2\) based on \(x_1\), \(x_2\) and \(x_3\) (see lines 12-14 below). Then, the vectors \(y_1\), \(y_2\), \(z_1\), \(z_2\) and \(z_3\) are put into a data long format, as the \texttt{glm}-function needs the data to be given in this format (see lines 16-27 below). Afterward, the Poisson regression is executed (see lines 29-31 below) and estimates for \(\mu_1\) and \(\mu_2\) are extracted (see lines 33-36 below).

```r
M_step <- function(data,N) {
#Function which calculates estimates for mu_1, mu_2
#Input:
# count data y_1,y_2 and current values of gamma variables x_1,x_2,x_3, all stored in 'data'
# number of centers N
#Output:
# Estimates of mu_1 and mu_2
#Number of time points, defined for clarity
T <- 2

# Function which calculates estimates for mu_1, mu_2
# Input:
# count data y_1,y_2 and current values of gamma variables x_1,x_2,x_3, all stored in 'data'
# number of centers N
# Output:
# Estimates of mu_1 and mu_2
# Number of time points, defined for clarity
T <- 2
```
# Compute frailty terms
z1 <- data$x_1 + data$x_2
z2 <- data$x_1 + data$x_3

# Put counts y_1, y_2 and frailty terms z_1, z_2 in data long format
# to be able to let glm work on it
datalong <- data.frame(id = rep(1:N, rep(T, N)),
                      time = rep(1:T, N),
                      y = as.vector(t(as.matrix(cbind(data$y_1, data$y_2)))),
                      z = as.vector(t(as.matrix(cbind(z1, z2)))))

data$long$logz <- log(data$long$z)

data$long$time <- factor(data$long$time)

# Fit a poisson regression model on the time points with frailty terms
# treated as offset
glmfit <- glm(y ~ time + offset(logz), data=datalong, family="poisson")

# Calculate mu_1, mu_2 by using regression coefficients
betas <- glmfit$coef
mus <- exp(c(betas[1], betas[1] + betas[-1]))
names(mus) <- c("mu_1", "mu_2")

# Return mu_1, mu_2
return(mus)

4.3.4 Observed Likelihood

The goal of this section is to write a function obs_log_lik which calculates the observed log likelihood based on the current estimates of \( \rho, \xi, \mu_1, \mu_2, \) and \( Z \). For \( T=2 \), the expression of the observed likelihood given in Theorem 4.2.4 is:

\[
L_{obs}(\mu_1, \mu_2, \xi, \rho|y_1, y_2) = \frac{\beta^{a_0+a_1+a_2}}{\Gamma(a_0)\Gamma(a_1)\Gamma(a_2)} \frac{\mu_1\mu_2}{y_1!y_2!} \sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \frac{y_1}{k} \frac{y_2}{l} \frac{\Gamma(y_1-k+1)\Gamma(y_2-l+1+\alpha_2)\Gamma(k+l+\alpha_0)}{(\mu_1+\beta)^{y_1-k+\alpha_1}(\mu_2+\beta)^{y_2-l+\alpha_2}(\mu_1+\mu_2+\beta)^{k+l+a_0}}
\] (4.38)

The logarithm of equation (4.38) is:
\[ \ell_{\text{obs}}(\mu_1, \mu_2, \xi, \rho | y_1, y_2) = (a_0 + a_1 + a_2) \log(\beta) - \log(\Gamma(a_0)) - \log(\Gamma(a_1)) - \log(\Gamma(a_2)) \\
+ y_1 \log(\mu_1) + y_2 \log(\mu_2) - \log(y_1!) - \log(y_2!) \\
+ \log\left( \sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \exp[ \log\left( \frac{y_1}{k} \right) + \log\left( \frac{y_2}{l} \right) ] \right) \\
+ \log(\Gamma(y_1 - k + a_1)) + \log(\Gamma(y_2 - l + a_2)) \\
+ \log(\Gamma(k + l + a_0)) \\
- (y_1 - k + a_1) \log(\mu_1 + \beta) - (y_2 - l + a_2) \log(\mu_2 + \beta) \\
- (k + l + a_0) \log(\mu_1 + \mu_2 + \beta) \]  

(4.39)

We took the log of the exp within the sum, like in Section 4.3.3, because the formula then works better in \( \mathbb{R} \) with high counts. We can further improve the implementation by introducing a constant. Let:

\[ A = (a_0 + a_1 + a_2) \log(\beta) - \log(\Gamma(a_0)) - \log(\Gamma(a_1)) - \log(\Gamma(a_2)) \]

(4.40)

and define:

\[ B_{kl} = \log\left( \frac{y_1}{k} \right) + \log\left( \frac{y_2}{l} \right) \\
+ \log(\Gamma(y_1 - k + a_1)) + \log(\Gamma(y_2 - l + a_2)) + \log(\Gamma(k + l + a_0)) \\
- (y_1 - k + a_1) \log(\mu_1 + \beta) - (y_2 - l + a_2) \log(\mu_2 + \beta) \\
- (k + l + a_0) \log(\mu_1 + \mu_2 + \beta) \]

(4.41)

for \( k = (1, \ldots, y_1) \) and \( l = (1, \ldots, y_2) \). Let \( C \) be defined as in (4.34). We can then rewrite expression (4.39) in the following way:

\[ \ell_{\text{obs}}(\mu_1, \mu_2, \xi, \rho | y_1, y_2) = A + \log\left( \sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \exp(B_{kl}) \right) = A + \log\left( \sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \exp(B_{kl} - C) \exp(C) \right) \]

(4.42)

\[ = A + C + \log\left( \sum_{k=0}^{y_1} \sum_{l=0}^{y_2} \exp(B_{kl} - C) \right). \]

The \( \mathbb{R} \)-code for the function \texttt{obs_log_lik} is given below. The code is very similar to the code given in Section 4.3.2. In lines 8 through 12, the values of \( a_1, a_2, a_3 \) and \( \beta \) are computed. In lines 14 through 18, help variables are defined. In lines 22 through 30, help variables are defined to prevent multiple evaluation of the same gamma and binomial terms. In lines 33 through
38, the elements to be summed over (see expression (4.41)) are computed.
In lines 41 through 43, the constant C is computed and in lines 45 through 47 the value of the observed log likelihood is computed.

```r
obs_log_lik <- function(y_1, y_2, N, rho, theta, mu_1, mu_2) {
  # Function which calculates the observed log likelihood
  # Input:
  # count data y_1, y_2 and N and current parameter estimates rho, theta, mu_1 and mu_2
  # Output:
  # Minus observed log likelihood
  # Calculate values a_1, a_2, a_3 and b based on rho and theta
  a_1 <- theta * rho
  a_2 <- theta * (1-rho)
  a_3 <- theta * (1-rho)
  b <- theta
  # Variable in which output will be stored
  loglik <- 0
  # In the following variables, matrices are stored which contain elements
to be summed over to calculate the log likelihood
  h_0 <- 0; h_1 <- 0; h_2 <- 0; h_3 <- 0; h_4 <- 0; h_5 <- 0
  # Calculation of observed log likelihood
  for(i in 1:N) {
    # Help vectors to reduce computation time by preventing gamma and
    # binomial coefficients to be calculated more times than necessary
    bin.1 <- lchoose(y_1[i], 0:y_1[i])
    bin.2 <- lchoose(y_2[i], 0:y_2[i])
    max.1 <- y_1[i] + y_2[i]
    max.2 <- y_1[i]
    max.3 <- y_2[i]
    frac.a1 <- lgamma(a_1 + (0:max.1)) - (a_1 + (0:max.1)) * log(mu_1 + mu_2 + b)
    frac.a2 <- lgamma(a_2 + (0:max.2)) - (a_2 + (0:max.2)) * log(mu_1 + b)
    frac.a3 <- lgamma(a_3 + (0:max.3)) - (a_3 + (0:max.3)) * log(mu_2 + b)
    # Values to be summed over are calculated
    h_5 <- 0
    h_0 <- (a_1 + a_2 + a_3) * log(b) - lgamma(a_1) - lgamma(a_2) - lgamma(a_3) - y_1[i]
      * log(mu_1 + y_2[i] * frac.a1 - 1factorial(y_1[i]) - 1factorial(y_2[i]))
    h_1 <- outer(0:y_1[i], 0:y_2[i], FUN=function(r, c) bin.1[r+1] + bin.2[c +1])
    h_2 <- outer(0:y_1[i], 0:y_2[i], FUN=function(r, c) frac.a1[r+c+1])
    h_3 <- outer(0:y_1[i], 0:y_2[i], FUN=function(r, c) frac.a2[y_1[i]-r+1])
    h_4 <- outer(0:y_1[i], 0:y_2[i], FUN=function(r, c) frac.a3[y_2[i]-c+1])
    # Calculate constant to add and subtract such that the expression
    # handles high counts better
    C <- mean(h_1 + h_2 + h_3 + h_4)
    # Calculate observed log likelihood
    h_5 <- sum(exp(h_1 + h_2 + h_3 + h_4 - C))
    loglik <- loglik + h_0 + C + log(h_5)
  }
  return(-loglik)
}
```

### 4.3.5 Main Function

The goal of this section is to write a function `EM_estimates` which executes the combination of the EM-algorithm and the profile likelihood method.
Input variables are the data in long format (datalong), the number of clusters (N), initial values for \( \rho \) and \( \xi \) (rho_IV and ksi_IV) and parameters \( \epsilon_\mu \) and \( \epsilon_\text{lik} \) indicating when the algorithm has to terminate (acc_mus and acc_lik).

The goal of the function EM_estimates is to find values of \( \rho \) and \( \xi \) which maximize the profile likelihood function (see Section 4.1). To find these maximum likelihood estimates, we used standard packages in \( \mathbb{R} \) provided by the R Core Team [2014]. The function we used in this package is the optim function (see lines 82-83 below). We chose the L-BFGS-B method, because it allows box constraints, which we need because \( \rho \) us bound to \([0, 1]\) and \( \theta \) to \((0, \infty)\). As explained in the introduction of Section 4.2, we do not take into account the special cases \( \rho = 0, \rho = 1 \) and \( \theta = \infty \). Therefore, we only allow values of \( \rho \) in \([0.001, 0.999]\) and for \( \theta \) in \([0.001, 100]\) to be chosen by the maximizer.

The function prof_log_lik returns the value of the profile log likelihood function for values of \( \rho \) and \( \xi \). This function is given in lines 35 through 79 below. The EM-algorithm is executed in lines 51 through 74. It terminates when the difference between two consecutive estimates of the observed log likelihood is less than acc_lik (see line 51). Within the EM-algorithm described lines 51 through 74, we do not use the difference between two consecutive estimates of \( \mu \) to determine when the algorithm terminates, because when searching for optimal values of \( \rho \) and \( \xi \), it is not yet necessary that values for \( \mu \) are estimated accurately, because they are deleted in a next iteration. At this stage, it is sufficient that the observed log likelihood is computed accurately. In order to compare the current and previous values of the observed log likelihood, in lines 46-47 two variables loglik_old and loglik_new are defined in which they are stored.

The E-step is executed in line 57 and the M-step in line 65. Initial estimates for \( \mu_1 \) and \( \mu_2 \) are computed in lines 24 through 26. The values for the \( \mu \)'s that are computed in the M-step are stored in the global variables mu_1 and mu_2. To do this, the operator «- is used, which makes sure that the values of \( \mu_1 \) and \( \mu_2 \) are stored for the next time that the function prof_log_lik is called to. Else, these values will be deleted after the value of the observed log likelihood is returned.

When the maximum likelihood estimators for \( \rho \) and \( \xi \) are found, we also want the values for \( \mu \) to be computed accurately. Therefore, the EM-algorithm is executed one last time after optimal values for \( \rho \) are \( \xi \) are found (see lines 94 through 110 below). For this execution, the accuracy parameter acc_mus is used. The EM-algorithm stops when the difference between two consecutive values of \( \mu \) is less than acc_mus (see line 98 below). Within this execution of the EM-algorithm, we need to keep track of the difference between two consecutive values of \( \mu \) instead of the observed log likelihood. In order to compare the current and previous estimates, in lines 94-96 two variables mu_1_new and mu_2_new are defined in which the estimates are stored, and the old values are stored in mu_1_old and mu_2_old (see lines 99-100).

There is no logical way of quickly estimating reasonable initial estimates for \( \rho \) and \( \xi \) before executing the algorithm. Therefore, it was decided that
estimates have to be inputted manually by the user. If the researcher before executing the algorithm has initial knowledge on these parameters, he can use this knowledge to input ‘reasonable’ first estimates.

We do not need to compare the estimates of \( x_1, x_2 \) and \( x_3 \) between two consecutive steps, therefore we store these (together with the count data \( y_1, y_2 \)) in a variable called \( x_{\text{and}_y} \) (see lines 17 through 22 below) and overwrite the old values by the new ones when we update these.

As we suspect that computation time will pose a problem when we expand the algorithm to more than two time points, we want to keep track of the computation time of the different parts of the algorithm so that we know where the bottle neck is. To do this, we make use of the function \texttt{proc.time()}. This function returns, amongst other things, the argument ‘elapsed’ which indicates the elapsed time since starting the process.

We chose to keep track of the total iteration time of the algorithm, and two separate steps: the E-step and M-step. We do this by storing the current processing time in a variable called \texttt{proc.time.current} before we start one of these three steps. After executing the step, we store \texttt{proc.time.current} in \texttt{proc.time.previous} and store the current processing time in \texttt{proc.time.current}. The difference between these two times is then gives the time the step took. We then add up this difference to a variable which stores the total execution time for that step (called \texttt{time_E_step} and \texttt{time_M_step} for the two steps respectively). The total execution of the algorithm is computed similarly. (See lines 28-33, 56, 58 and 66 below.)

```r
EM_estimates <- function(datalong, N, rho_IV, ksi_IV, acc_mus, acc_loglik){
  #Input:
  # count data in datalong format,
  # number of centers N,
  # initial estimates (values) for rho and xi,
  # accuracy parameters acc_mus and acc_loglik which indicates when the
  # EM-algorithm terminates
  #Output:
  # Estimates for mu_1,mu_2,rho,theta(=1/xi)
  # and execution times of different parts of the algorithm
  #Store count-data in better readable variables
  y_1 <- subset(datalong$count, datalong$Time == 1)
  y_2 <- subset(datalong$count, datalong$Time == 2)
  #Make dataframe to store current estimates of x_1, x_2, x_3, minus
  # observed likelihood, and y_1,y_2
  x_1 <- rep(0,N)
  x_2 <- rep(0,N)
  x_3 <- rep(0,N)
  minloglik <- rep(0,N)
  x_and_y <- data.frame(x_1, x_2, x_3, minloglik, y_1, y_2)
  #Initial (biased) estimates for mu_1 and mu_2
  mu_1 <- mean(y_1)
  mu_2 <- mean(y_2)
  #To keep track of computation time for different parts of algorithm,
  # we define a couple of variables to store proces times in
  proc_time_total <- proc.time()
  proc_time_current <- proc.time()
  proc_time_previous <- 0
  time_E_step <- 0
  time_M_step <- 0
  prof_log_lik <- function(par){
```
Chapter 4. EM Algorithm

#Function which returns minus profile log likelihood for values of rho and theta
#par[1] = rho
#par[2] = theta

#Variable which keeps track of the number of iterations of the EM-algorithm
count <- 0

#Make sure the EM-algorithm starts
diff_loglik <- acc_loglik*2

#Variable in which current and previous observed log likelihood values are stored
loglik_old <- 0; loglik_new <- 0

#EM-algorithm, terminates when either the difference in mu's or difference in consecutive values of the observed log likelihood are below the accuracy parameters
while(diff_loglik >= acc_loglik){
  #Keep track of number of iterations of algorithm
  count = count+1

  #E-step: update values of x_1,x_2,x_3 based on previous estimates of mu_1,mu_2
  proc_time_current <- proc.time()
  for(i in 1:N){x_and_y[i,1:4] <- E_step(y_1=y_1[i], y_2=y_2[i], a_1=par[2]*par[1], a_2 = par[2]*(1-par[1]), a_3=par[2]*(1-par[1]), b=par[2], mu_1=mu_1, mu_2=mu_2)}
  proc_time_previous <- proc_time_current; proc_time_current <- proc.time(); time_E_step <<- time_E_step + proc_time_current["elapsed"]-proc_time_previous["elapsed"]

  #Store new value of log observed likelihood
  loglik_old <- loglik_new
  loglik_new <- sum(x_and_y[,4])

  #M-step: update values of mu_1,mu_2 based on new estimates x_1,x_2,x_3 and store in help variable h1
  h1 <- M_step(data = x_and_y,N=N)
  proc_time_previous <- proc_time_current; proc_time_current <- proc.time(); time_M_step <<- time_M_step + proc_time_current["elapsed"]-proc_time_previous["elapsed"]

  #Store new values of mu_1,mu_2
  mu_1 <<- h1["mu_1"]
  mu_2 <<- h1["mu_2"]

  #Calculate values which determine whether the EM-algorithm terminates
  diff_loglik <- abs(loglik_old-loglik_new)
}

#The function 'opt' optimizes the function 'prof_log_lik' over rho and theta
#It searches for rho in the interval [0.001,0.999] and for theta in [0.001,100]
opt <- optim(par=c(rho_IV,(1/ksi_IV)), fn=prof_log_lik, method="L-BFGS-B", lower= c(0.001,0.001), upper=c(0.999,100))
proc_time_total <- proc.time()-proc_time_total

#We store the results (rho,theta,minloglik) in the help variable together with computation times
h <- c(mu_1,mu_2,opt$par,proc_time_total["elapsed"],time_E_step,time_M_step)
names(h) <- c("mu_1", "mu_2", "rho", "theta", "Total", "E_step", "M_step")


Chapter 4. EM Algorithm

60

# Last execution of EM-algorithm to calculate values of mu_1, mu_2 which
# are accurate up to acc_mus
cat("Last loop","\n")
diff_mus <- acc_mus*2

# Define variables such that we can calculate difference between two
# consecutive values mu_1 and mu_2
mu_1_new <- mu_1
mu_2_new <- mu_2

while (diff_mus >= acc_mus){
  mu_1_old <- mu_1_new
  mu_2_old <- mu_2_new
  for(i in 1:N){x_and_y[i,1:4] <- E_step(y_1=y_1[i], y_2=y_2[i], a_1=h["theta"]*h["rho"], a_2 = h["theta"]*(1-h["rho"]), a_3=h["theta"]*(1-
  h["rho"]), b=h["theta"], mu_1=mu_1_old, mu_2=mu_2_old)}
h1 <- M_step(data = x_and_y,N=N)

  # Store new values of mu_1, mu_2
  mu_1_new <- h1["mu_1"]
  mu_2_new <- h1["mu_2"]
  diff_mus <- abs(mu_1_new-mu_1_old) + abs(mu_2_new-mu_2_old)
}
cat(mu_1_new,mu_2_new,"\n")
h["mu_1"] <- mu_1_new
h["mu_2"] <- mu_2_new
return(h)
}

4.3.6 Standard Error

The goal of this section is to write a function StandardError.Bootstrap
to estimate the standard error for estimates ($\hat{\mu}, \hat{\theta}, \hat{\rho}$) by using a bootstrap procedure. The estimates ($\hat{\mu}, \hat{\theta}, \hat{\rho}$) are given to the function as arguments called mumat, rho and theta.

Data is simulated by using the function SimData (see Section 4.4.1) and bootstrap estimates are estimated by using the function EM_estimates (Section 4.3.5). This is done in lines 21-31 below. The results from the $N_{boot}$ ($nrep$) number of bootstrap simulations is stored in the matrix boot.mat.2stage.

Standard errors for the estimates are then estimated by using the following formula (lines 33-37):

$$SE_{N_{boot}}(\hat{\beta}) = \sqrt{\frac{1}{N_{boot} - 1} \sum_{i=1}^{N_{boot}} (\hat{\beta}^*_i - \hat{\beta}^*)^2}, \quad (4.43)$$

where $\beta$ denotes the parameter and $\hat{\beta}^*$ denotes the mean value of the bootstrap estimates (Wehrens et al. 2000).

```r
# Function to calculate standard error by bootstrap
source("SimulationData.R")
source("EmAlgorithm.R")
StandardError.Bootstrap <- function(nrep,mumat,rho,theta,acc_mus, acc_loglik){
```
#Function to perform bootstrap which estimates standard errors
#Input:
# nrep: number of bootstrap simulations
# mumat,rho,theta: estimated values for parameters for which
# variance has to be calculated
#Output:
# Standard errors for mu, rho and theta through bootstrap

T <- ncol(mumat)
N <- nrow(mumat) #Number of studies or centers
boot.mat.2stage <- matrix(NA,nrep,T+2) #Empty matrix to store results in

for (irep in 1:nrep) {
  cat("Replication",irep,"of",nrep,"\n") #So that user can keep track of progress
  #Simulation of data
  data <- SimData(N,T,irep,rho,theta,mumat)
datalog <- data.frame(Time=rep(1:T,N),Study=rep(1:N,each=T),count=as.vector(t(data)),pyrs=1)

  #Estimation of parameters for simulated data
  estimates <- EM_estimates(datalog, N, rho_IV=rho, ksi_IV=theta, acc_mus, acc_loglik)
  print("Estimates")
  print(estimates[1:(T+2)])
  boot.mat.2stage[irep,] <- estimates[1:(T+2)]
}

#Calculate standard errors
SE <- rep(0,T+2)
for(i in 1:(T+2)){
  SE[i] <- sqrt(sum((boot.mat.2stage[,i] - mean(boot.mat.2stage[,i]))^2)/(nrep-1))
}
names(SE) <- c("mu1","mu2","rho","th")
return(SE)

4.4 Simulation Study

Five research questions were defined:

1. How well does the EM-algorithm work on different configurations of \( \rho \) and \( \xi \) compared to the two-stage procedure?

2. How well does the EM-algorithm work on different configurations of \( \mu_1 \) and \( \mu_2 \) compared to the two-stage procedure? In particular, how well does the algorithm work on large values of \( \mu_1 \) and \( \mu_2 \)? (Because this proposed a problem with the method of Henderson and Shimakura [2003].)

3. How does the number of clusters 'N' influence the performance of the EM-algorithm compared to the two-stage procedure?

4. How is the EM-algorithm influenced by the initial estimates for \( \rho \) and \( \xi \)?

5. How does the EM-algorithm perform in terms of computation time compared to the two-stage method?
See table 4.4 for a summary of the 13 simulation studies which were conducted to answer these questions. To make the simulation replicable for other researchers, the `set.seed()` function was used. R makes use of a pseudo-random number generator to generate random numbers. The ‘seed’ can be given to R by using this function. This means that if you simulate data twice and define the same seed before simulating data, you get the same results. The first simulation of a study started with seed ‘Start.seed’. Before starting the next simulation, the seed is increased by 1.

Performance of the EM-algorithm and the two stage method were investigated by calculating the mean bias and the root mean square error (RMSE) for the model parameters. The mean bias is defined as follows:

$$\text{Bias}(\theta) = \frac{1}{N_{\text{rep}}} \sum_{i=1}^{N_{\text{rep}}} (\hat{\theta}_i - \theta)$$  \quad (4.44)

where $N_{\text{rep}}$ denotes the number of simulated samples, $\theta$ is the true value of the parameter of interest, and $\hat{\theta}$ is the estimate of $\theta$ based on sample $i$. The RMSE is defined as follows:

$$\text{RMSE}(\theta) = \sqrt{\frac{1}{N_{\text{rep}}} \sum_{i=1}^{N_{\text{rep}}} (\hat{\theta}_i - \theta)^2}.$$  \quad (4.45)

We can then investigate whether the two methods have systematic errors by examining the former, and investigate how far the estimates are off from the true values on average by examining the latter. To answer the fifth research question, execution times of different parts of the EM-algorithm are reported (see Section 4.3).

The simulations were carried out on a personal computer with a 2.50 GHz processor. As each simulation study took approximately 7 hours to execute, some simulation studies are used to answer multiple research questions. Each simulation study consists of 500 simulations. This number was determined by first performing 1000 repetitions for a single simulation study and then examining at what number of simulations the mean bias and RMSE seemed to have converged. A margin was added.

In Section 4.4.1, the R-code used to conduct the simulation is discussed. In Section 4.4.2, the results of the simulation study are given.

### 4.4.1 R-code

The count data was simulated with R-code written by Fiocco et al. (2009b). See Section 3.2 for an explanation on how the data is simulated. See Appendix A.3 for the code.

The R-code written by Fiocco et al. (2009b) to implement the two-stage procedure is included in Appendix A.5. For details on this estimation procedure, see their article.
Chapter 4. EM Algorithm

4.4. Start.

\[ \rho \xi (\mu_1, \mu_2) N \rho_{IV} \xi_{IV} \]

\begin{table}[h]
\begin{tabular}{cccccccc}
Start.seed & \rho & \xi & (\mu_1, \mu_2) & N & \rho_{IV} & \xi_{IV} & Research Question \\
0 & 0.1 & 1 & (20,20) & 20 & 0.5 & 1 & x \\
1000 & 0.5 & 1 & (20,20) & 20 & 0.5 & 1 & x x x x x \\
2000 & 0.9 & 1 & (20,20) & 20 & 0.5 & 1 & x \\
3000 & 0.5 & 0.25 & (20,20) & 20 & 0.5 & 1 & x \\
4000 & 0.5 & 0.5 & (20,20) & 20 & 0.5 & 1 & x \\
5000 & 0.5 & 1 & (10,10) & 20 & 0.5 & 1 & x x \\
6000 & 0.5 & 1 & (50,50) & 20 & 0.5 & 1 & x x \\
7000 & 0.5 & 1 & (20,20) & 10 & 0.5 & 1 & x x \\
8000 & 0.5 & 1 & (20,20) & 50 & 0.5 & 1 & x x \\
9000 & 0.5 & 1 & (20,20) & 20 & 0.1 & 1 & x \\
10000 & 0.5 & 1 & (20,20) & 20 & 0.9 & 1 & x \\
11000 & 0.5 & 1 & (20,20) & 20 & 0.5 & 0.25 & x \\
12000 & 0.5 & 1 & (20,20) & 20 & 0.5 & 0.5 & x \\
\end{tabular}
\end{table}

\textbf{Table 4.4:} Table with simulation studies. Column 1 gives the seed with which is started. Columns 2-5 give the ‘true’ parameter values. Columns 6-7 give the starting values (Initial Values) for $\rho$ and $\xi$ for the EM-algorithm. The last 5 columns indicate which simulation study is used to assess which research question.

The two \texttt{R}-files which perform the simulation study are included in Appendix A.1 and Appendix A.2. The latter contains a function which performs a single simulation study, and returns the results. The former calls to this function for each of the simulation studies. This code is not discussed in this thesis, as there is no novelty to doing simulation studies.

4.4.2 Results

The results of the simulation study are shown in tables 4.5 through 4.9. The order of the tables corresponds to the order of the research questions stated in the beginning of this section.

The main goal of this simulation study was to compare the EM-algorithm to the two-stage method. Therefore, we made use of coloring to show where the EM-algorithm performed better than the two-stage method or the other way around. The mean bias or RMSE for the EM-algorithm for a certain parameter and simulation study is colored green when it is both 0.1 lower and 10% less than the corresponding value of the two-stage method. The corresponding value is colored red. The same is done the other way around. The threshold of 0.1 was included to prevent exaggeration of differences for values of the bias and RMSE close to zero.
<table>
<thead>
<tr>
<th>$\rho$</th>
<th>$\xi$</th>
<th>$\mu_1$</th>
<th>$\mu_2$</th>
<th>$\rho_{IV}$</th>
<th>$\xi_{IV}$</th>
<th>$N$</th>
<th>$\hat{\rho}$</th>
<th>$\hat{\xi}$</th>
<th>$\hat{\mu}_1$</th>
<th>$\hat{\mu}_2$</th>
<th>EM bias</th>
<th>RMSE</th>
<th>EM bias</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>1</td>
<td>20</td>
<td>20</td>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td>0.042</td>
<td>-0.039</td>
<td>0.044</td>
<td>0.150</td>
<td>0.163</td>
<td>0.216</td>
<td>4.601</td>
<td>4.615</td>
</tr>
<tr>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td>20</td>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td>-0.032</td>
<td>-0.052</td>
<td>-0.209</td>
<td>-0.346</td>
<td>0.190</td>
<td>0.249</td>
<td>4.407</td>
<td>4.238</td>
</tr>
<tr>
<td>0.9</td>
<td>1</td>
<td>20</td>
<td>20</td>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td>0.006</td>
<td>-0.031</td>
<td>0.235</td>
<td>0.242</td>
<td>0.075</td>
<td>0.299</td>
<td>4.693</td>
<td>4.636</td>
</tr>
<tr>
<td>0.5</td>
<td>0.25</td>
<td>20</td>
<td>20</td>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td>-0.016</td>
<td>-0.009</td>
<td>0.108</td>
<td>0.076</td>
<td>0.239</td>
<td>0.071</td>
<td>2.574</td>
<td>2.550</td>
</tr>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>20</td>
<td>20</td>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td>-0.026</td>
<td>-0.026</td>
<td>0.108</td>
<td>-0.169</td>
<td>0.210</td>
<td>0.127</td>
<td>2.574</td>
<td>3.400</td>
</tr>
</tbody>
</table>

Table 4.5: Results of the simulation study for the first research question: different configurations of $\rho$ and $\xi$. 
<table>
<thead>
<tr>
<th>$\rho$</th>
<th>$\xi$</th>
<th>$\mu_1$</th>
<th>$\mu_2$</th>
<th>$\rho_{IV}$</th>
<th>$\xi_{IV}$</th>
<th>$N$</th>
<th>EM bias</th>
<th>RMSE</th>
<th>two-stage bias</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td></td>
<td>-0.022</td>
<td>0.214</td>
<td>-0.034</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\rho}$</td>
<td></td>
<td>$\xi$</td>
<td>0.270</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_1$</td>
<td>0.025</td>
<td>2.280</td>
<td>0.053</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_2$</td>
<td>-0.019</td>
<td>2.303</td>
<td>0.004</td>
</tr>
<tr>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td>20</td>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td></td>
<td>-0.032</td>
<td>0.190</td>
<td>-0.050</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\rho}$</td>
<td></td>
<td>$\xi$</td>
<td>0.249</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_1$</td>
<td>-0.209</td>
<td>4.407</td>
<td>-0.237</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_2$</td>
<td>-0.346</td>
<td>4.238</td>
<td>-0.244</td>
</tr>
<tr>
<td>0.5</td>
<td>1</td>
<td>50</td>
<td>50</td>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td></td>
<td>-0.032</td>
<td>0.190</td>
<td>-0.061</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\rho}$</td>
<td></td>
<td>$\xi$</td>
<td>0.261</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_1$</td>
<td>-0.212</td>
<td>10.916</td>
<td>-0.144</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_2$</td>
<td>-0.326</td>
<td>11.741</td>
<td>-0.455</td>
</tr>
</tbody>
</table>

Table 4.6: Results of the simulation study for the second research question: different configurations of $\mu_1$ and $\mu_2$.

<table>
<thead>
<tr>
<th>$\rho$</th>
<th>$\xi$</th>
<th>$\mu_1$</th>
<th>$\mu_2$</th>
<th>$\rho_{IV}$</th>
<th>$\xi_{IV}$</th>
<th>$N$</th>
<th>EM bias</th>
<th>RMSE</th>
<th>two-stage bias</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td>20</td>
<td>0.5</td>
<td>1</td>
<td>10</td>
<td></td>
<td>-0.023</td>
<td>0.261</td>
<td>-0.053</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\rho}$</td>
<td></td>
<td>$\xi$</td>
<td>0.379</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_1$</td>
<td>0.131</td>
<td>6.804</td>
<td>0.096</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_2$</td>
<td>0.106</td>
<td>6.297</td>
<td>0.197</td>
</tr>
<tr>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td>20</td>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td></td>
<td>-0.032</td>
<td>0.190</td>
<td>-0.050</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\rho}$</td>
<td></td>
<td>$\xi$</td>
<td>0.249</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_1$</td>
<td>-0.209</td>
<td>4.407</td>
<td>-0.237</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_2$</td>
<td>-0.346</td>
<td>4.238</td>
<td>-0.244</td>
</tr>
<tr>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td>20</td>
<td>0.5</td>
<td>1</td>
<td>50</td>
<td></td>
<td>0.002</td>
<td>0.120</td>
<td>-0.007</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\rho}$</td>
<td></td>
<td>$\xi$</td>
<td>0.156</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_1$</td>
<td>-0.075</td>
<td>3.060</td>
<td>-0.040</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_2$</td>
<td>-0.096</td>
<td>2.894</td>
<td>-0.076</td>
</tr>
</tbody>
</table>

Table 4.7: Results of the simulation study for the third research question: different number of clusters $N$. 

4.5 Conclusions

The results in Section 4.4.2 show that for two time points \((T = 2)\), the two-stage procedure and EM-algorithm perform similarly in terms of mean bias and RMSE. The results from the second simulation study (where set.seed
is 1000) contains colored values, and these are repeated in Tables 4.5 through 4.8. This overemphasizes the number of colored values. Out of the thirteen simulation studies performed, only 4 out of 52 values for the mean bias were colored. For the RMSE, zero values were colored. Also, from Table 4.8 we can conclude that the EM-algorithm is not influenced by initial estimates for \( \rho \) and \( \xi \), as it performs similarly to the two-stage method on all configurations.

From Table 4.5 we can conclude that the RMSE increases for both methods when the variance of the frailty terms \( \xi \) increases. An explanation for this might be found in the structure of the model. The count data \( Y_{ij} \) for \( i = (1, \ldots, N) \) and \( j = (1, \ldots, T) \) are distributed as \( \text{NB}(\mu_i, \theta) \) (see Lemma 5.2.1). The variance is thus given by:

\[
\text{var}(Y_{ij}) = \mu_i + \mu_i^2 \xi. \tag{4.46}
\]

Variance in the data can thus be explained by both the hazard rate and the frailty variance. When the variance in the count data is higher, it might be more difficult to determine where it comes from.

In Table 4.6 we see that the RMSE seems to increase linearly with the hazard rates \( \mu_1 \) and \( \mu_2 \) for both methods. For all three configurations, the RMSE is approximately 25% for both methods.

In Table 4.7 we see that the RMSE decreases when \( N \) increases. This makes sense, as the number of observed counts increases as \( N \) increases. With more data present, you would expect the two methods to perform better, and they do.

From Table 4.9, we can conclude that the E-step takes up most of the computation time of the EM-algorithm. When we combine Table 4.1 and Table 4.9 we see that the computation time of the E-step increases approximately linearly with the number of summands. If we assume that the computation time of the E-step increases linearly with the number of summands, and the rest of the EM-algorithm performs similarly, the E-step would take approximately 500 times slower for \( T = 3 \) and \( y = 10 \) than for \( T = 2 \) and \( y = 50 \). This would mean that the E-step would take approximately 24 hours on a computer with a 2.50 GHz processor. The true computation time will probably be higher, because:

- the number of gamma variables to be estimated in the E-step increases from 3 to 6;
- the complexity of the summands increases. Instead of the product of 3 fractions, the product over 6 fractions has to be taken;
- other parts of the algorithm might take up a lot more computation time.

An attempt has been made to write a R-script for \( T = 3 \), but this failed in an early stage due to computational infeasibility. It might be possible to perform single iterations of the EM-algorithm for \( T = 3 \) on a super computer, but performing simulation studies will probably take up too much time.
Chapter 5

Flexible Two-stage

The goal of this chapter is to rewrite the algorithm of Fiocco et al. such that the distance between time points does not have to be the same. In Section 5.1 the available literature on this subject is discussed and it is explained how the latter method can be improved. In Section 5.2 algebraic expressions to execute the algorithm are derived. In Section 5.3 the R-code to implement the algorithm is discussed. In Section 5.4 a simulation study is carried out to assess performance of the algorithm. In Section 5.5 conclusions are drawn on the performance of the algorithm.

5.1 Introduction

The notation used in this section is introduced in Section 2.4.3. The idea behind the flexible two-stage method is that instead of maximizing the observed likelihood (2.22), we maximize a composite likelihood function (see Section 2.4.6). Composite likelihood procedures based on marginal and bivariate distributions are consistent under some assumptions Varin (2008). Henderson and Shimakura proposed a composite likelihood procedure based on all pairs of observations. The model is estimated by maximizing:

\[ \ell_2(\mu, \rho, \theta | Y) = \sum_{i=1}^{N} \sum_{s=1}^{T-1} \sum_{t=s+1}^{T} \ell_{\text{obs}}(\mu, \rho, \theta | y_{is}, y_{it}) \]  

(5.1)

over \( \mu, \rho, \theta \). The multivariate gamma process was constructed by making use of a particular relation between the multivariate normal and multivariate gamma distribution. The expression for \( \ell_{\text{obs}}(\mu, \rho, \theta | y_{is}, y_{it}) \) was based on this multivariate gamma process. However, as explained in Section 1.2 when counts are high, rounding errors can occur. Therefore, Fiocco et al. proposed a two-stage composite likelihood procedure based on a different construction of the gamma frailty process. In the first step, estimates for \( \mu, \theta \) are found by maximizing a composite likelihood function based on the marginal distribution of all counts:

\[ \ell_1(\mu, \theta | Y) = \sum_{i=1}^{N} \sum_{t=1}^{T} \ell_{\text{obs}}(\mu, \theta | y_{it}) \]  

(5.2)

over \( \mu, \theta \). In the second stage, an estimate for \( \rho \) is found by maximizing:
\[
\ell_2(\rho|Y, \hat{\mu}, \hat{\theta}) = \sum_{i=1}^{N} \sum_{s=1}^{T-1} \sum_{t=s+1}^{T} \ell_{\text{obs}}(\rho|y_{is}, y_{it}, \hat{\mu}, \hat{\theta})
\]  
(5.3)

over \(\rho\). The authors used model 1 (see Section 3.2) to construct the gamma process and to derive expressions for \(\ell_1(\mu, \theta|Y)\) and \(\ell_2(\rho|Y, \hat{\mu}, \hat{\theta})\). An advantage of the latter method is that it can handle big counts. A disadvantage is that model 1 assumes that the distance between the time points \(t_1, \ldots, t_T\) is equal. Therefore, it can only be applied when the length of the intervals is the same. In some applications, it may not be possible to make this assumption. Therefore, we propose to use model 2 (see Section 3.3) to construct the gamma process and derive expressions for \(\ell_1(\mu, \theta|Y)\) and \(\ell_2(\rho|Y, \hat{\mu}, \hat{\theta})\) by using this new, more flexible gamma process. We named this method the 'Flexible Two-stage' method.

Note that in this context, \(\mu\) is a matrix \((\mu_{it})\) where \(i = (1, \ldots, N)\) denotes the cluster and \(t = (1, \ldots, T)\) the time interval, while in Chapter 4, \(\mu\) is a vector \((\mu_1, \ldots, \mu_T)\). In this way, it is relatively easy to incorporate covariates (see Section 2.2) into the flexible two-stage procedure.

Standard errors for \((\mu, \rho, \theta)\) can be estimated by using a parametric bootstrap approach. Let \((\hat{\mu}, \hat{\theta}, \hat{\rho})\) be the estimated parameters from the original data. For one bootstrap data set, we proceed as follows:

1. Generate \((z_{it}^*)\) with \(i = (1, \ldots, N)\) and \(t = (1, \ldots, T)\) from a multivariate gamma distribution with parameters \(\theta = \hat{\theta}\) and \(\rho = \hat{\rho}\).
2. Generate \((y_{it}^*)\) with \(i = (1, \ldots, N)\) and \(t = (1, \ldots, T)\) from \(y_{it}^* \overset{d}{=} \text{Pois}((\hat{\mu}_{it}z_{it}^*)\).
3. Estimate \((\mu, \theta, \rho)\) from the bootstrap data set \(y^*\) by using the flexible two-stage method, obtaining \((\hat{\mu}^*, \hat{\theta}^*, \hat{\rho}^*)\).

Let \(N_{\text{boot}}\) be the number of bootstrap simulations giving the estimates \((\hat{\mu}^*, \hat{\theta}^*, \hat{\rho}^*)\) for each bootstrap simulation. These estimates can then be used to obtain standard errors or confidence intervals using standard procedures (Wehrens et al. 2000).

5.2 Algebraic Expressions

The goal of this section is to find algebraic expressions for \(\ell_1(\mu, \theta|Y)\) and \(\ell_2(\rho|Y, \hat{\mu}, \hat{\theta})\), where the frailty term are modeled as in Section 3.3.

5.2.1 Marginal Distribution

The goal of this section is to derive an algebraic expression for \(\ell_1(\mu, \theta|Y)\). The result is given in Theorem 5.2.2. In this theorem, we make use of a general property of Poisson-gamma mixture distributions. This property is stated in Lemma 5.2.1.
Lemma 5.2.1. Let $Z \overset{d}{=} \Gamma(\alpha, \beta)$ and let $Y|Z \overset{d}{=} P_{\text{pois}}(\mu Z)$, where $\mu$, $\alpha$ and $\beta$ are parameters. Then $Y \overset{d}{=} \text{NB}(\mu^2, \alpha)$. Here, $P_{\text{pois}}$ denotes the Poisson distribution and $\text{NB}(\mu, \theta)$ denotes the negative binomial distribution with parametrization:

$$P_{\text{NB}}(y; \mu, \theta) = \frac{\Gamma(y + \alpha)}{y! \Gamma(\alpha)} \left( \frac{\beta}{\beta + \mu} \right)^y (\theta + \mu)^\theta.$$

Proof. 

$$P(Y = y) = \mathbb{E}_z[P(Y = y|Z = z)] = \int_0^\infty P(Y = y|Z = z) f(z) dz = \int_0^\infty e^{-\mu z} \left( \frac{\beta^a}{y!} \frac{\Gamma(\alpha)}{\Gamma(\alpha)} \right)^{y+1} e^{-\beta z} dz = \frac{\mu^y \beta^a}{y! \Gamma(\alpha)} \int_0^\infty z^{y+a-1} e^{-(\beta+\mu) z} dz = \frac{\mu^y \beta^a}{y! \Gamma(\alpha)} \text{NB}(y; \mu, \alpha, \alpha + \beta).$$

We applied Lemma 4.2.2 to evaluate the integral. This concludes the proof.

Theorem 5.2.2. The composite likelihood function $\ell_1(\mu, \theta|Y)$, where frailty gamma process is constructed as in Section 3.3, is given by:

$$\ell_1(\mu, \theta|Y) = \sum_{i=1}^N \sum_{t=1}^T \log P_{\text{NB}}(y_{it}; \mu_{it}, \theta).$$

Proof. When we apply Lemma 5.2.1 to expression (5.2), where $Y = Y_{it}$, $Z = Z_{it}$, and $\alpha = \beta = \theta$, we get the desired result.

5.2.2 Joint Distribution

The goal of this section is to find an algebraic expression for $\ell_2(\rho|Y, \hat{\mu}, \hat{\theta})$, where the frailty term are constructed as in Section 3.3. We could use the expression found in Section 4.3.4, but we will see that the expression derived in this section has computational advantages.
Lemma 5.2.3. The joint distribution of a pair \((Y_s, Y_t)\) with \(s, t \in \{1, \ldots, T\}\) and \(t \neq s\) is given by:

\[
P(Y_s = y_s, Y_t = y_t) = \sum_{k=0}^{y_s} \sum_{l=0}^{y_t} P_{\text{NB}}(k; \mu_s (1 - \rho^{s,t}), \theta (1 - \rho^{s,t})) *
\]

\[
P_{\text{NB}}(l; \mu_t (1 - \rho^{s,t}), \theta (1 - \rho^{s,t})) *
\]

\[
P_{\text{NB}}(y_s + y_t - k - l; (\mu_s + \mu_t) \rho^{s,t}, \theta \rho^{s,t}) *
\]

\[
P_{\text{BIN}}(y_s - k; \mu_s + y_t - k - l, \frac{\mu_s}{\mu_s + \mu_t})
\]

Here, \(\rho^{s,t} = \rho^{|t_s - t_l|}\) is defined as in Section 3.3.

Proof. Instead of integrating out the frailty terms from the full likelihood function (see Theorem 4.2.4), we can also determine an expression for the observed data likelihood by differentiating Laplace transform of \(L(\mu) = \mathbb{E}[\exp(-\mu^T Z)]\) (Henderson and Shimakura 2003):

\[
P(Y_1 = y_1, \ldots, Y_T = y_T) = \left( \prod_{t=1}^{T} \frac{\mu_t^{y_t}}{y_t!} \right) \times \mathbb{E}[Z_1^{y_1} \cdots Z_T^{y_T}] \exp(-\mu^T Z). \tag{5.6}
\]

The joint distribution of a pair \((Y_s, Y_t)\) where \(s \neq t\) is then given by:

\[
P(Y_s = y_s, Y_t = y_t) = \frac{\mu_s^{y_s} \mu_t^{y_t}}{y_s! y_t!} \mathbb{E}[Z_s^{y_s} Z_t^{y_t} e^{-\mu_s Z_s} e^{-\mu_t Z_t}]. \tag{5.7}
\]

From the proof of Theorem 3.3 (see expression (3.9)) we can rewrite \(Z_s = X_0 + X_s\) and \(Z_t = X_0 + X_t\), where \(X_0, X_s\) and \(X_t\) are defined as:

\[
X_0 = Z_{s\cap t} = \sum_{j=i+1}^{s} X_{ij} \overset{d}{=} \Gamma(\rho^{s,t} \alpha, \beta)
\]

\[
X_s = Z_{s\setminus t} = \sum_{j=t+1}^{s} X_{ij} \overset{d}{=} \Gamma((1 - \rho^{s,t}) \alpha, \beta), \tag{5.8}
\]

\[
X_t = Z_{t\setminus s} = \sum_{j=s+1}^{t} X_{ij} \overset{d}{=} \Gamma((1 - \rho^{s,t}) \alpha, \beta)
\]

where \(X_0, X_s\) and \(X_t\) are independent. By applying the linearity of the expected value operator and the binomial theorem, expression (5.7) becomes:

\[
= \frac{\mu_s^{y_s} \mu_t^{y_t}}{y_s! y_t!} \mathbb{E}[(X_0 + X_s)^{y_s} (X_0 + X_t)^{y_t} e^{-\mu_s (X_0 + X_s)} e^{-\mu_t (X_0 + X_t)}]
\]

\[
= \sum_{k=0}^{y_s} \sum_{l=0}^{y_t} \mathbb{E}[e^{-\mu_s X_k} e^{-\mu_t X_l} e^{-\mu_s + \mu_t} X_0 X_s^{y_s - k} X_t X_0^{y_t - l} \frac{X_k X_0^{y_s - k} X_l X_0^{y_t - l}}{k!(y_s - k)! l!(y_t - l)!} \mu_s^{y_s} \mu_t^{y_t}]. \tag{5.9}
\]
Since $X_0$, $X_s$, and $X_t$ are independent:
\[
\begin{align*}
\ell_2(\rho|Y, \hat{\mu}, \hat{\theta}) &= \sum_{i=1}^{N} \sum_{s=1}^{T-1} \sum_{l=1}^{T} \sum_{k=0}^{Y_{is}} \sum_{l=0}^{Y_{it}} P_{NB}(k; \hat{\mu}_{is}(1 - \rho^{s,t}), \hat{\theta}(1 - \rho^{s,t})) \times \\
P_{NB}(l; \hat{\mu}_{it}(1 - \rho^{s,t}), \hat{\theta}(1 - \rho^{s,t})) \times \\
P_{NB}(y_{is} + y_{it} - k - l; (\hat{\mu}_{is} + \hat{\mu}_{it})\rho^{s,t}, \hat{\theta}\rho^{s,t}) \times \\
P_{BIN}(y_{is} - k; y_{is} + y_{it} - k - l; \frac{\hat{\mu}_{is}}{\hat{\mu}_{is} + \hat{\mu}_{it}})
\end{align*}
\]

Proof. Results follow by applying Lemma 5.2.3 to expression (5.3). \qed

Theorem 5.2.4. The composite likelihood function $\ell_2(\rho|Y, \hat{\mu}, \hat{\theta})$, where frailty gamma process is constructed as in Section 3.3 is given by:

\[
\ell_2(\rho|Y, \hat{\mu}, \hat{\theta}) = \sum_{k=0}^{y_s} \sum_{l=0}^{y_t} \mathbb{E}[e^{\mu_s X_0} (\mu_s X_0)^k k!] \mathbb{E}[e^{\mu_t X_t} (\mu_t X_t)^l l!] \times \\
\mathbb{E}[e^{-(\mu_s + \mu_t) X_0} ((\mu_s + \mu_t) X_0)^{y_s} + y_t - k - l] \\
(y_s + y_t - k - l)! (\mu_s + \mu_t)^{y_s} - k (\frac{\mu_t}{\mu_s + \mu_t})^{y_t - l} (y_s - k)!(y_t - l)! \times \\
\mathbb{E}[P_{pois}(k; \mu_s X_s) \mathbb{E}[P_{pois}(l; \mu_t X_t|X_t)] \times \\
P_{BIN}(y_s - k; y_s + y_t - k - l; \frac{\mu_s}{\mu_s + \mu_t})
\]

Proof. Results follow by applying Lemma 5.2.3 to expression (5.3). \qed

In $\mathbb{R}$, it is easier to evaluate the expression in Theorem 5.2.4 than expression (4.3.4), because there are stable and efficient R-functions available to compute the distribution function for the negative binomial distribution.

### 5.3 R-code

In this section, we discuss the R-code for the implementation of the flexible two-stage method. Since the R-code is one of the novelties of this thesis, it is discussed in detail. The complete code is given in Appendix B.

In Sections 5.3.1 and 5.3.2 simulations from the model and implementation of $\ell_2(\rho|Y, \hat{\mu}, \hat{\theta})$ are respectively discussed. In Section 5.3.3 we discuss the main function implemented for the flexible two-stage method. We did not write a separate R-function to evaluate $\ell_1(\mu, \theta|Y)$, because evaluation of this expression can be done in a few lines of code, whereas evaluation of $\ell_2(\rho|Y, \hat{\mu}, \hat{\theta})$ is more complex. In Section 5.3.4 the code to calculate the
standard error by using bootstrap is discussed. The input is given to the main function in a long format (see Section 4.3.1).

5.3.1 Simulation of Data

The goal of this section is to write a \texttt{R}-function which simulates data from the model. Given \((\mu, \theta, \rho)\), this function has to simulate \((y_{it})\), where \(i = (1, \ldots, N)\) and \(t = (1, \ldots, T)\).

\texttt{Colrep} and \texttt{rowrep} (lines 7-27 below in this section) are functions which repeat a vector \(v\), \(n\) times and store the result in a matrix. The former stores the vector in the columns, the latter in the rows. The function \texttt{corr.ARR} (lines 29-40) calculates the correlation matrix \((\rho^{ij}) = (\rho^{\mid t_i - t_j \mid})\) for \(1 \leq i, j \leq T\) given a value for \(\rho\) and the vector of time points \(t_1, \ldots, t_T\), which are stored in \(\texttt{TP}\).

The function \texttt{shape.matrix} (lines 42-79) calculates the shape parameters of the \(X_{ij}\)’s (see (3.5)), given values for \(\theta, \rho\) and the time points \(t_1, \ldots, t_T\). The result is stored in a \(T \times T\) matrix \(\texttt{shape.mat}\). As the \(X_{ij}\)’s are only defined for \(1 \leq ij \leq T, j \geq i\), the matrix is filled with zero’s for all \(j < i\).

The function \texttt{sum.frailty} (lines 71-88) sums over the gamma random variables \(X_{ij}\) to calculate the values of the frailty terms \((Z_1, \ldots, Z_T)\) (3.7).

The function \texttt{SimFrailty} (lines 90-116) simulates frailty terms \((Z_1, \ldots, Z_T)\). For given \(\rho\) and \(\theta\), it calculates the shapes of the \(X_{ij}\)’s using previously named functions (line 103), and then simulates outcomes for the \(X_{ij}\)’s by using the \texttt{R}-function \texttt{rgamma} (lines 106-111). The simulated values for \((Z_1, \ldots, Z_T)\) are then calculated (line 113) and stored in a \(N \times T\) matrix \(Z\). This procedure is repeated \(N\) times (line 105), until the frailty terms have been simulated for all \(N\) units.

The function \texttt{SimData} (lines 118-141) simulated longitudinal count data with serially correlated gamma frailty. It uses all previously named functions (directly or indirectly). In line 129, the frailty terms are simulated by using \texttt{SimFrailty}. By using these simulated frailty terms, the count data is generated by using the \texttt{R}-function \texttt{rpois} (line 136), by making use of expression (2.19). Results are stored in a long format (line 139).
#function to repeat vector v n times
# Input:
# v: vector
# n: number of rows in the matrix
# Output:
# matrix of repeated rows, dimension: length(v) by n
return(t(colrep(v,n)))

corr.AR <- function(rho,TP){
#Function to create correlations matrix for autoregressive models
#Input:
# rho: correlation parameter
# TP: time points
#Output:
# T by T matrix R where R[i,j] is the correlation between Z_i and Z_j
T <- length(TP)
R <- outer(1:T,1:T, FUN <- function(r,c) rho^(abs(TP[r]-TP[c])))
return(R)
}

shape.matrix <- function(rho,theta,TP){
#Function to create matrix containing shapes of X_ij's
#Input:
# rho and theta: values of model parameter parameters
# TP: vector of time points
#Output:
# T by T matrix shape.mat where the shape.mat[i,j] is the shape of X_ij
T <- length(TP) #Number of time points
rho.mat <- corr.AR(rho,TP) #Correlation matrix
shape.mat <- matrix(0,T,T) #Empty shape matrix

#The shape for i=1, j=1
shape.mat[1,1] <- rho.mat[1,1]*theta
#The shape for i=1, j=2:T
for(j in 2:T){shape.mat[1,j] <- rho.mat[1,j] - rho.mat[1,j+1]*theta}
#The shape for i=2:T, j=1
for(i in 2:T){shape.mat[i,1] <- rho.mat[i,1] - rho.mat[i-1,1] - rho.mat[i,1+1] + rho.mat[i-1,1+1]*theta}
if(T != 2){
for(i in 2:(T-1)){
for(j in i:(T-1)){shape.mat[i,j] <- rho.mat[i,j] - rho.mat[i-1,j] - rho.mat[i,j+1] + rho.mat[i-1,j+1]*theta}
}
}
return(shape.mat)
}

sum.frailty <- function(gamma.mat){
#This function sums over the elements of X_ij to calculate Z_t
#Z_t = sum_{i=1}^{T} sum_{j=1}^{T} X_{i,j}
# Input:
# T by T matrix containing simulated values of X_ij
# Output:
# Vector Z of length T where the t'th element is Z_t
T <- nrow(gamma.mat)
Z <- rep(0,T)
for(t in 1:T){
  sum = 0
  for(j in t:T){
    for(i in 1:t){sum = sum + gamma.mat[i,j]}
  }
  Z[t] <- sum
}
return(Z)
}

SimFrailty <- function(N,TP,irep,rho,theta){
Chapter 5. Flexible Two-stage

5.3.2 Likelihood Pairs of Observations

The goal of this section is to write a R-function which can calculate the likelihood $\ell_2(\rho | Y, \hat{\mu}, \hat{\theta})$ for a value $\rho$. Recall that:

---

#Function which simulates serially correlated gamma frailty terms
#Input:
# N, TP: number of studies and time points
# irep: number of previous simulations (for input in set.seed())
# rho,theta: true values of model parameters
#Output:
# matrix with N rows and T columns containing simulated Z_1,...,Z_T

set.seed(irep) #This is done so that results can be replicated
T <- length(TP)
Z <- matrix(0,nrow=N,ncol=T) #Empty matrix to store Z_1,...,Z_T for every study
shape.mat <- shape.matrix(rho,theta,TP) #Calculate shapes of X_ij’s

for(n in 1:N){
  for(i in 1:T){
    for(j in i:T){
      #Generate X_ij’s
      gamma.mat[i,j] <- rgamma(n=1,shape=shape.mat[i,j],rate=theta)
    }
  }
  #Calculate frailty’s by summing over X_ij’s
  Z[n,] <- sum.frailty(gamma.mat)
}
return(Z)

SimData <- function(N,TP,irep,rho,theta,mumat){
  #Function which simulates longitudinal count data with serially correlated
gamma terms
  #Input:
  # number of studies 'N', time points 'TP'
  # number of previous simulations 'irep' (for input in set.seed())
  # true values of model parameters 'rho', 'theta'
  #Output:
  # matrix with N rows and T columns containing simulated y_1,...,y_T for N studies

  #Simulate the frailty’s
  Z <- SimFrailty(N,TP,irep,rho,theta)
  #Simulate the count data y_1,dots,y_T
  T <- length(TP)
events <- matrix(0,N,T) #Empty matrix to store count data
  #Simulate count data
  for(n in 1:N){events[n,] <- rpois(T,mumat[n,]*Z[n,])}
  #Store data in data long frame
  events.DL <- data.frame(Time=rep(TP,N),Study=rep(1:N,each=T),count=as.vector(t(events))
  return(events.DL)
}
\[ \ell_{\text{obs}}(\rho|y_{is}, y_{it}, \hat{\mu}, \hat{\theta}) = \sum_{k=0}^{y_{is}} \sum_{l=0}^{y_{it}} P_{NB}(k; \hat{\mu}_{is}(1 - \rho^s), \hat{\theta}(1 - \rho^s))* \\
\quad P_{NB}(l; \hat{\mu}_{it}(1 - \rho^s), \hat{\theta}(1 - \rho^s))* \\
\quad P_{NB}(y_{is} + y_{it} - k - l; (\hat{\mu}_{is} + \hat{\mu}_{it})\rho^s, \hat{\theta}\rho^s)* \\
\quad P_{BIN}(y_{is} - k; y_{is} + y_{it} - k - l, \frac{\hat{\mu}_{is}}{\hat{\mu}_{is} + \hat{\mu}_{it}}) \]

and that

\[ \ell_2(\rho|Y, \hat{\mu}, \hat{\theta}) = N \sum_{s=1}^{T-1} \sum_{t=s+1}^{T} \ell_{\text{obs}}(\rho|y_{is}, y_{it}, \hat{\mu}, \hat{\theta}). \]

The function \texttt{loglikpair} (lines 8-43 below in this section) calculates the former, the function \texttt{loglikrho} (lines 45-71) calculates the latter and uses \texttt{loglikpair}. The function \texttt{loglikpair} is called quite often. Every time \texttt{loglikrho} is called, the function \texttt{loglikpair} is called \(N * \frac{1}{2}(T^2 - T)\) times. It is therefore important that this function is computationally efficient.

We evaluate the three negative binomial terms in \( \ell_{\text{obs}}(\rho|y_{is}, y_{it}, \hat{\mu}, \hat{\theta}) \) by making use of the function \texttt{dnbinom} and the binomial term with the function \texttt{dbinom}. These two functions are provided by \texttt{R}, and they are computationally efficient. The first negative binomial term does not change for fixed \(k\). It is therefore computationally inefficient to evaluate it for every combination of \(k\) and \(l\). The same is true for the second negative binomial term. Therefore, we proceed as follows:

before summing over the elements, the evaluation of the four distribution functions is stored in matrices \(P1, P2, P3\) and \(P4\). Thus, \(P1\) is a \(y_{is} + 1 \times y_{it} + 1\) matrix, where the element \((P1_{kj})\) is

\( P_{NB}(k; \hat{\mu}_{is}(1 - \rho^s), \hat{\theta}(1 - \rho^s)) \)

The advantage of this method is that for the matrices \(P1\) and \(P2\), we can calculate one of the columns or rows and then simply repeat it \(y_{it} + 1\) or \(y_{is} + 1\) to calculate the matrix \(P1\) or \(P2\) (respectively). These matrices are computed in lines 21-38. In lines 40-41, these matrices are multiplied with each other, the sum is taken over the elements and minus the log is taken over the likelihood.

```r
source("SimulationData.R")

loglik.pair <- function(y_s,y_t,mu_s,mu_t,rho_st,theta){
# Function which calculates the contribution to the minus log likelihood
# of a pair of observations
# Input:
# y_s,y_t: number of counts for time points s and t
# mu_s,mu_t,theta: estimated values of mu_s, mu_t and theta
# rho_st: correlation between Z_s and Z_t
# Output:
# Minus log likelihood for pair of observations
```
mu_st <- mu_s + mu_t #For convenience of notation
rho <- rho_st #For convenience of notation

#First NB-term (NB = Negative Binomial)
Pist1 <- dnbionm(y_s:0, size = theta * (1 - rho), mu = mu_s * (1 - rho))
P1 <- colrep(Pist1, y_t + 1)

#Second NB-term
Pist2 <- dnbionm(y_t:0, size = theta * (1 - rho), mu = mu_t * (1 - rho))
P2 <- rowrep(Pist2, y_s + 1)

#Help matrices
outerm <- outer(0:y_s, 0:y_t, "+")
outerv <- as.vector(outerm)
helpv <- as.vector(colrep(0:y_s, y_t + 1))

#Third NB-term
P3 <- matrix(dnbionm(outerv, size = theta * rho, mu = mu_st * rho), y_s + 1, y_t + 1)

#The binomial-term
P4 <- matrix(dbionm(helpv, outerv, mu_s/mu_st), y_s + 1, y_t + 1)

#Calculation of likelihood
lik <- P1 * P2 * P3 * P4
return(minloglik = -log(sum(lik)))

loglikrho <- function(rho, ymat, mumat, theta, TP=TP){
  #Function which calculates the composite likelihood for all pairs of
  #observations
  #Input: 
  # rho, theta: current estimates for rho and theta
  # ymat: N by T matrix filled with count data
  # mumat: N by T matrix filled with estimates for mu
  # TP: time points
  #Output: 
  # Minus composite log likelihood
  loglik <- 0 #Variable in which output will be stored
  R <- corr.AR(rho,TP) #Calculating correlation matrix
  N <- nrow(ymat)
  T <- ncol(ymat)

  #Sum over all studies
  for (i in 1:N) {
    #Sum over all possible unique pairs s,t, where s,t in (1,...,T)
    for (s in 1:(T - 1)) {
      for (t in (s + 1):T) {
        loglik <- loglik + loglik.pair(ymat[i,s], ymat[i,t], mumat[i,s], mumat[i,t], R[s,t], theta)
      }
    }
  }
  cat(rho,"=>",loglik,"\n") #Print so that user can keep progress
  return(loglik)
}

5.3.3 Estimation

The goal of this section is to write a R-function which estimates \((\mu, \theta, \rho)\) based on longitudinal count data \(y\). It uses the function loglikrho from Section 5.3.2. The procedure results in a function called Estimation.2.stage (see R-code below in this section).
Chapter 5. Flexible Two-stage

In lines 19-21, the first step of the flexible two-stage procedure is executed. The composite likelihood $\ell_1(\mu, \theta | Y) = \sum_{i=1}^{N} \sum_{t=1}^{T} \log P_{NB}(y_{it}; \mu_{it}, \theta)$ is maximized over the parameters $\mu$ and $\theta$. For fixed $\theta$, the negative binomial distribution can be formulated as a generalized linear model (GLM). This is advantageous, as there are standard R-functions to perform regression for generalized linear models. Moreover, in the MASS library by [Venables and Ripley (2002)](https://CRAN.R-project.org/package=MASS), there is a function glm.nb which can also estimate $\theta$. This function builds in extra maximization steps of the log-likelihood with respect to $\theta$ to estimate $\mu$ and $\theta$ jointly.

The function glm.nb can be used to estimate covariates (see Section 2.2). Let $x_{it}$ ($i = 1, \ldots, N$ and $t = 1, \ldots, T$) denote a design vector and $\beta$ a covariate vector. Let the hazard matrix $\mu$ depend on the design vector through the link $\mu_{it} = \exp(x_{it}^T\beta)$; the glm.nb-function can estimate both $\mu$ and $\beta$. As covariates are included in almost every survival analysis, this is a major advantage.

In lines 30-38, an estimate for $\rho$ is found by maximizing $\ell_2(\rho | Y, \hat{\mu}, \hat{\theta})$ (evaluated by using loglikrho) over $\rho$. This is done with the function maximize, provided in R which performs one dimensional optimization.
5.3.4 Standard Error

The goal of this section is to write a \texttt{R}-function to estimate the standard error for estimates \((\hat{\mu}, \hat{\theta}, \hat{\rho})\) by using a bootstrap procedure. The result is the function \texttt{StandardError.Bootstrap} (see \texttt{R}-code below in this section). The estimates \((\hat{\mu}, \hat{\theta}, \hat{\rho})\) are given to the function as arguments called \texttt{mumat}, \texttt{rho} and \texttt{theta}.

Data is simulated by using the function \texttt{SimData} (Section 5.3.1) and bootstrap estimates are calculated by using the function \texttt{Estimation.2.stage} (Section 5.3.3). This is done in lines 21-31 below. The results from the \(N_{\text{boot}}\) number of bootstrap simulations is stored in the matrix \texttt{boot.mat.2stage}.

Standard errors for the estimates are then calculated by using the following formula (lines 33-37):

\[
SE_{N_{\text{boot}}} (\hat{\beta}) = \sqrt{\frac{1}{N_{\text{boot}} - 1} \sum_{i=1}^{N_{\text{boot}}} (\hat{\beta}_i - \hat{\beta}^*)^2},
\]  

(5.11)

where \(\beta\) denotes the parameter and \(\hat{\beta}^*\) denotes the mean value of the bootstrap estimates (Wehrens et al., 2000).
5.4 Simulation Study

In this section, we discuss the simulation study to assess performance of the flexible two-stage method. In Section 5.4.1, the goals are illustrated. In Section 5.4.2, the R-code to perform the study is discussed. In Section 5.4.3, the results are given.

5.4.1 Goal

The following research questions were defined to assess performance of the method:

1. How does the flexible two-stage method perform compared to the two-stage method of Fiocco et al. (2009) on configurations where the distance between time points is equal?

2. Can the flexible two-stage method handle high counts?

3. How does the flexible two-stage method perform when the distances between time points are unequal compared to where they are equal?

Fiocco et al. performed large simulation study to test the non-flexible two-stage method. When the first question is answered with ‘they perform similarly’ (which is likely), we assume that the results of the simulation study also apply to the flexible two-stage method for configurations where the distance between time points is equal.

We want to address the second question, because high counts posed a problem with the method of Henderson and Shimakura (2003). When high counts occur in the data, their method fails to converge.

To answer the third question, we use configurations where the length of the interval increases with the time points. This is realistic, as in many applications, counts are high early in the study. Researchers will therefore choose for shorter intervals in the beginning of a study.

See Table 5.1 for an overview of the conducted simulation studies. For simplicity, we assumed that \( \mu = \mu_1 = \cdots = \mu_T \). \( T^1 \), \( T^2 \) and \( T^3 \) denote the vectors \((1, 2, 3, 4, 5, 6, 7), (0.25, 0.5, 1, 1.75, 2.75, 4, 5.5)\) and \((0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4)\).
respectively. Each simulation study consists of 1000 repetitions. Performance of the method is assessed by reporting the mean bias and RMSE. To make the simulation replicable, the seeds of the simulations are reported. See Section 4.4 for details on the mean bias, RMSE and the procedure on the seeds.

<table>
<thead>
<tr>
<th>Start.seed</th>
<th>ρ</th>
<th>ξ</th>
<th>µ</th>
<th>N</th>
<th>T</th>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.5</td>
<td>1</td>
<td>20</td>
<td></td>
<td>T^1</td>
<td>x</td>
</tr>
<tr>
<td>1000</td>
<td>0.25</td>
<td>4</td>
<td>10</td>
<td>20</td>
<td>T^1</td>
<td>x</td>
</tr>
<tr>
<td>2000</td>
<td>0.5</td>
<td>1</td>
<td>50</td>
<td>20</td>
<td>T^2</td>
<td>x</td>
</tr>
<tr>
<td>3000</td>
<td>0.5</td>
<td>1</td>
<td>10</td>
<td>20</td>
<td>T^2</td>
<td>x</td>
</tr>
<tr>
<td>4000</td>
<td>0.5</td>
<td>1</td>
<td>10</td>
<td>20</td>
<td>T^3</td>
<td>x</td>
</tr>
</tbody>
</table>

Table 5.1: Table with simulation studies. Column 1 gives the seed. Columns 2-4 give the true parameter values. The last 3 columns indicate which simulation study is used to assess which research question.

5.4.2 R-code

The R-files written to perform the simulation study are included in Appendix B.1 and Appendix B.2. The latter contains a function which performs a single simulation study, and returns the results. The former calls to this function for each of the simulation studies. This code is not discussed in this thesis, as there is no novelty to doing simulation studies.

The R-code written to implement the non-flexible two-stage procedure is included in Appendix A.5.

5.4.3 Results

The results of the simulation study are shown in tables 5.2 through 5.4. The order of the tables corresponds to the order of the research questions stated in Section 5.4.1.

The simulation studies with µ = 10 took approximately 5 seconds per run, the simulation study with µ = 50 approximately 1 minute. Also, both the flexible and non-flexible two-stage method converged for each run in each simulation study.
### Chapter 5. Flexible Two-stage

<table>
<thead>
<tr>
<th>$\rho$</th>
<th>$\zeta$</th>
<th>$\mu$</th>
<th>$N$</th>
<th>$T$</th>
<th>bias</th>
<th>RMSE</th>
<th>bias</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>1</td>
<td>10</td>
<td>20</td>
<td></td>
<td>$\hat{\rho}$</td>
<td>-0.016</td>
<td>0.101</td>
<td>-0.016</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\zeta}$</td>
<td>-0.056</td>
<td>0.169</td>
<td>-0.056</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_1$</td>
<td>0.035</td>
<td>2.436</td>
<td>0.035</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_2$</td>
<td>-0.036</td>
<td>2.307</td>
<td>-0.036</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_3$</td>
<td>-0.015</td>
<td>2.321</td>
<td>-0.015</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_4$</td>
<td>0.073</td>
<td>2.343</td>
<td>0.073</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_5$</td>
<td>0.048</td>
<td>2.483</td>
<td>0.048</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_6$</td>
<td>0.099</td>
<td>2.318</td>
<td>0.099</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_7$</td>
<td>0.147</td>
<td>2.293</td>
<td>0.147</td>
</tr>
</tbody>
</table>

| 0.25  | 4     | 10    | 20  |     | $\hat{\rho}$ | -0.041 | 0.104 | -0.041 | 0.104 |
|       |       |       |     |     | $\hat{\zeta}$ | -0.216 | 0.624 | -0.216 | 0.624 |
|       |       |       |     |     | $\hat{\mu}_1$ | 0.183  | 4.549 | 0.183  | 4.549 |
|       |       |       |     |     | $\hat{\mu}_2$ | 0.289  | 4.335 | 0.289  | 4.335 |
|       |       |       |     |     | $\hat{\mu}_3$ | 0.069  | 4.587 | 0.069  | 4.587 |
|       |       |       |     |     | $\hat{\mu}_4$ | 0.009  | 4.424 | 0.009  | 4.424 |
|       |       |       |     |     | $\hat{\mu}_5$ | -0.089 | 4.444 | -0.089 | 4.444 |
|       |       |       |     |     | $\hat{\mu}_6$ | -0.112 | 4.440 | -0.112 | 4.440 |
|       |       |       |     |     | $\hat{\mu}_7$ | -0.156 | 4.384 | -0.156 | 4.384 |

**Table 5.2:** Results of the simulation study for the first research question: comparison of two-stage and flexible two-stage.

<table>
<thead>
<tr>
<th>$\rho$</th>
<th>$\zeta$</th>
<th>$\mu$</th>
<th>$N$</th>
<th>$T$</th>
<th>bias</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>1</td>
<td>50</td>
<td>20</td>
<td></td>
<td>$\hat{\rho}$</td>
<td>-0.021</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\zeta}$</td>
<td>-0.045</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_1$</td>
<td>-0.022</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_2$</td>
<td>0.162</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_3$</td>
<td>0.328</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_4$</td>
<td>0.119</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_5$</td>
<td>0.045</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_6$</td>
<td>-0.190</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\hat{\mu}_7$</td>
<td>-0.005</td>
</tr>
</tbody>
</table>

**Table 5.3:** The results of the simulation study for the second research question: high counts.
5.5 Conclusion

The results in Section 5.4.3 show that the flexible and non-flexible two-stage method perform similarly. Table 5.2 shows that the two methods give exactly the same results when the distance between time points is equal. (In individual runs, the two methods give the exact same estimators up to 6 digits.) Therefore, we assume that the results of the simulation study performed by Fiocco et al. for the non-flexible two-stage method also apply to the flexible two-stage method when distances between time points are equal. We therefore conclude that the flexible two-stage method is robust against misspecification of the multivariate frailty distribution and
that it performs similarly to the one-stage procedure of Henderson and Shimakura in terms of efficiency.

From Table 5.4, we can conclude that the length of the intervals does not influence performance of the flexible two-stage method; the reported mean bias and RMSE do not seem to be influenced.

Also, when we compare Table 5.3 to Table 4.6, we can conclude that the flexible method handles high counts similarly to the non-flexible method; the estimators are unbiased and the RMSE for the hazard rates is approximately 25% of the true parameter value for both methods.
Chapter 6

Application to Data Set

In this chapter, the methodology developed in Chapters 4 and 5 is applied to a real data set. In Section 6.1 the data set is introduced while in Section 6.2 the use of the Poisson model in the context of meta-analysis for survival curves is illustrated. In Section 6.3 the EM-algorithm, two-stage and flexible two-stage method are applied to the data set. In Section 6.4 the performance of the three methods is compared.

6.1 Introduction Data Set

The data was collected by The European Organization for Research and Treatment of Cancer (EORTC trial 10854). The goal of the research was to study the effect of one course of perioperative chemotherapy given directly after surgery, compared to surgery alone for women with stage I or II invasive breast cancer (van der Hage et al., 2001). The study was a phase III trial, which means the treatments involved have been tested before on human subjects. The goal of a phase III trial is to test a method on a large group (1000-3000 subjects) to confirm safety and effectiveness. After a successful phase III trial, the method is approved for general use.

6.1.1 Breast Cancer

Usually, breast cancer begins in the cells of the lobules or the ducts. The lobules are the cells which produce the milk and the ducts are the passages that drain milk from the lobules to the nipple. When the cancer is non-invasive, it means that the cancer has not spread yet to the surrounding tissue. If it is invasive, it did. (Breastcancer.org)

The severity of the breast cancer can be described with the TNM (Tumor, Node, Metastasis) system, where 0 describes the least severe breast cancer (non-invasive) and IV describes the most severe one. The stage is determined (as the name suggests) by looking at the size of the tumor, whether or not the cancer has spread to lymph nodes in the armpits, and whether the tumor has metastasized. A tumor is metastasized when it has spread to parts of the body other than the breast itself or the nearby lymph nodes. (Breastcancer.org)

Normally, the tumor(s) are removed from the body by using surgery, which can be followed by chemotherapy, radiotherapy, or both. The surgery can
Chapter 6. Application to Data Set

<table>
<thead>
<tr>
<th>Country</th>
<th>Institution</th>
</tr>
</thead>
<tbody>
<tr>
<td>France</td>
<td>Centre Henri Becquerel, Rouen</td>
</tr>
<tr>
<td></td>
<td>Centre René Huguenin, St Cloud</td>
</tr>
<tr>
<td></td>
<td>Centre Regional François Baclesse</td>
</tr>
<tr>
<td></td>
<td>Centre Oscar Lambret, Lille</td>
</tr>
<tr>
<td>Netherlands</td>
<td>Leiden University Medical Center, Leiden</td>
</tr>
<tr>
<td></td>
<td>Maria Hospital Tilburg</td>
</tr>
<tr>
<td></td>
<td>Netherlands Cancer Institute, Amsterdam</td>
</tr>
<tr>
<td></td>
<td>Academic Medical Center, Amsterdam</td>
</tr>
<tr>
<td>Republic of South Africa</td>
<td>University of Stellenbosch, Tygerberg Hospital, Tygerberg</td>
</tr>
<tr>
<td>Poland</td>
<td>Medical University of Gdansk, Gdansk</td>
</tr>
<tr>
<td></td>
<td>Institute of Oncology, Medical Academy of Lodz</td>
</tr>
<tr>
<td>Spain</td>
<td>Hospital Virgen de las Nieves, Granada</td>
</tr>
<tr>
<td>Italy</td>
<td>Policlinico A. Gemelli- Universita del sacro Cuore, Rome</td>
</tr>
<tr>
<td>Austria</td>
<td>Innsbruck universitaetsklinik, Innsbruck</td>
</tr>
<tr>
<td>Russia</td>
<td>Petrov Research Institute of Oncology, St Petersburg</td>
</tr>
<tr>
<td>Greece</td>
<td>Hellenic Cancer Institute- Saint Savas Hospital, Athens</td>
</tr>
</tbody>
</table>

**Table 6.1: Hospitals participating in OERTC trial 10854**

be classified as breast conserving (partial breast removal) or mastectomy (complete breast removal). Perioperative chemotherapy stands for chemotherapy given around the time of surgery. The only difference between perioperative and normal chemotherapy is thus the moment at which the chemotherapy is started.

### 6.1.2 Data Characteristics

The data set contains 2795 women treated for invasive stage I or II breast cancer. These patients were randomized for treatment in 16 centers. This means that whether a patient received perioperative chemotherapy or normal chemotherapy was randomized. Table 6.1 gives a list of the centers that participated in the study. They were made anonymous in the data set. The list of hospitals was retrieved from the original publication of the data (van der Hage et al. 2001). Most of the hospitals in this list are university hospitals. The hospitals are thus comparable in the type of patients they receive, but differences between countries can be expected. Patients entered at the time they had surgery and left the study when they died or when they were lost to follow-up.

In table 6.2 characteristics of the centers are displayed. A number of patients had to be excluded from analysis. These were removed before constructing the table. In Section 6.1.3 it is explained why these patients are excluded.
### Table 6.2: Number of patients for each center in OERTC trial 10854 (2336 patients)

<table>
<thead>
<tr>
<th>Hospital Number</th>
<th>Perioperative Chemotherapy patients</th>
<th>Control patients</th>
</tr>
</thead>
<tbody>
<tr>
<td>206</td>
<td>149</td>
<td>155</td>
</tr>
<tr>
<td>208</td>
<td>298</td>
<td>301</td>
</tr>
<tr>
<td>234</td>
<td>91</td>
<td>91</td>
</tr>
<tr>
<td>252</td>
<td>447</td>
<td>438</td>
</tr>
<tr>
<td>310</td>
<td>82</td>
<td>83</td>
</tr>
<tr>
<td>890</td>
<td>100</td>
<td>101</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>1167</strong></td>
<td><strong>1169</strong></td>
</tr>
</tbody>
</table>

6.1.3 Exclusion of Patients from Analysis

A number of patients were excluded from analysis. Among these were 40 patients who received preoperative chemotherapy or were not eligible for study due to false inclusion or severe protocol violation. Another 42 patients did not meet the requirement of having stage I or stage II breast cancer. In addition, 2 patients were deleted because they did not have full information on the variables needed for the analysis. Lastly, hospitals with fewer than 100 patients participating were removed (375 patients).

6.1.4 Analysis

Figure 6.1 displays the estimated survival curve for each hospital and each treatment. Differences between the survival curves might be due to chance, therefore we want to conduct a statistical analysis to study impact of perioperative chemotherapy on overall survival and to investigate hospital heterogeneity.

We have the full data set at our disposal. In meta-analyses, this is often not the case and only the survival curves are available as in Figure 6.1. To illustrate how the methods developed in this thesis can be used in such studies, we will act as if we do not have the full data set and need to base our analysis on the reported survival curves of the 6 hospitals.

In meta-analyses, there is often no information available on the covariates. Therefore, we will not include these in our analysis. See van der Hage et al. (2001) for an analysis of this data set in which covariates are included.

6.2 Data Reconstruction

In this section, we describe the use of the Poisson model in the context of meta-analysis for survival curves. First, we describe how to derive the number at risk and number of events starting from the estimated survival
Chapter 6. Application to Data Set

Figure 6.1: Kaplan Meier survival curves for the 6 hospitals in the data set with more than 100 patients split over the two patient groups.

function. See for example Parmar et al. (1998) for details about data reconstruction. The longitudinal count data frame and notation used in this Section are introduced in Section 2.4.3.

We proceed as follows: let 0 denote the begin of the study and $W$ the endpoint. The interval $[0, W]$ is split into $T$ intervals. Let $t_1, \ldots, t_T$ denote the end points of these intervals and let $N$ denote the number of hospitals included in the study. We will distinguish between time $t$, which refers to the follow-up time, and time $j$, which indicates the index of the time intervals. By assuming that the patients are censored at a constant rate within the intervals, we reconstruct the number of patients at risk $r_{ij}$, the number of deaths $d_{ij}$ and the number of censored patients $c_{ij}$ for each study $i$ and time interval $j$. The number of patients at risk varies throughout the interval. A decision therefore has to be made on how to define it and we use the following definition:

$$r_{ij} = \#(\text{at risk at beginning of interval}) - \frac{c_{ij} + d_{ij}}{2}. \quad (6.1)$$

We assume that the hazard rate for individuals (see Section 2.1.2) is piecewise constant. It is assumed to be the same for all hospitals, but varies over time: $\lambda_j$ for $j = (1, \ldots, T)$. Hospital heterogeneity is included by introducing a frailty term $Z_{ij}$. This frailty process has first order autoregressive correlation structure $\text{corr}(Z_{il}, Z_{ik}) = \rho^{|k-l|}$ and marginal distribution $\Gamma(\theta, \theta)$. The frailty variance is constant and is denoted by $\xi (= \theta^{-1})$. The hazard rate is modeled as follows:

$$D_{ij} | Z_{ij} \sim \text{Po}(Z_{ij} \lambda_j \Delta_j r_{ij}) \quad (6.2)$$

where $\Delta_j$ denotes the length of interval $j$. $\Delta_j$ and $r_{ij}$ are fixed and therefore can be treated as offset in the analysis. Since they are constant, they can be combined into one component: $\tilde{r}_{ij} = \Delta_j r_{ij}$, often called the person-years. Using this notation, (6.2) can be written as:
\[ D_{ij} | Z_{ij} \overset{d}{=} P_{\text{pois}}(Z_{ij} \lambda_j \tilde{r}_{ij}). \quad (6.3) \]

When \( \tilde{r}_{ij} \) is treated as offset, the EM-algorithm of Chapter 4, the two-stage method of Fiocco et al., and the flexible two-stage method of Chapter 5 can be used to estimate \((\lambda, \theta, \rho)\).

The R-code of Fiocco et al. to implement the two-stage method (Appendix A.5) allows offset. However, the R-code discussed in Sections 4.3 and 5.3 to implement the EM-algorithm and flexible two-stage method do not allow an offset. R-code therefore needed to be rewritten. These adjustments are trivial, and therefore they are not discussed in this thesis.

The data was provided in a SPSS-file. See Appendix C for the R-code used to transform the data set to a longitudinal count data frame.

Standard errors for the parameters \((\mu, \rho, \theta)\) are estimated by using a parametric bootstrap approach (see Sections 4.3.6 and 5.3.4). Confidence intervals are approximated with basic bootstrap confidence intervals. Let \( \beta \) denote the parameter of interest, \( N_{\text{boot}} \) the number of bootstrap simulations and \((\hat{\beta}_1^*, \ldots, \hat{\beta}_{N_{\text{boot}}}^*)\) the ordered bootstrap sample for \( \beta \). Then

\[ \left[ \hat{\beta}_\left\lfloor (N_{\text{boot}}+1)(\alpha/2) \right\rfloor, \hat{\beta}_\left\lceil (N_{\text{boot}}+1)(1-\alpha/2) \right\rceil \right] \quad (6.4) \]

is a confidence interval with approximate \((1-\alpha)100\%\) coverage probability (Wehrens et al., 2000).

To compare the two treatments, the data set is divided in two. Let treatment group \( A \) denote the control group, and \( B \) the group who receives perioperative chemotherapy. The parameters, standard errors and confidence intervals are computed separately for the two groups. To compare the effect of the perioperative chemotherapy, the hazard ratio:

\[ \hat{\text{HR}}_j = \frac{\hat{\lambda}_j^B}{\hat{\lambda}_j^A} \quad (6.5) \]

is computed for \( j = (1, \ldots, T) \). The variance of the hazard ratio is computed by making use of the following equality:

\[ \log(\hat{\text{HR}}_j) = \log(\hat{\lambda}_j^B) - \log(\hat{\lambda}_j^A). \quad (6.6) \]

\( \hat{\lambda}_j^A \) and \( \hat{\lambda}_j^B \) are independent, because the two arms of treatment are independent. Therefore, the variance of the log of the hazard ratio is:

\[ \text{Var}(\log(\hat{\text{HR}}_j)) = \text{Var}(\log(\hat{\lambda}_j^B)) + \text{Var}(\log(\hat{\lambda}_j^A)). \quad (6.7) \]

An estimate for \( \text{Var}(\log(\hat{\text{HR}}_j)) \) can be found by plugging in bootstrap estimates of \( \text{Var}(\log(\hat{\lambda}_j^B)) \) and \( \text{Var}(\log(\hat{\lambda}_j^A)) \) into (6.7). An approximate \((1-\alpha)100\%\) confidence interval for \( \log(\hat{\text{HR}}_j) \) is then:
\[
\left[ \log(\hat{HR}_j) - z_{\alpha/2} \times \text{SE}(\log(\hat{HR}_j)), \log(\hat{HR}_j) + z_{\alpha/2} \times \text{SE}(\log(\hat{HR}_j)) \right]
\] (6.8)

where \(z_{\alpha/2}\) denotes the \(\alpha/2\) percentile of the standard normal distribution. A confidence interval for \(\hat{HR}_j\) can be found by taking the exponent of (6.8).

Finally, we obtain an estimate of the overall survival functions for both patient groups by using the estimated \(\lambda_A, \lambda_B\) as parameters of the piecewise exponential function. The survival function is defined as:

\[
\hat{S}(t) = \hat{S}_{j-1} \exp(-\hat{\lambda}_j (t - t_{j-1}))
\] (6.9)

for \(t_{j-1} < t \leq t_j\), where \(\hat{S}_j\) is defined recursively as:

\[
\hat{S}_j = \hat{S}_{j-1} \exp(-\hat{\lambda}_j (t_j - t_{j-1}))
\] (6.10)

for \(j = (1, \ldots, T)\), and \(\hat{S}_0 = 1\).

### 6.3 Model Estimation

In this section, the estimation of the model by using the different methods developed in this thesis is illustrated. We follow the procedure described in Section \[6.2\].

The following endpoints were chosen for the time intervals for the EM-algorithm, two-stage method and flexible two-stage method respectively: \((2, W)\), \((3, 6, 9, 12, W)\) and \((1, 3, 5, 10, W)\), where \(W = 4.19\) is the maximum follow-up time of patients in the data set.

In Tables \[6.3, 6.4, 6.6, 6.7, 6.9\] and \[6.10\] for each group and method separately, the number of patients at risk and number of person-years for each interval and hospital are displayed. Each element \(a/b\) in the table refers to number of events \((a)\) and the number of person-years \((b)\). They have been rounded to the nearest integer.

In Tables \[6.5, 6.8\] and \[6.11\] estimated parameters are shown for each of the three methods. The estimated correlation, frailty variance and the hazard rate for each time point along with their corresponding 95% confidence interval are given. In addition, the hazard ratio for each time point is provided.

In Figures \[6.2, 6.5\] and \[6.8\] survival curves for each center along with the estimated survival based on the method three methods respectively and the estimated survival curve based on the Kaplan Meier method are shown. In Figures \[6.3, 6.6\] and \[6.9\] the estimated survival curves based on the three methods respectively are shown. Figures \[6.4, 6.7\] and \[6.10\] show the hazard ratio for each time point and their corresponding 95% confidence interval.
### 6.3.1 EM

**Hospital Interval**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[0-2]</td>
<td>[2-W]</td>
</tr>
<tr>
<td>206</td>
<td>4/604</td>
<td>31/884</td>
</tr>
<tr>
<td>208</td>
<td>7/1191</td>
<td>80/1773</td>
</tr>
<tr>
<td>234</td>
<td>1/363</td>
<td>20/548</td>
</tr>
<tr>
<td>252</td>
<td>12/1758</td>
<td>78/2651</td>
</tr>
<tr>
<td>310</td>
<td>13/317</td>
<td>19/420</td>
</tr>
<tr>
<td>890</td>
<td>10/392</td>
<td>32/548</td>
</tr>
</tbody>
</table>

**TABLE 6.3:** Reconstructed data for EM-algorithm perioperative-group.

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[0-2]</td>
<td>[2-W]</td>
</tr>
<tr>
<td>206</td>
<td>6/602</td>
<td>45/908</td>
</tr>
<tr>
<td>208</td>
<td>8/1190</td>
<td>76/1785</td>
</tr>
<tr>
<td>234</td>
<td>3/361</td>
<td>16/536</td>
</tr>
<tr>
<td>252</td>
<td>12/1757</td>
<td>80/2590</td>
</tr>
<tr>
<td>310</td>
<td>7/323</td>
<td>36/463</td>
</tr>
<tr>
<td>890</td>
<td>6/396</td>
<td>41/579</td>
</tr>
</tbody>
</table>

**TABLE 6.4:** Reconstructed data for EM-algorithm control-group.

<table>
<thead>
<tr>
<th>Estimation</th>
<th>Perioperative</th>
<th>Control</th>
<th>Hazard Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho$</td>
<td>0.450</td>
<td>0.384</td>
<td>[0.001, 1.000]</td>
</tr>
<tr>
<td>$\xi$</td>
<td>0.166</td>
<td>0.085</td>
<td>[0.010, 0.338]</td>
</tr>
<tr>
<td>$\mu_1$</td>
<td>0.013</td>
<td>0.003</td>
<td>[0.007, 0.018]</td>
</tr>
<tr>
<td>$\mu_2$</td>
<td>0.042</td>
<td>0.007</td>
<td>[0.029, 0.058]</td>
</tr>
</tbody>
</table>

**TABLE 6.5:** Model estimation based on the EM-algorithm.
Chapter 6. Application to Data Set

Figure 6.2: Survival curves for each single center (gray line) along with the overall estimated survival based on the EM-algorithm (black line) and the estimate Kaplan Meier curve based on the whole data set (light blue line).

Figure 6.3: Estimated overall survival based on EM-algorithm for perioperative-group (black, solid) and for control-group (black, dashed) and the estimate Kaplan Meier curve based on the whole data set for perioperative-group (light blue, solid) and control group (light blue, dashed).
6.3.2 Two-stage

**TABLE 6.6:** Reconstructed data for two-stage method perioperative-group.

<table>
<thead>
<tr>
<th>Hospital</th>
<th>Interval</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>206</td>
<td>[0-3]</td>
<td>7/902</td>
<td>7/416</td>
<td>15/380</td>
<td>6/208</td>
<td>0/23</td>
</tr>
<tr>
<td>208</td>
<td>[3-6]</td>
<td>17/1772</td>
<td>33/794</td>
<td>21/705</td>
<td>16/448</td>
<td>0/84</td>
</tr>
<tr>
<td>252</td>
<td>[9-12]</td>
<td>21/2622</td>
<td>31/1222</td>
<td>23/1078</td>
<td>14/543</td>
<td>1/36</td>
</tr>
<tr>
<td>890</td>
<td></td>
<td>20/537</td>
<td>14/218</td>
<td>6/168</td>
<td>2/82</td>
<td>0/9</td>
</tr>
</tbody>
</table>

**TABLE 6.7:** Reconstructed data for two-stage method control-group.

<table>
<thead>
<tr>
<th>Hospital</th>
<th>Interval</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>206</td>
<td>[0-3]</td>
<td>12/894</td>
<td>16/405</td>
<td>12/356</td>
<td>9/190</td>
<td>2/19</td>
</tr>
<tr>
<td>234</td>
<td>[6-9]</td>
<td>6/537</td>
<td>8/243</td>
<td>4/225</td>
<td>1/126</td>
<td>0/12</td>
</tr>
<tr>
<td>890</td>
<td></td>
<td>13/584</td>
<td>22/226</td>
<td>10/159</td>
<td>2/76</td>
<td>0/9</td>
</tr>
</tbody>
</table>

**FIGURE 6.4:** Estimation of hazard ratio’s based on the EM-algorithm.
Chapter 6. Application to Data Set

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>Conf. int.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perioperative</td>
<td>ρ</td>
<td>0.512</td>
<td>0.321</td>
</tr>
<tr>
<td></td>
<td>ζ</td>
<td>0.123</td>
<td>0.061</td>
</tr>
<tr>
<td></td>
<td>μ₁</td>
<td>0.016</td>
<td>0.003</td>
</tr>
<tr>
<td></td>
<td>μ₂</td>
<td>0.035</td>
<td>0.006</td>
</tr>
<tr>
<td></td>
<td>μ₃</td>
<td>0.026</td>
<td>0.005</td>
</tr>
<tr>
<td></td>
<td>μ₄</td>
<td>0.029</td>
<td>0.006</td>
</tr>
<tr>
<td></td>
<td>μ₅</td>
<td>0.022</td>
<td>0.011</td>
</tr>
<tr>
<td>Control</td>
<td>ρ</td>
<td>0.999</td>
<td>0.139</td>
</tr>
<tr>
<td></td>
<td>ζ</td>
<td>0.144</td>
<td>0.091</td>
</tr>
<tr>
<td></td>
<td>μ₁</td>
<td>0.015</td>
<td>0.003</td>
</tr>
<tr>
<td></td>
<td>μ₂</td>
<td>0.044</td>
<td>0.008</td>
</tr>
<tr>
<td></td>
<td>μ₃</td>
<td>0.039</td>
<td>0.007</td>
</tr>
<tr>
<td></td>
<td>μ₄</td>
<td>0.029</td>
<td>0.007</td>
</tr>
<tr>
<td></td>
<td>μ₅</td>
<td>0.043</td>
<td>0.018</td>
</tr>
<tr>
<td>Hazard Ratio</td>
<td>HR₁</td>
<td>1.050</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HR₂</td>
<td>0.799</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HR₃</td>
<td>0.674</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HR₄</td>
<td>0.999</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HR₅</td>
<td>0.508</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.8: Model estimation based on the two-stage method.

![Perioperative survival curves](image1)

![Control survival curves](image2)

Figure 6.5: Survival curves for each single center (gray line) along with the overall estimated survival based on the two-stage method (black line) and the estimate Kaplan Meier curve based on the whole data set (light blue line).
Figure 6.6: Estimated overall survival based on two-stage method for perioperative-group (black, solid) and for control-group (black, dashed) and the estimate Kaplan Meier curve based on the whole data set for perioperative-group (light blue, solid) and control group (light blue, dashed).
Figure 6.7: Estimation of hazard ratio’s based on the two-stage method.

6.3.3 Flexible Two-stage

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>208</td>
<td>1/598</td>
<td>16/578</td>
<td>23/539</td>
<td>43/1122</td>
<td>4/400</td>
<td></td>
</tr>
<tr>
<td>234</td>
<td>0/182</td>
<td>6/176</td>
<td>6/164</td>
<td>8/362</td>
<td>1/138</td>
<td></td>
</tr>
<tr>
<td>252</td>
<td>3/884</td>
<td>18/869</td>
<td>21/829</td>
<td>43/1595</td>
<td>5/490</td>
<td></td>
</tr>
<tr>
<td>310</td>
<td>2/164</td>
<td>14/145</td>
<td>5/125</td>
<td>6/272</td>
<td>5/103</td>
<td></td>
</tr>
<tr>
<td>890</td>
<td>3/200</td>
<td>17/177</td>
<td>9/151</td>
<td>11/250</td>
<td>2/61</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.9: Reconstructed data for flexible two-stage method perioperative-group.
### Table 6.10: Reconstructed data for flexible two-stage method control-group.

<table>
<thead>
<tr>
<th>Hospital</th>
<th>Interval</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(0-1]</td>
<td>[1-3]</td>
<td>[3-5]</td>
<td>[5-10]</td>
<td>[10-W]</td>
<td></td>
</tr>
<tr>
<td>206</td>
<td>1/894</td>
<td>11/405</td>
<td>13/356</td>
<td>19/190</td>
<td>7/19</td>
<td></td>
</tr>
<tr>
<td>208</td>
<td>0/1776</td>
<td>14/806</td>
<td>24/698</td>
<td>36/436</td>
<td>10/83</td>
<td></td>
</tr>
<tr>
<td>234</td>
<td>0/537</td>
<td>6/243</td>
<td>7/225</td>
<td>6/126</td>
<td>0/12</td>
<td></td>
</tr>
<tr>
<td>890</td>
<td>2/584</td>
<td>11/226</td>
<td>17/159</td>
<td>17/76</td>
<td>0/9</td>
<td></td>
</tr>
</tbody>
</table>

### Table 6.11: Model estimation based on the flexible two-stage method.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>Conf. int.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perioperative</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \rho )</td>
<td>0.945</td>
<td>0.246</td>
<td>[0.000, 1.000]</td>
</tr>
<tr>
<td>( \zeta )</td>
<td>0.136</td>
<td>0.076</td>
<td>[0.000, 0.286]</td>
</tr>
<tr>
<td>( \mu_1 )</td>
<td>0.005</td>
<td>0.002</td>
<td>[0.003, 0.010]</td>
</tr>
<tr>
<td>( \mu_2 )</td>
<td>0.043</td>
<td>0.008</td>
<td>[0.029, 0.061]</td>
</tr>
<tr>
<td>( \mu_3 )</td>
<td>0.035</td>
<td>0.007</td>
<td>[0.022, 0.051]</td>
</tr>
<tr>
<td>( \mu_4 )</td>
<td>0.031</td>
<td>0.006</td>
<td>[0.021, 0.043]</td>
</tr>
<tr>
<td>( \mu_5 )</td>
<td>0.016</td>
<td>0.004</td>
<td>[0.008, 0.026]</td>
</tr>
<tr>
<td>Control</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \rho )</td>
<td>0.450</td>
<td>0.169</td>
<td>[0.000, 1.000]</td>
</tr>
<tr>
<td>( \zeta )</td>
<td>0.166</td>
<td>0.105</td>
<td>[0.000, 0.389]</td>
</tr>
<tr>
<td>( \mu_1 )</td>
<td>0.003</td>
<td>0.001</td>
<td>[0.001, 0.006]</td>
</tr>
<tr>
<td>( \mu_2 )</td>
<td>0.042</td>
<td>0.009</td>
<td>[0.026, 0.062]</td>
</tr>
<tr>
<td>( \mu_3 )</td>
<td>0.047</td>
<td>0.010</td>
<td>[0.029, 0.068]</td>
</tr>
<tr>
<td>( \mu_4 )</td>
<td>0.041</td>
<td>0.008</td>
<td>[0.026, 0.057]</td>
</tr>
<tr>
<td>( \mu_5 )</td>
<td>0.019</td>
<td>0.006</td>
<td>[0.009, 0.031]</td>
</tr>
<tr>
<td>Hazard Ratio</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( HR_1 )</td>
<td>1.723</td>
<td></td>
<td>[0.111, 26.560]</td>
</tr>
<tr>
<td>( HR_2 )</td>
<td>1.002</td>
<td></td>
<td>[0.446, 2.225]</td>
</tr>
<tr>
<td>( HR_3 )</td>
<td>0.747</td>
<td></td>
<td>[0.318, 1.744]</td>
</tr>
<tr>
<td>( HR_4 )</td>
<td>0.767</td>
<td></td>
<td>[0.358, 1.644]</td>
</tr>
<tr>
<td>( HR_5 )</td>
<td>0.822</td>
<td></td>
<td>[0.249, 2.708]</td>
</tr>
</tbody>
</table>

*Excerpt from Chapter 6. Application to Data Set*
**Chapter 6. Application to Data Set**

**Figure 6.8:** Survival curves for each single center (gray line) along with the overall estimated survival based on the flexible two-stage method (black line) and the estimate Kaplan Meier curve based on the whole data set (light blue line).

**Figure 6.9:** Estimated overall survival based on flexible two-stage for perioperative-group (black, solid) and for control-group (black, dashed) and the estimate Kaplan Meier curve based on the whole data set for perioperative-group (light blue, solid) and control group (light blue, dashed).
6.4 Conclusion

The three methods seem to estimate the overall survival curve in a similar way (Figures 6.2, 6.5 and 6.8); the estimated curves are similar to the estimated survival curve based on the full data set. This suggests that the methods proposed in this thesis can be applied in meta-analysis for the survival curve when the individual patient data is not available and only aggregate data can be used. The EM-algorithm has the disadvantage that it can only be applied on two time intervals, and therefore the use of this method is unfortunately rather limited.

There seems to be no significant effect of perioperative chemotherapy on overall survival. The hazard ratio's do not differ significantly from 1 (Figures 6.4, 6.7, and 6.10). The same conclusion was drawn by van der Hage et al. (2001).

The analysis performed suggests that there is no significant hospital heterogeneity; the estimated frailty variance $\xi$ is close to zero for each method and treatment type. This suggests that the frailty model does not need to be applied on this data set. As can be observed in Tables 6.5, 6.8 and 6.11, the correlation parameter $\rho$ is estimated poorly. We suspect that it might be due to the low frailty variance and a low ratio of counts versus person years.
Chapter 7

Discussion

In this chapter, a short critical discussion about the issues discussed in this thesis is given. In Section 1.3, three research questions were addressed:

1. Is it possible to apply a full-likelihood estimation method to estimate the parameters of the model described in Section 2.4.3?

2. How does the two-stage method perform compared to a full likelihood method?

3. Can we improve the two-stage method?

In Sections 7.1, 7.2 and 7.3, answers to these questions are discussed. In Section 7.4, possibilities for further research are outlined.

7.1 Full Likelihood

In Chapter 3, we proposed two new methods to construct a multivariate gamma process. The idea was to make the application of full-likelihood procedures easier by simplifying the underlying gamma frailty process. Models 1 and 2 use the infinite divisibility property of the gamma distribution. Model 3 uses a combination of the latter property and a particular relationship between the gamma and beta distribution. We concluded that Model 2 (Section 3.3) is more general than Model 1 (Section 3.2). However, the number of frailty terms is still quadratic as a function of the number of time points $T$. Model 3 (Section 3.4) needs a linear amount of random variables as a function of the number of time points, some components have a beta-distribution.

In Chapter 4, an attempt was made to apply a combination of the EM-algorithm and the profile likelihood method (both full-likelihood methods). An attempt was made to use Model 3 in this context (undocumented), but we did not succeed in finding the algebraic expressions needed to execute the algorithm.

From Chapter 4, we concluded that it is possible to apply the EM-algorithm for two time points ($T = 2$). However, it is probably not feasible for larger number of time points.

Other full-likelihood methods were explored (undocumented), amongst which were the Monte Carlo Markov Chain (MCMC) method and the hierarchical likelihood method. (The latter is not a full-likelihood method officially.) We
did not investigate these options extensively, but enough to conclude that it will not be straightforward. Computation time of the algorithms will be the biggest obstacle. Applying and testing these methods requires a lot of time, and was not feasible in the time frame allocated to this thesis. Further research will be needed whether these full-likelihood methods can be applied.

7.2 Comparison

In Chapter 4, the two-stage method of Fiocco et al. was compared to the EM-algorithm for two time points ($T = 2$). A large simulation study was carried out to compare performance of both methods. We concluded that the methods performed similarly in terms of mean bias and Root Mean Square Error (RMSE).

We explored the possibility of comparing the two-stage method for $T > 2$ to the continuous time EM-algorithm of Putter and van Houwelingen (undocumented). A discrete time method applied to many intervals should give approximately the same results as a continuous time method. The latter has been implemented in R by the authors, but it is only computationally feasible for small counts.

7.3 Two-Stage Method

In Chapter 5, we extended the two-stage method to situations where the distance between time points is not constant. This was achieved by basing the two-stage method on Model 2 instead of Model 1. Results from the simulation study showed that the flexible two-stage method has the same performance as the non-flexible two-stage on configurations where the distance between time points is equal. This is logical, as model 2 reduces to model 1 when the distance between the time points is equal. In addition, the flexible two-stage method performs similarly on configurations where the distance between time points is unequal compared to where they are equal.

7.4 Further Research

From Chapter 4, we concluded that the combination of the EM-algorithm and the profile likelihood on Model 2 is computationally unfeasible for $T > 2$. The number of terms to sum to evaluate the E-step is too large. It might be feasible to use it for three time points and small counts, but we suspect that performing simulation studies will take too long, even on a super computer. R-code could be written to implement the algorithm for $T > 2$ to test its performance.

It might be of interest to investigate whether it is possible to analytically determine the variance of the estimators of the EM-algorithm of Chapter 4.
for $T > 2$. This might make it possible to compute the variance without executing the EM-algorithm. This variance could be compared to the variance of the estimators for the two-stage method of Chapter 5 to study the efficiency of the two methods.

The EM-algorithm of Chapter 4 could be simplified by making use of an MCMC method to execute the E-step (Sherman et al., 1999). As a consequence it will not be necessary to combine the EM-algorithm and the profile likelihood method: only the EM-algorithm would need to be applied. When a MCMC method is used, these expected values can be computed based on the MCMC samples.

However, applying MCMC would involve sampling many gamma random variables, which might make the algorithm too slow. We explored the possibility of using Gibbs Sampling, but we refrained from it because this would involve integrating out a large number of gamma random variables to find the algebraic expressions necessary to execute the algorithm, leading to the same large number of summands as with the combination of profile likelihood and EM. Further research is needed to investigate whether it is feasible to apply MCMC methods in this context.

Another option is to use a hierarchical likelihood approach (Ha et al., 2009). With this method, the frailty and gamma variables are estimated jointly. We shortly examined this option. The number of gamma random variables to be estimated is rather big; it increases quadratically (Model 2) or linearly (Model 3) with $T$ and linearly with $N$. Applying this method would thus result in a high-dimensional maximization problem, but further research is needed.

For the discussed full-likelihood methods, we recommend to investigate whether it is possible to apply Model 3 instead of Model 2. As the number of gamma variables increases linearly with $T$ for Model 3, methods based on this model might be easier to generalize to higher time points than methods based on Model 2.

The flexible two-stage method of Chapter 5 can be easily be extended to incorporate covariates into the model. The only part that has to be changed in the R-code is line 21 in Section 5.3.3; the glm.nb function can estimate regression coefficients for Cox’s proportional hazards model (see Section 2.2). The same is true for the EM-algorithm of Chapter 4 (line 31 in the R-code of Section 4.3.3). In case these methods are used in the context of meta-analysis, decisions have to be made on how to include covariates on a study level. See for example Lambert et al. (2001) for a comparison of different methods.

Further research could be conducted on the robustness of the flexible two-stage method. Both misspecification of the frailty distribution (Gamma) and the count distribution (Poisson) could be investigated. Fiocco et al. only performed a simulation study to assess performance of the non-flexible two-stage method on log-normal distributed frailty terms.

To make the flexible two-stage method more accessible to practitioners. A R-package could be written. Also, guidelines on how to interpret observed values of the heterogeneity $\xi$ and the correlation $\rho$ could be given. There
are only few papers in which the method of Fiocco et al. or similar methods have been applied to real data sets. The flexible two-stage method of Chapter 5 could be applied to many different data sets to create a benchmark for these parameters.

It might also be possible to generalize the (flexible) two-stage method to other correlation structures than the auto-regressive correlation structure.

The multivariate gamma distribution is studied extensively in (Kotz et al., 2004). Furman (2007) recently published a paper where a new multivariate gamma distribution is introduced. Higher order moments and cumulants, Chebyshev-type inequalities and the multivariate probability density function were derived. In case these moments could be derived for the gamma processes described in Chapter 3 it might make a relevant contribution to this field.
Appendix A

R-code EM

A.1 Main R-file

```r
#########################################################################
#Main Code Simulation Study
#########################################################################

#Read in required R-files
source("TwoStage.R")
source("EmAlgorithm.R")
source("SimulationData.R")
source("SingleSimulation.R")

#setting parameters which are the same for all simulations
acc_mus <- 1e-05 #Accuracy parameter for EM-algorithm
acc_loglik <- 1e-05 #Accuracy parameter for EM-algorithm
T=2 #Number of time points
pyrs <- c(1,2) #Person years (time values corresponding to time points, necessary for two-stage procedure)

#Functions to calculate statistics of interest

#Function to calculate mean absolute bias
RMSE <- function(results,rho_t,th_t,mumat_t){
  # Input:
  # Results of a simulation (see SingleSimulation.R for details on this matrix)
  # True values of rho, xi, mu_1 and mu_2
  # Output:
  # matrix with mean absolute biases for rho, xi, mu_1 and mu_2 (rows)
  # and for two estimation procedures EM and two-stage (columns)
  RMSE <- matrix(NA,nrow=4,ncol=2)
  colnames(RMSE) <- c("EM","two-stage")
  rownames(RMSE) <- c("rho","xi","mu_1","mu_2")
  xi_t <- 1/th_t
  mu1_t <- mumat_t[1]
  mu2_t <- mumat_t[2]
  N <- nrow(results)
  RMSE["rho","EM"] <- sqrt(sum((results[,"rho_EM"]-rho_t)^2)/N)
  RMSE["xi","EM"] <- sqrt(sum((results[,"xi_EM"]-th_t)^2)/N)
  RMSE["mu_1","EM"] <- sqrt(sum((results[,"mu_1_EM"]-mu1_t)^2)/N)
  RMSE["mu_2","EM"] <- sqrt(sum((results[,"mu_2_EM"]-mu2_t)^2)/N)
  RMSE["rho","two-stage"] <- sqrt(sum((results[,"rho_two"]-rho_t)^2)/N)
  RMSE["xi","two-stage"] <- sqrt(sum((results[,"xi_two"]-th_t)^2)/N)
  RMSE["mu_1","two-stage"] <- sqrt(sum((results[,"mu_1_two"]-mu1_t)^2)/N)
  RMSE["mu_2","two-stage"] <- sqrt(sum((results[,"mu_2_two"]-mu2_t)^2)/N)
  return(RMSE)
}

#Function to calculate mean bias
MeanBias <- function(results,rho_t,th_t,mumat_t){
  # Input:
  # Results of a simulation (see SingleSimulation.R for details on this matrix)
  # True values of rho, xi, mu_1 and mu_2
  # Output:
  # vector with mean biases for rho, xi, mu_1 and mu_2
  # and for two estimation procedures EM and two-stage
  meanbias <- matrix(NA,nrow=4,ncol=2)
  colnames(meanbias) <- c("EM","two-stage")
  rownames(meanbias) <- c("rho","xi","mu_1","mu_2")
  xi_t <- 1/th_t
  mu1_t <- mumat_t[1]
  mu2_t <- mumat_t[2]
  N <- nrow(results)
  meanbias["rho","EM"] <- sum((results[,"rho_EM"]-rho_t))/N
  meanbias["xi","EM"] <- sum((results[,"xi_EM"]-th_t))/N
  meanbias["mu_1","EM"] <- sum((results[,"mu_1_EM"]-mu1_t))/N
  meanbias["mu_2","EM"] <- sum((results[,"mu_2_EM"]-mu2_t))/N
  meanbias["rho","two-stage"] <- sum((results[,"rho_two"]-rho_t))/N
  meanbias["xi","two-stage"] <- sum((results[,"xi_two"]-th_t))/N
  meanbias["mu_1","two-stage"] <- sum((results[,"mu_1_two"]-mu1_t))/N
  meanbias["mu_2","two-stage"] <- sum((results[,"mu_2_two"]-mu2_t))/N
  return(meanbias)
}

#Function to calculate standard deviation of bias
StdBias <- function(results,rho_t,th_t,mumat_t){
  # Input:
  # Results of a simulation (see SingleSimulation.R for details on this matrix)
  # True values of rho, xi, mu_1 and mu_2
  # Output:
  # vector with standard deviations of biases for rho, xi, mu_1 and mu_2
  # and for two estimation procedures EM and two-stage
  stdbias <- matrix(NA,nrow=4,ncol=2)
  colnames(stdbias) <- c("EM","two-stage")
  rownames(stdbias) <- c("rho","xi","mu_1","mu_2")
  xi_t <- 1/th_t
  mu1_t <- mumat_t[1]
  mu2_t <- mumat_t[2]
  N <- nrow(results)
  stdbias["rho","EM"] <- sqrt(sum((results[,"rho_EM"]-rho_t)^2)/N)
  stdbias["xi","EM"] <- sqrt(sum((results[,"xi_EM"]-th_t)^2)/N)
  stdbias["mu_1","EM"] <- sqrt(sum((results[,"mu_1_EM"]-mu1_t)^2)/N)
  stdbias["mu_2","EM"] <- sqrt(sum((results[,"mu_2_EM"]-mu2_t)^2)/N)
  stdbias["rho","two-stage"] <- sqrt(sum((results[,"rho_two"]-rho_t)^2)/N)
  stdbias["xi","two-stage"] <- sqrt(sum((results[,"xi_two"]-th_t)^2)/N)
  stdbias["mu_1","two-stage"] <- sqrt(sum((results[,"mu_1_two"]-mu1_t)^2)/N)
  stdbias["mu_2","two-stage"] <- sqrt(sum((results[,"mu_2_two"]-mu2_t)^2)/N)
  return(stdbias)
}

#Function to calculate standard deviation of bias
StdBias <- function(results,rho_t,th_t,mumat_t){
  # Input:
  # Results of a simulation (see SingleSimulation.R for details on this matrix)
  # True values of rho, xi, mu_1 and mu_2
  # Output:
  # vector with standard deviations of biases for rho, xi, mu_1 and mu_2
  # and for two estimation procedures EM and two-stage
  stdbias <- matrix(NA,nrow=4,ncol=2)
  colnames(stdbias) <- c("EM","two-stage")
  rownames(stdbias) <- c("rho","xi","mu_1","mu_2")
  xi_t <- 1/th_t
  mu1_t <- mumat_t[1]
  mu2_t <- mumat_t[2]
  N <- nrow(results)
  stdbias["rho","EM"] <- sqrt(sum((results[,"rho_EM"]-rho_t)^2)/N)
  stdbias["xi","EM"] <- sqrt(sum((results[,"xi_EM"]-th_t)^2)/N)
  stdbias["mu_1","EM"] <- sqrt(sum((results[,"mu_1_EM"]-mu1_t)^2)/N)
  stdbias["mu_2","EM"] <- sqrt(sum((results[,"mu_2_EM"]-mu2_t)^2)/N)
  stdbias["rho","two-stage"] <- sqrt(sum((results[,"rho_two"]-rho_t)^2)/N)
  stdbias["xi","two-stage"] <- sqrt(sum((results[,"xi_two"]-th_t)^2)/N)
  stdbias["mu_1","two-stage"] <- sqrt(sum((results[,"mu_1_two"]-mu1_t)^2)/N)
  stdbias["mu_2","two-stage"] <- sqrt(sum((results[,"mu_2_two"]-mu2_t)^2)/N)
  return(stdbias)
}
```

```r
```
# Results of a simulation (see SingleSimulation.R for details on this matrix)
# True values of rho, xi, mu_1 and mu_2
# Output:
# matrix with mean biases for rho, xi, mu_1, mu_2 (rows)
# and for two estimation procedures EM and two-stage (columns)

Biasses <- matrix(NA, nrow=4, ncol=2)
colnames(Biasses) <- c("EM", "two-stage")
rownames(Biasses) <- c("rho", "xi", "mu_1", "mu_2")

Biasses["rho","EM"] <- mean(results[,"rho_EM"] - rho_t)
Biasses["xi","EM"] <- mean(results[,"xi_EM"] - (1/th_t))
Biasses["mu_1","EM"] <- mean(results[,"mu_1_EM"] - mumat_t[1])
Biasses["mu_2","EM"] <- mean(results[,"mu_2_EM"] - mumat_t[2])
Biasses["rho","two-stage"] <- mean(results[,"rho_two"] - rho_t)
Biasses["xi","two-stage"] <- mean(results[,"xi_two"] - (1/th_t))
Biasses["mu_1","two-stage"] <- mean(results[,"mu_1_two"] - mumat_t[1])
Biasses["mu_2","two-stage"] <- mean(results[,"mu_2_two"] - mumat_t[2])

return(Biasses)
Appendix A. R-code EM

#Simulation study 4:
.seed.start <- 3000

#Number of replications
.nrep = 500  #Number of replications
.N = 20  #Number of studies
.rho_IV = 0.5; th_IV = 1  #Initial estimates for rho and xi
.rho_t = 0.5 ; th_t = 4 ; mumat_t = c(20,20)  #True model parameters

#Simulation study 4:
$results14 <- Simulation(nrep=nrep,N=N,T=T,rho_IV=rho_IV,th_IV=th_IV,rho_t=rho_t,th_t=th_t,mumat_t=mumat_t,pyrs,acc_mus,acc_loglik,seed.start)

#Simulation study 4:
$meanbias14 <- MeanBias(results14,rho_t,th_t,mumat_t)

#Simulation study 4:
$RMSE14 <- RMSE(results14,rho_t,th_t,mumat_t)

#Simulation study 5:
.seed.start <- 4000

#Number of replications
.nrep = 500  #Number of replications
.N = 20  #Number of studies
.rho_IV = 0.5; th_IV = 1  #Initial estimates for rho and xi
.rho_t = 0.5 ; th_t = 2 ; mumat_t = c(20,20)  #True model parameters

#Simulation study 5:
$results15 <- Simulation(nrep=nrep,N=N,T=T,rho_IV=rho_IV,th_IV=th_IV,rho_t=rho_t,th_t=th_t,mumat_t=mumat_t,pyrs,acc_mus,acc_loglik,seed.start)

#Simulation study 5:
$meanbias15 <- MeanBias(results15,rho_t,th_t,mumat_t)

#Simulation study 5:
$RMSE15 <- RMSE(results15,rho_t,th_t,mumat_t)

#Study 2

#Simulation study 1:
.seed.start <- 5000

#Number of replications
.nrep = 500  #Number of replications
.N = 20  #Number of studies
.rho_IV = 0.5; th_IV = 1  #Initial estimates for rho and xi
.rho_t = 0.5 ; th_t = 2 ; mumat_t = c(10,10)  #True model parameters

#Simulation study 1:
$results21 <- Simulation(nrep=nrep,N=N,T=T,rho_IV=rho_IV,th_IV=th_IV,rho_t=rho_t,th_t=th_t,mumat_t=mumat_t,pyrs,acc_mus,acc_loglik,seed.start)

#Simulation study 1:
$meanbias21 <- MeanBias(results21,rho_t,th_t,mumat_t)

#Simulation study 1:
$RMSE21 <- RMSE(results21,rho_t,th_t,mumat_t)

#Simulation study 2:
.seed.start <- 6000

#Number of replications
.nrep = 500  #Number of replications
.N = 20  #Number of studies
.rho_IV = 0.5; th_IV = 1  #Initial estimates for rho and xi
.rho_t = 0.5 ; th_t = 2 ; mumat_t = c(50,50)  #True model parameters

#Simulation study 2:
$results22 <- Simulation(nrep=nrep,N=N,T=T,rho_IV=rho_IV,th_IV=th_IV,rho_t=rho_t,th_t=th_t,mumat_t=mumat_t,pyrs,acc_mus,acc_loglik,seed.start)

#Simulation study 2:
$meanbias22 <- MeanBias(results22,rho_t,th_t,mumat_t)

#Simulation study 2:
$RMSE22 <- RMSE(results22,rho_t,th_t,mumat_t)

#Study 3
#Simulation study 1:
seed.start <- 7000
nrep = 500  #Number of replications
N = 10  #Number of studies
rho_IV = 0.5; th_IV = 1  #Initial estimates for rho and xi
rho_t = 0.5; th_t = 1; mumat_t = c(20,20)  #True model parameters
results31 <- Simulation(nrep=nrep,N=N,T=T,rho_IV=rho_IV,th_IV=th_IV,rho_t=rho_t,th_t=th_t,mumat_t=mumat_t,pyrs,acc_mus,acc_loglik,seed.start)
meanbias31 <- MeanBias(results31,rho_t,th_t,mumat_t)
RMSE31 <- RMSE(results31,rho_t,th_t,mumat_t)

#Simulation study 2:
seed.start <- 8000
nrep = 500  #Number of replications
N = 50  #Number of studies
rho_IV = 0.5; th_IV = 1  #Initial estimates for rho and xi
rho_t = 0.5; th_t = 1; mumat_t = c(20,20)  #True model parameters
results32 <- Simulation(nrep=nrep,N=N,T=T,rho_IV=rho_IV,th_IV=th_IV,rho_t=rho_t,th_t=th_t,mumat_t=mumat_t,pyrs,acc_mus,acc_loglik,seed.start)
meanbias32 <- MeanBias(results32,rho_t,th_t,mumat_t)
RMSE32 <- RMSE(results32,rho_t,th_t,mumat_t)

#Study 4

#Simulation study 1
seed.start <- 9000
nrep = 500  #Number of replications
N = 20  #Number of studies
rho_IV = 0.1; th_IV = 1  #Initial estimates for rho and xi
rho_t = 0.5; th_t = 1; mumat_t = c(20,20)  #True model parameters
results41 <- Simulation(nrep=nrep,N=N,T=T,rho_IV=rho_IV,th_IV=th_IV,rho_t=rho_t,th_t=th_t,mumat_t=mumat_t,pyrs,acc_mus,acc_loglik,seed.start)
meanbias41 <- MeanBias(results41,rho_t,th_t,mumat_t)
RMSE41 <- RMSE(results41,rho_t,th_t,mumat_t)

#Simulation study 2
seed.start <- 10000
nrep = 500  #Number of replications
N = 20  #Number of studies
rho_IV = 0.9; th_IV = 1  #Initial estimates for rho and xi
rho_t = 0.5; th_t = 1; mumat_t = c(20,20)  #True model parameters
results42 <- Simulation(nrep=nrep,N=N,T=T,rho_IV=rho_IV,th_IV=th_IV,rho_t=rho_t,th_t=th_t,mumat_t=mumat_t,pyrs,acc_mus,acc_loglik,seed.start)
meanbias42 <- MeanBias(results42,rho_t,th_t,mumat_t)
RMSE42 <- RMSE(results42,rho_t,th_t,mumat_t)

#Simulation study 3
seed.start <- 11000
nrep = 500  #Number of replications
N = 20  #Number of studies
Appendix A. R-code EM

108

rho_IV = 0.5; th_IV = 4 # Initial estimates for rho and xi
rho_t = 0.5 ; th_t = 1 ; mumat_t = c(20,20) # True model parameters

results43 <- Simulation(nrep=nrep,N=N,T=T,rho_IV=rho_IV,th_IV=th_IV,rho_t=rho_t,th_t=th_t,mumat_t=mumat_t,pyrs,acc_mus,acc_loglik,seed.start)
meanbias43 <- MeanBias(results43,rho_t,th_t,mumat_t)
RMSE43 <- RMSE(results43,rho_t,th_t,mumat_t)

#########################################################################
##Simulation study 4
#########################################################################
seed.start <- 12000
nrep = 500 # Number of replications
N = 20 # Number of studies
rho_IV = 0.5; th_IV = 2 # Initial estimates for rho and xi
rho_t = 0.5 ; th_t = 1 ; mumat_t = c(20,20) # True model parameters

results44 <- Simulation(nrep=nrep,N=N,T=T,rho_IV=rho_IV,th_IV=th_IV,rho_t=rho_t,th_t=th_t,mumat_t=mumat_t,pyrs,acc_mus,acc_loglik,seed.start)
meanbias44 <- MeanBias(results44,rho_t,th_t,mumat_t)
RMSE44 <- RMSE(results44,rho_t,th_t,mumat_t)

A.2 Single Simulation

# Create matrices to store results of two estimation procedures
estmat2stage <- matrix(NA,nrep,T+2)
estmatEM <- matrix(NA,nrep,T+5)

for (irep in 1:nrep){
cat("Replication",irep,\"n")
flush.console()
events <- SimData(N=N,T=T,irep=irep,rho=rho_t,th=th_t,mumat=mumat_t)

data in a long format
datalong <- data.frame(Time=rep(1:T,N),Study=rep(1:N,each=T),count=as.vector(t(events)),pyrs=1)

# Estimates through 2-stage procedure
twostage_est <- TwoStageEstimates(datalong=datalong,T=T)
estmat2stage[irep,1:T] <- twostage_est[1:T]
estmat2stage[irep,T+1] <- twostage_est[T+1]
estmat2stage[irep,T+2] <- twostage_est[T+2]

# Estimates through EM-algorithm
EM_est <- EM_estimates(datalong=datalong,N=N,rho_IV=rho_IV,ksi_IV=(1/th_IV),acc_mus=acc_mus,acc_loglik)
estmatEM[irep,1] <- EM_est["mu_1"]
estmatEM[irep,2] <- EM_est["mu_2"]
estmatEM[irep,3] <- 1/EM_est["theta"]
estmatEM[irep,4] <- EM_est["rho"]
estmatEM[irep,5] <- EM_est["Total"]
estmatEM[irep,6] <- EM_est["E_step"]
Appendix A. R-code EM

estmatEM[irep,7] <- EM_est["M_step"]
results <- cbind(estmatEM,estmat2stage)
colnames(results) <- c("mu_1_EM","mu_2_EM","xi_EM","rho_EM","total_EM","E_step","M_step","mu_1_two","mu_2_two","xi_two","rho_two")
return(results)

A.3 Simulation of Data

SimData <- function(N,T,irep,rho,th,mumat){
  events <- Z <- matrix(NA,N,T)
  set.seed(irep)
  for (i in 1:N) {
    # generate vector of elements X_{i+} ~ Ga(th(1-rho)rho^{T+1-i})
    Xk <- rgamma(T, th*(1-rho)*rho^(T:1),rate = th)
    # generate vector of elements X_{+j} ~ Ga(th(1-rho)rho^j,th), l=1,...,T
    XL <- rgamma(T, th*(1-rho)*rho^[1:T],rate = th)
    # generate X_{++} ~ Ga(th*rho^{T+1},th)
    XX <- rgamma(1, th*rho^[T+1],rate = th)
    # generate matrix X_{ij}
    Xkl <- matrix(0,T,T)
    for (k in 1:T)
      for(l in k:T)
        Xkl[k,l] <- rgamma(1,th*(1-rho)^2*rho^[l-k],rate=th)
    # generate frailties Z_t
    for(t in 1:T)
      Z[i,t] <- sum(Xk[1:t]) + sum(XL[t:T]) + XX + sum(Xkl[1:t,t:T])
    # generate number of events from a Poisson(lambda) with
    # lambda=mu[i,t]*Z[i,t] m=exp(X*beta) ipvmumat
    events[i,] <- rpois(T,mumat*Z[i,])
  }
  return(events)
}

A.4 EM-algorithm
Appendix A. R-code EM

E_step <- function(y_1,y_2,a_1,a_2,a_3,b,mu_1,mu_2){
  #Function to calculate expected value of x_1, x_2, x_3
  #Input:
  # The count data y_1, y_2 and current estimates of model parameters mu_1, 
  # mu_2, rho, xi,
  # a_1,a_2,a_3 depend on rho and xi (e.g. a_1 = rho/xi),
  #Output:
  # Expected values of x_1,x_2,x_3
  #Variables to store expected values x_1,x_2,x_3 in
  e_x_1 <- 0; e_x_2 <- 0; e_x_3 <- 0
  #Variables to store denominator and numerator for x_1,x_2,x_3 in
  den <- 0; num_1 <- 0; num_2 <- 0; num_3 <- 0
  #In the following variables, matrices are stored which contain elements
  # to be summed over to calculate the denominator and numerators
  h_0 <- 0; h_1 <- 0; h_2 <- 0; h_3 <- 0; h_4 <- 0
  h_1_extra <- 0; h_2_extra <- 0; h_3_extra <- 0
  #Values to be summed over are calculated
  h_0 <- outer(0:y_1, 0:y_2, FUN=function(r,c) lchoose(y_1,r) + lchoose(y_2,c))
  h_1 <- outer(0:y_1, 0:y_2, FUN=function(r,c) lgamma(r+c+a_1) - (r+c+a_1)*
              log(mu_1+mu_2+b))
  h_2 <- outer(0:y_1, 0:y_2, FUN=function(r,c) lgamma(y_1-r+a_2) - (y_1-r+a_2)*
              log(mu_1+b))
  h_3 <- outer(0:y_1, 0:y_2, FUN=function(r,c) lgamma(y_2-c+a_3)-(y_2-c+a_3)*
              log(mu_2+b))
  h_1_extra <- outer(0:y_1, 0:y_2, FUN=function(r,c) lgamma(r+c+a_1+1) - (r+
              c+a_1+1)*log(mu_1+mu_2+b))
  h_2_extra <- outer(0:y_1, 0:y_2, FUN=function(r,c) lgamma(y_1-r+a_2+1) -
              (y_1-r+a_2+1)*log(mu_1+b))
  h_3_extra <- outer(0:y_1, 0:y_2, FUN=function(r,c) lgamma(y_2-c+a_3+1) -
              (y_2-c+a_3+1)*log(mu_2+b))
  #Values to be summed over where there is an extra ‘+1’ in the terms
  h_1_extra <- outer(0:y_1, 0:y_2, FUN=function(r,c) lgamma(r+c+a_1+1) - (r+
              c+a_1+1)*log(mu_1+mu_2+b))
  h_2_extra <- outer(0:y_1, 0:y_2, FUN=function(r,c) lgamma(y_1-r+a_2+1) -
              (y_1-r+a_2+1)*log(mu_1+b))
  h_3_extra <- outer(0:y_1, 0:y_2, FUN=function(r,c) lgamma(y_2-c+a_3+1) -
              (y_2-c+a_3+1)*log(mu_2+b))
  #Calculate constant to substract from both denominator and numerator
  # to make sure the function handles high counts well
  C <- mean(h_0+h_1+h_2+h_3)
  #Denominator and numerators of x_1,x_2,x_3 are calculated
  den <- sum(exp(h_0+h_1+h_2+h_3-C))
  num_1 <- sum(exp(h_0+h_1_extra+h_2+h_3-C))
  num_2 <- sum(exp(h_0+h_1+h_2_extra+h_3-C))
  num_3 <- sum(exp(h_0+h_1+h_2+h_3_extra-C))
  #Calculating expected values
  e_x_1 <- num_1 / den
  e_x_2 <- num_2 / den
  e_x_3 <- num_3 / den
  #Calculating log observed data likelihood with new values of x_1,x_2,x_3
  #A call to ‘obs_log_lik’ would also be an option, but then part of the
  # calculations
  # would be done twice
  loglik <- (a_1+a_2+a_3)*log(b)-lgamma(a_1)-lgamma(a_2)-lgamma(a_3)+y_1*
            log(mu_1)+y_2*log(mu_2)-lfactorial(y_1)-lfactorial(y_2)+C+log(den)
  return(cbind(e_x_1,e_x_2,e_x_3,-loglik))
}

M_step <- function(data,N) {
  #Function which calculates estimates for mu_1, mu_2
  #Input:
  # count data y_1,y_2 and current values of gamma variables x_1,x_2,x_3,
  # all stored in ‘data’
  # number of centers N
  #Output:
  # Estimates of mu_1 and mu_2
# Number of time points, defined for clarity
T <- 2

# Compute frailty terms
z1 <- data$x_1 + data$x_2
z2 <- data$x_1 + data$x_3

# Put counts y_1, y_2 and frailty terms z_1, z_2 in data long format
# to be able to let glm work on it
datalong <- data.frame(id = rep(1:N, rep(T, N)),
  time = rep(1:T, N),
  y = as.vector(t(as.matrix(cbind(data$y_1, data$y_2)))),
  z = as.vector(t(as.matrix(cbind(z1, z2)))))

# Calculate log of frailty terms
datalong$logz <- log(datalong$z)

data$long$time <- factor(data$long$time)

# Fit a poisson regression model on the time points with frailty terms
# treated as offset
glmfit <- glm(y ~ time + offset(logz), data = datalong, family = "poisson")

# Calculate mu_1, mu_2 by using regression coefficients
betas <- glmfit$coef
mus <- exp(c(betas[1], betas[1] + betas[-1]))
names(mus) <- c("mu_1", "mu_2")

# Return mu_1, mu_2
return(mus)

obs_log_lik <- function(y_1, y_2, N, rho, theta, mu_1, mu_2) {
  # Function which calculates the observed log likelihood
  # Input:
  # count data y_1, y_2 and N and current parameter estimates rho, theta, mu_1 and mu_2
  # Output:
  # minus observed log likelihood
  # Variable in which output will be stored
  loglik <- 0

  # In the following variables, matrices are stored which contain elements
  # to be summed over to calculate the log likelihood
  h_0 <- 0; h_1 <- 0; h_2 <- 0; h_3 <- 0; h_4 <- 0

  # Calculation of profile log likelihood
  for (i in 1:N) {
    h_0 <- (a_1+a_2+a_3) * log(b) - lgamma(a_1)-lgamma(a_2)-lgamma(a_3)+y_1[i]*log(mu_1)+y_2[i]*log(mu_2)-lfactorial(y_1[i])-lfactorial(y_2[i])
    h_1 <- outer(0:y_1[i], 0:y_2[i], FUN=function(r,c) lchoose(y_1[i],r) + lchoose(y_2[i],c))
    h_2 <- outer(0:y_1[i], 0:y_2[i], FUN=function(r,c) lgamma(y_1[i]-r+a_2) + lgamma(y_2[i]-c+a_3)+lgamma(r+c+a_1))
    h_3 <- outer(0:y_1[i], 0:y_2[i], FUN=function(r,c) -(y_1[i]-r+a_2)*log(mu_1+b)-(y_2[i]-c+a_3)*log(mu_2+b)-(r+c+a_1)*log(mu_1+mu_2+b))
    h_4 <- 0
  }

  # Calculate constant to add and subtract such that the expression...
# handles high counts better
C <- mean(h_1+h_2+h_3)

#Calculate profile log likelihood
h_4 <- sum(exp(h_1+h_2+h_3-C))
loglik <- loglik+ h_0 + C + log(h_4)

} return(-loglik)

} # This is the MAIN function which calls to the above functions

EM_estimates <- function(datalong, N, rho_IV, ksi_IV, acc_mus, acc_loglik){
#Input:
# count data in datalong format,
# number of centers N,
# initial estimates (values) for rho and xi,
# accuracy parameters acc_mus and acc_loglik which indicates when the
# EM-algorithm terminates
#
#Output:
# Estimates for mu_1, mu_2, rho, theta=(1/xi)
# and execution times of different parts of the algorithm
#
#Store count-data in better readable variables
y_1 <- subset(datalong$count, datalong$Time == 1)
y_2 <- subset(datalong$count, datalong$Time == 2)

#Make dataframe to store current estimates of x_1, x_2, x_3, minus
observed likelihood, and y_1, y_2
x_1 <- rep(0,N)
x_2 <- rep(0,N)
x_3 <- rep(0,N)
minloglik <- rep(0,N)
x_and_y <- data.frame(x_1, x_2, x_3,minloglik, y_1, y_2)

#Initial (biased) estimates for mu_1 and mu_2
mu_1_new <- mean(y_1)
mu_2_new <- mean(y_2)

#To keep track of computation time for different parts of algorithm,
# we define a couple of variables to store process times in
proc_time_total <- proc.time()
proc_time_current <- proc.time()
proc_time_previous <- 0
time_E_step <- 0; time_M_step <- 0

prof_log_lik <- function(par){
#Function which returns minus profile log likelihood for values of rho
# and theta
#par[1] = rho
#par[2] = theta
#Variable which keeps track of the number of iterations of the EM-
algorithm
count <- 0

#Make sure the EM-algorithm starts
diff_mus <- acc_mus+2
diff_loglik <- acc_loglik+2

#Variable in which current and previous observed log likelihood values
are stored
loglik_old <- 0
loglik_new <- 0

#EM-algorithm, terminates when either the difference in mu’s or
difference in
# consecutive values of the observed log likelihood are below the
accuracy parameters
while(diff_mus >= acc_mus && diff_loglik >= acc_loglik){
#For the new iteration, the previous values of the parameters have to be stored as 'old' to keep track of when the algorithm terminates
mu_1_old <- mu_1_new
mu_2_old <- mu_2_new

#Keep track of number of iterations of algorithm
count = count+1

#E-step: update values of x_1,x_2,x_3 based on previous estimates of mu_1,mu_2
proc_time_current <- proc.time()
for(i in 1:N){x_and_y[i,1:4] <- E_step(y_1=y_1[i], y_2=y_2[i], a_1=par[2]*par[1], a_2 = par[2]*(1-par[1]), a_3=par[2]*(1-par[1]), b=par[2], mu_1=mu_1_old, mu_2=mu_2_old)}
proc_time_previous <- proc_time_current; proc_time_current <- proc.time(); time_E_step <<- time_E_step + proc_time_current['elapsed']
loglik_old <- loglik_new
loglik_new <- sum(x_and_y[,4])

#M-step: update values of mu_1,mu_2 based on new estimates x_1,x_2,x_3 and store in help variable h1
h1 <- M_step(data = x_and_y,N=N)
proc_time_previous <- proc_time_current; proc_time_current <- proc.time(); time_M_step <<- time_M_step + proc_time_current['elapsed']

#Store new values of mu_1,mu_2
mu_1_new <<- h1['mu_1']
mu_2_new <<- h1['mu_2']

#Calculate values which
diff_mus <- abs(mu_1_new-mu_1_old) + abs(mu_2_new-mu_2_old)
diff_loglik <- abs(loglik_old-loglik_new)

#if(is.na(diff_mus >= acc_mus && diff_loglik >= acc_loglik)){print(x_and_y);print(mu_1_new,mu_2_new)}
}

profloglik <- prof_log_lik(y_1,y_2,N,par[1],par[2],mu_1_new,mu_2_new)
cat(par,mu_1_new,mu_2_new,"=>",profloglik, " ",count,"\n")
return(profloglik)

#The funtion 'opt' optimizes the function 'prof_log_lik' over rho and theta
#It searches for rho in the interval [0.001,0.999] and for theta in [0.001,100]
opt <- optim(par=c(rho_IV,(1/ksi_IV)), fn=prof_log_lik, method="L-BFGS-B", lower= c(0.001,0.001), upper=c(0.999,100))
proc_time_total <- proc.time()-proc_time_total

#We store the results (rho,theta,minloglik) in the help variable together with computation times
h <- c(mu_1_new,mu_2_new,opt$par,proc_time_total['elapsed'],time_E_step,time_M_step)
names(h) <- c("mu_1","mu_2","rho","theta","Total","E_step","M_step")

#Last execution of EM-algorithm to calculate values of mu_1,mu_2 which are accurate up to acc_mus
cat("Last loop","\n")
diff_mus <- acc_mus*2
while (diff_mus >= acc_mus){
  mu_1_old <- mu_1_new
  mu_2_old <- mu_2_new
  for(i in 1:N){x_and_y[i,1:4] <- E_step(y_1=y_1[i], y_2=y_2[i], a_1=h["theta"]*h["rho"]+h["theta"]*(1-h["rho"])), a_2 = h["theta"]*(1-par[1]), a_3=par[2]-(1-par[1]), b=par[2], mu_1=mu_1_old, mu_2=mu_2_old)}
  h1 <- M_step(data = x_and_y,N=N)
  #Store new values of mu_1,mu_2
A.5 Two-stage Procedure

```r
library(MASS)
colrep <- function(v,n){
  # function to repeat vector v n times
  # Input:
  # v: vector
  # n: number of columns in the matrix
  # Output:
  # matrix of repeated columns, dimension: length(v)X n
  return(matrix(rep(v,n),length(v),n))
}

loglik1.ist<-
function (y1, y2, mu1, mu2, xi, rhost)
{
  mu12 <- mu1 + mu2
  theta <- 1/xi
  rho <- rhost
  Pist1 <- dnbinom(y1:0, size = theta * (1 - rho), mu = mu1 *
                   (1 - rho))
  Pist2 <- dnbinom(y2:0, size = theta * (1 - rho), mu = mu2 *
                   (1 - rho))
  P1 <- colrep(Pist1, y2 + 1)
  P2 <- rowrep(Pist2, y1 + 1)
  outerm <- outer(0:y1, 0:y2, "*")
  outerv <- as.vector(outerm)
  helpv <- as.vector(colrep(0:y1, 0:y2, "="))
  P3 <- matrix(dnbinom(outerv, size = theta * rho, mu = mu12 *
                     rho), y1 + 1, y2 + 1)
  P4 <- matrix(dbinom(helpv, outerv, mu1/mu12), y1 + 1, y2 + 1)
  P <- P1 * P2 * P3 * P4
  return(logP = log(sum(P)))
}
corr.AR <- function(rho,T){
  # Function to create correlations matrix for autoregressive models

```
Appendix A. R-code EM

```r
# Input:
# rho: correlation parameter
# T: number of time points
R <- diag(T)
for (s in 1:T)
  for (t in 1:T)
    R[s,t] <- rho^(abs(s-t))
return(R)
```

```r
pcgf.loglik <-
  function (ymat, mumat, th, rho)
  {
    xi <- 1/th
    N <- nrow(ymat)
    T <- ncol(ymat)
    loglik <- 0
    R <- corr.AR(rho, T)
    for (i in 1:N) {
      for (s in 1:(T - 1)) {
        yis <- ymat[i, s]
        muis <- mumat[i, s]
        for (t in (s + 1):T) {
          yit <- ymat[i, t]
          muit <- mumat[i, t]
          rhost <- R[s, t]
          loglik <- loglik + loglik1.ist(yis, yit, muis,
                                      muit, xi, rhost)
        }
      }
    }
    return(sum(loglik))
  }

loglikrho <-
  function (rho, y, mu, th)
  {
    res <- pcgf.loglik(y, mu, th, rho)
    return(res)
  }
```

```r
PoisCorrGammFrailty <-
  function (formula, data, rho = TRUE)
  {
    data$Study <- as.factor(data$Study)
    data$Time <- as.factor(data$Time)
    formula <- as.formula(formula)
    glmnb <- glm.nb(formula, data = data, link = "log")
    bbeta <- glmnb$coef
    b <- c(bbeta[1], bbeta[-1] + bbeta[1])
    th <- glmnb$theta
    xi <- 1/th
    p <- length(bbeta)
    n <- nrow(data)
    T <- length(unique(data$Time))
    N <- n/T
    y <- glmnb$y
    mu <- glmnb$fit
    x <- y - mu
    X <- model.matrix(glmnb)
    mumat <- matrix(mu, N, T, byrow=TRUE)
    if (!rho)
      return(list(b = b, th = th, xi = xi, glmnb = glmnb, mumat = mumat))
    else {
      ymat <- matrix(y, N, T)
      opt <- optimize(f = loglikrho, interval = c(0, 1), lower = 0,
                       upper = 1, maximum = TRUE, tol = .Machine$double.eps ^0.25,
                       y = ymat, mu = mumat, th = th)
      rho <- opt$maximum
      return(list(b = b, th = th, xi = xi, rho = rho, glmnb = glmnb,
```
A.6 Standard Error

```r
### Function to calculate standard error by bootstrap ####
source("SimulationData.R")
source("EmAlgorithm.R")

StandardError.Bootstrap <- function(nrep,mumat,rho,theta,acc_mus, acc_loglik) {
  # Function to perform bootstrap which estimates standard errors
  # Input:
  # nrep: number of bootstrap simulations
  # mumat, rho, theta: estimated values for parameters for which
  # variance has to be calculated
  # Output:
  # Standard errors for mu, rho and theta through bootstrap
  T <- ncol(mumat)
  N <- nrow(mumat) # Number of studies or centers
  boot.mat.2stage <- matrix(NA,nrep,T+2) # Empty matrix to store results in
  
  for (irep in 1:nrep) {
    cat("Replication",irep,"of",nrep, "\n") # So that user can keep track of progress
    
    # Simulation of data
    data <- SimData(N,T,irep,rho,theta,mumat)
    datalong <- data.frame(Time=rep(1:T,N),Study=rep(1:N,each=T),count=as.vector(t(data)),pyrs=1)
    
    # Estimation of parameters for simulated data
    estimates <- EM_estimates(datalong, N, rho_IV=rho, ksi_IV=theta, acc_mus, acc_loglik)
    print("Estimates")
    print(estimates[1:(T+2)])
    boot.mat.2stage[irep,] <- estimates[1:(T+2)]
  }

  # Calculate standard errors
  SE <- rep(0,T+2)
  for(i in 1:(T+2)) {
    SE[i] <- sqrt(sum((boot.mat.2stage[,i] - mean(boot.mat.2stage[,i]))^2)/(nrep-1))
  }
  names(SE) <- c("mu1","mu2","rho","th")
  return(SE)
}
```

T <- 2
Appendix A. R-code EM

```r
nrep <- 50
N <- 20
mu <- c(10,10)
mumat <- matrix(rep(mu,N),N,T)
rho <- 0.5
theta <- 1
acc_mus <- 1e-05 #Accuracy parameter for EM-algorithm
acc_loglik <- 1e-05 #Accuracy parameter for EM-algorithm

SE <- StandardError.Bootstrap(nrep,mumat,rho,theta,acc_mus, acc_loglik)
```

./StandardError.R
Appendix B

R-code Flexible Two-stage

B.1 Main R-file

```r
##########################################################################
#Main Code Simulation Study
##########################################################################

callback()
#Read in required R-files
source("SimulationData.R")
source("EstimationParameters.R")
source("TwoStage.R")
source("SingleSimulation.R")

callback()
#Function to calculate mean bias
MeanBias <- function(results,rho_t,th_t,mumat_t){
  #Input:
  # Results of a simulation (see SingleSimulation.R for details on this
  # matrix)
  # True values of rho, xi, mu_1 and mu_2
  #Output:
  # matrix with mean biases for rho,xi,mu_1,mu_2 (rows)
  # and for two estimation procedures EM and two-stage (columns)
  Biasses <- matrix(NA,nrow=9,ncol=1)
  colnames(Biasses) <- c("bias")
  rownames(Biasses) <- c("mu1","mu2","mu3","mu4","mu5","mu6","mu7","xi","rho")

  Biasses["mu1","bias"] <- mean(results[,"mu1.f"] - mumat_t[1,1])
  Biasses["mu2","bias"] <- mean(results[,"mu2.f"] - mumat_t[1,2])
  Biasses["mu3","bias"] <- mean(results[,"mu3.f"] - mumat_t[1,3])
  Biasses["mu4","bias"] <- mean(results[,"mu4.f"] - mumat_t[1,4])
  Biasses["mu5","bias"] <- mean(results[,"mu5.f"] - mumat_t[1,5])
  Biasses["mu6","bias"] <- mean(results[,"mu6.f"] - mumat_t[1,6])
  Biasses["mu7","bias"] <- mean(results[,"mu7.f"] - mumat_t[1,7])
  Biasses["xi","bias"] <- mean(results[,"xi.f"] - 1/th_t)
  Biasses["rho","bias"] <- mean(results[,"rho.f"] - rho_t)
  return(Biasses)
}

callback()
#Function to calculate mean bias
RMSE <- function(results,rho_t,th_t,mumat_t){
  #Input:
  # Results of a simulation (see SingleSimulation.R for details on this
  # matrix)
  # True values of rho, xi, mu_1 and mu_2
  #Output:
  # matrix with mean biases for rho,xi,mu_1,mu_2 (rows)
  # and for two estimation procedures EM and two-stage (columns)
  RMSE <- matrix(NA,nrow=9,ncol=1)
  colnames(RMSE) <- c("RMSE")
  rownames(RMSE) <- c("mu1","mu2","mu3","mu4","mu5","mu6","mu7","xi","rho")
  return(RMSE)
}
```
Appendix B. R-code Flexible Two-stage

50 \(xi_t \leftarrow \frac{1}{th_t}\)
51 N \leftarrow nrow(results)
52
53 RMSE["mu1","RMSE"] \leftarrow \sqrt{\frac{\text{sum}((\text{results}[,\text{"mu1.f"}] \text{ - mumat}_t[1,1])^2)}{N}}
54 RMSE["mu2","RMSE"] \leftarrow \sqrt{\frac{\text{sum}((\text{results}[,\text{"mu2.f"}] \text{ - mumat}_t[1,2])^2)}{N}}
55 RMSE["mu3","RMSE"] \leftarrow \sqrt{\frac{\text{sum}((\text{results}[,\text{"mu3.f"}] \text{ - mumat}_t[1,3])^2)}{N}}
56 RMSE["mu4","RMSE"] \leftarrow \sqrt{\frac{\text{sum}((\text{results}[,\text{"mu4.f"}] \text{ - mumat}_t[1,4])^2)}{N}}
57 RMSE["mu5","RMSE"] \leftarrow \sqrt{\frac{\text{sum}((\text{results}[,\text{"mu5.f"}] \text{ - mumat}_t[1,5])^2)}{N}}
58 RMSE["mu6","RMSE"] \leftarrow \sqrt{\frac{\text{sum}((\text{results}[,\text{"mu6.f"}] \text{ - mumat}_t[1,6])^2)}{N}}
59 RMSE["mu7","RMSE"] \leftarrow \sqrt{\frac{\text{sum}((\text{results}[,\text{"mu7.f"}] \text{ - mumat}_t[1,7])^2)}{N}}
60 RMSE["xi","RMSE"] \leftarrow \sqrt{\frac{\text{sum}((\text{results}[,\text{"xi.f"}] \text{ - xi}_t)^2)}{N}}
61 RMSE["rho","RMSE"] \leftarrow \sqrt{\frac{\text{sum}((\text{results}[,\text{"rho.f"}] \text{ - rho}_t)^2)}{N}}
62 return(RMSE)
63 #RMSE
64
65 #Simulation study 1:
66 #Simulation study 2:
67 #Simulation study 3:
### Appendix B. R-code Flexible Two-stage

#### B.2 Single Simulation

```r
#Simulation study 4:
#########################################################################
#Simulation study 4:
#########################################################################
cat("Simulation study 4","\n")

seed.start <- 3000
nrep = 1000 #Number of replications
N = 20    #Number of studies
TP <- c(0.25,0.5,1,1.75,2.75,4,5.5)
rho_t = 0.5 ; th_t = 1 #True model parameters
mu <- rep(10,7)
mumat_t = rowrep(mu,N)

Results4 <- Simulation(nrep,N,TP,rho_t,th_t,mumat_t,seed.start)
Bias4 <- MeanBias(Results4,rho_t,th_t,mumat_t)
RMSE4 <- RMSE(Results4,rho_t,th_t,mumat_t)

#########################################################################
#Simulation study 5:
#########################################################################
cat("Simulation study 5","\n")

seed.start <- 4000
nrep = 1000 #Number of replications
N = 20    #Number of studies
TP <- c(0.1,0.2,0.4,0.8,1.6,3.2,6.4)
rho_t = 0.5 ; th_t = 1 #True model parameters
mu <- rep(10,7)
mumat_t = rowrep(mu,N)

Results5 <- Simulation(nrep,N,TP,rho_t,th_t,mumat_t,seed.start)
Bias5 <- MeanBias(Results5,rho_t,th_t,mumat_t)
RMSE5 <- RMSE(Results5,rho_t,th_t,mumat_t)

./Mainflex.R
```

#### Simulation study

```r
#Simulation study
#########################################################################
### Single simulation
#########################################################################
Simulation <- function(nrep,N,TP,rho_t,th_t,mumat_t,seed.start){

T <- length(TP)

estmat2stage <- matrix(NA,nrep,T+3)
estmat2stage_flex <- matrix(NA,nrep,T+3)

#Simulation study
for(irep in 1:nrep){
    #User can keep track of progress
cat("Repitition",irep, "of", nrep, "\n")

    #Simulation of data
    events <- SimData(N,TP,seed.start+irep,rho_t,th_t,mumat_t)

    #Processing time flexible
    proc_time_current <- proc.time()

    #Estimation flexible
    estmat2stage_flex[irep,1:(T+2)] <- Estimation.2.stage(events)

    #Processing time flexible
    proc_time_previous <- proc_time_current
```

```
Appendix B. R-code Flexible Two-stage

```
proc_time_current <- proc.time()
estmat2stage_flex[irep,T+3] <- proc_time_current["elapsed"] - proc_time_previous["elapsed"]

#Add person years to data
events$pyrs <- 1

#Estimation non-flexible
estmat2stage[irep,1:(T+2)] <- TwoStageEstimates1(datalong=events,T=T)

#Processing time non-flexible
proc_time_previous <- proc_time_current
proc_time_current <- proc.time()
estmat2stage[irep,T+3] <- proc_time_current["elapsed"] - proc_time_previous["elapsed"]

}#nrep
results <- cbind(estmat2stage_flex)
colnames(results) <- c("mu1","mu2","mu3","mu4","mu5","mu6","mu7","xi","rho","time")
return(results)
```

./SingleSimulationflex.R

B.3 Simulation of Data

```
library(MASS)

colrep <- function(v,n){
  #function to repeat vector v n times
  # Input:
  # v: vector
  # n: number of columns in the matrix
  # Output:
  # matrix of repeated columns, dimension: length(v) by n
  return(matrix(rep(v,n),length(v),n))
}

corr.AR <- function(rho,TP){
  #Function to create correlations matrix for autoregressive models
  #Input:
  # rho: correlation parameter
  # TP: time points
  #Output:
  # T by T matrix R where R[i,j] is the correlation between Z_i and Z_j
  T <- length(TP)
  R <- outer(1:T,1:T, FUN <- function(r,c) rho^(abs(TP[r]-TP[c])))
  return(R)
}
Appendix B. R-code Flexible Two-stage

41 shape.matrix <- function(rho, theta, TP){
42 #Function to create matrix containing shapes of X_ij’s
43 #Input:
44 # rho and theta: values of model parameter parameters
45 # TP: vector of time points
46 #Output:
47 # T by T matrix shape.mat where the shape.mat[i,j] is the shape of X_ij
48 T <- length(TP) #Number of time points
49 rho.mat <- corr.AR(rho, TP) #Correlation matrix
50 shape.mat <- matrix(0, T, T) #Empty shape matrix
51 #The shape for i=1, j=T
52 shape.mat[1,T] <- rho.mat[1,T]*theta
53 #The shape for i=1, j=1:(T-1)
54 for(j in 1:(T-1)){shape.mat[1,j] <- (rho.mat[1,j]-rho.mat[1,j+1])*theta}
55 #The shape for i=2:T, j=T
56 for(i in 2:T){shape.mat[i,T] <- (rho.mat[i,T]-rho.mat[i-1,T])*theta}
57 #IF T=2, these steps are unneccesary
58 #The shape for the other indices
59 if(T != 2){
60   for(i in 2:(T-1)){
61     for(j in i:(T-1)){shape.mat[i,j] <- (rho.mat[i,j] - rho.mat[i-1,j] - rho.mat[i,j+1] + rho.mat[i-1,j+1])*theta}
62   }
63 }
64 return(shape.mat)
65 }
66 }
67 sum.frailty <- function(gamma.mat){
68 #This function sums over the elements of X_ij to calculate Z_t
69 #Z_t = sum_(j=t)^T sum_(i=1)^t X_ij
70 #Input:
71 # T by T matrix containing simulated values of X_ij
72 #Output:
73 # Vector Z of length T where the t'th element is Z_t
74 T <- nrow(gamma.mat)
75 Z <- rep(0, T)
76 for(t in 1:T){
77   sum = 0
78   for(j in t:T){
79     for(i in 1:t){sum = sum + gamma.mat[i,j]}
80   }
81   Z[t] <- sum
82 }
83 return(Z)
84 }
85 SimFrailty <- function(N, TP, irep, rho, theta){
86 #Function which simulates serially correlated gamma frailty terms
87 #Input:
88 # N, TP: number of studies and time points
89 # irep: number of previous simulations (for input in set.seed())
90 # rho, theta: true values of model parameters
91 #Output:
92 # matrix with N rows and T columns containing simulated Z_1,...,Z_T
93 set.seed(irep) #This is done so that results can be replicated
94 T <- length(TP)
95 Z <- matrix(0, nrow=N, ncol=T) #Empty matrix to store Z_1,...,Z_T for every study
96 gamma.mat <- matrix(0, T, T) #Empty matrix to store the X_ij
97 shape.mat <- shape.matrix(rho, theta, TP) #Calculate shapes of X_ij’s
98 for(n in 1:N){
99   for(i in 1:T){
100     for(j in i:T){
101       #Generate X_ij’s
102       gamma.mat[i,j] <- rgamma(n=1, shape=shape.mat[i,j], rate=theta)
103     }
104   }
105   for(t in 1:T){
106     sum = 0
107     for(j in t:T){
108       for(i in 1:t){sum = sum + gamma.mat[i,j]}
109     }
110     Z[t] <- sum
111   }
112   return(Z)
113 }
114 }
Appendix B. R-code Flexible Two-stage

```r
#Calculate frailty’s by summing over X_ij’s
Z[n,] <- sum.frailty(gamma.mat)
return(Z)
```

```r
SimData <- function(N,TP,irep,rho,theta,mumat){
#Function which simulates longitudinal count data with serially
#correlated
# gamma frailty terms
#Input:
# number of studies ‘N’, time points ‘TP’
# number of previous simulations ‘irep’ (for input in set.seed())
# true values of model parameters ‘rho’, ‘theta’
#Output:
# matrix with N rows and T columns containing simulated y_1,...,y_T for N
#studies
#Simulate the frailty’s
Z <- SimFrailty(N,TP,irep,rho,theta)
#Simulate count data y_1,dots,y_T
T <- length(TP)
events <- matrix(0,N,T) #Empty matrix to store count data
#Simulate count data
for(n in 1:N){events[n,] <- rpois(T,mumat[n,]*Z[n,])}
#Store data in data long frame
events.DL <- data.frame(Time=rep(TP,N),Study=rep(1:N,each=T),count=as.
vectoe(t(events)))
return(events.DL)
}
```

.//SimulationDataModel2.R

B.4 Estimation of Parameters

```r
source("CompositeLikelihoodPairs.R")
Estimation.2.stage <- function(data){
#Function which estimates mu’s, theta and rho by using flexible 2-stage
#method
#Input:
# data: count data in datalong format
#Output:
# estimates for mu’s, theta and rho
#Extract information from data
tP <- unique(data$Time) #Time Points
tN <- length(tP) #Number of time points
N <- nrow(data)/tN #Number of studies
#Optimization of composite likelihood of marginal distributions
# to estimate mu’s and theta
glmnb <- glm.nb(count ~ factor(Time), data = data, link = "log")
#Extract mu’s and theta from glmnb
bbeta <- glmnb$coef
b <- c(bbeta[1],bbeta[-1]+bbeta[1])
mu <- exp(b)
mumat <- rowrep(mu,N)
```
theta <- glmnb$theta

# Transforming count data from datalong to N x T matrix
ymat <- matrix(0,N,T)
for(t in 1:T){ymat[,] <- as.numeric(subset(data$count, data$Time==TP[t])}

# Optimization of composite likelihood of all pairs of observations
# to estimate rho
opt <- optimize(f = loglikrho, interval = c(0, 1), lower = 0,
upper = 1, maximum = FALSE, tol = .Machine$double.eps^0.25,
ymat = ymat, mumat = mumat, theta = theta, TP=TP)

# Extract MLE-estimate for rho
rho <- opt$minimum

# Calculate xi
xi <- 1/theta

# Return estimates of parameters mu’s, theta, rho
estimates <- c(mumat[1,],xi,rho)
return(estimates)

.B.5 Pairwise Composite Likelihood

### Functions which calculate the composite likelihood of all pairs of observations

```r
source("SimulationData.R")

loglik.pair <- function(y_s,y_t,mu_s,mu_t,rho_st,theta) {
  # Function which calculates the contribution to the minus log likelihood
  # of a pair of observations
  # Input:
  # y_s,y_t: number of counts for time points s and t
  # mu_s,mu_t,theta: estimated values of mu_s, mu_t and theta
  # rho_st: correlation between Z_s and Z_t
  # Output:
  # Minus log likelihood for pair of observations

  mu_st <- mu_s + mu_t # For convenience of notation
  rho <- rho_st # For convenience of notation

  # First NB-term (NB = Negative Binomial)
  P1 <- dnorm(y_s:0, size = theta * (1 - rho), mu = mu_s * (1 - rho))
  P1 <- colrep(P1, y_t + 1)

  # Second NB-term
  P2 <- dnorm(y_t:0, size = theta * (1 - rho), mu = mu_t * (1 - rho))
  P2 <- rrowrep(P2, y_s + 1)

  # Help matrices
  outerm <- outer(0:y_s, 0:y_t, "+")
  outerv <- as.vector(outerm)
  helpv <- as.vector(c(rep(0:y_s, y_t + 1)))

  # Third NB-term
  P3 <- matrix(dnorm(outerv, size = theta * rho, mu = mu_st * rho), y_s + 1, y_t + 1)

  # The binomial-term
  P4 <- matrix(dbinom(helpv, outerv, mu_s/mu_st), y_s + 1, y_t + 1)
```
# Calculation of likelihood
lik <- P1 * P2 * P3 * P4
return(minloglik = -log(sum(lik)))

loglikrho <- function(rho, ymat, mumat, theta, TP=TP){
  # Function which calculates the composite likelihood for all pairs of observations
  # Input:
  # rho, theta: current estimates for rho and theta
  # ymat: N by T matrix filled with count data
  # mumat: N by T matrix filled with estimates for mu
  # TP: time points
  # Output:
  # Minus composite log likelihood
  loglik <- 0 # Variable in which output will be stored
  R <- corr.AR(rho,TP) # Calculating correlation matrix
  N <- nrow(ymat)
  T <- ncol(ymat)
  # Sum over all studies
  for (i in 1:N) {
    # Sum over all possible unique pairs s,t, where s,t in (1,...,T)
    for (s in 1:(T - 1)) {
      for (t in (s + 1):T) {
        loglik <- loglik + loglik.pair(ymat[i,s], ymat[i,t], mumat[i,s],
                                       mumat[i,t], R[s,t], theta)
      }
    }
  }
  return(loglik)
}

B.6 Standard Error

# Function to calculate standard error by bootstrap
source("SimulationData.R")
source("EstimationParameters.R")

StandardError.Bootstrap <- function(nrep,TP,mumat,rho,theta){
  # Function which calculates bootstrap standard errors for parameters
  # Input:
  # nrep: number of bootstrap simulations
  # TP: vector containing time points
  # mumat,rho,theta: estimated values for parameters for which variance has to be calculated
  # Output:
  # Standard errors for mu, rho and theta through bootstrap
  T <- length(TP) # Number of time points
  N <- nrow(mumat) # Number of studies or centers
  boot.mat.2stage <- matrix(NA,nrep,T+2) # Empty matrix to store results in
  for (irep in 1:nrep) {
    cat("Replication",irep,"of",nrep, "\n") # So that user can keep track of progress
    data <- SimData(N,TP,irep,rho,theta,mumat)
    # Estimation of parameters for simulated data
Appendix B. R-code Flexible Two-stage

```r
boot.mat.2stage[irep,] <- Estimation.2.stage(data)

#Calculate standard errors
SE <- rep(0,T+2)
for(i in 1:(T+2)){
    SE[i] <- sqrt(sum((boot.mat.2stage[,i] - mean(boot.mat.2stage[,i]))^2)/(nrep-1))
}
return(SE)

./StandardErrorflex.R
```
Appendix C

R-code Data Reconstruction

```r
library(foreign)
library(survival)
library(rms)
library(RColorBrewer)

SurvData <- read.spss(file= "surv10854.sav", use.value.labels=TRUE, to.data.frame=TRUE)
total.patients <- nrow(SurvData)

#Delete patients with missing data on one of the entries needed for analysis
SurvData <- subset(SurvData,!is.na(SurvData$treatm)) #Treatment type
#2 deleted
SurvData <- subset(SurvData,!is.na(SurvData$survyrs)) #Survival time
#0 deleted
SurvData <- subset(SurvData,!is.na(SurvData$survstat)) #Survival status at end survival time
#0 deleted
SurvData <- subset(SurvData,!is.na(SurvData$hospno)) #Hospital number
#0 deleted

#Delete patients who do not meet requirements of data
SurvData <- subset(SurvData,(is.na(SurvData$el19)|SurvData$el19!=1)) #preoperative chemotherapy or severe protocol violation
#40 deleted
SurvData <- subset(SurvData,SurvData$tusi!= "any size + local adv") #Delete patients with stage III breast cancer
#42 deleted

#Delete hospitals with less than 100 patients
SurvData <- subset(SurvData,SurvData$hospno != 78)
SurvData <- subset(SurvData,SurvData$hospno != 301)
SurvData <- subset(SurvData,SurvData$hospno != 342)
SurvData <- subset(SurvData,SurvData$hospno != 353)
SurvData <- subset(SurvData,SurvData$hospno != 551)
SurvData <- subset(SurvData,SurvData$hospno != 554)
SurvData <- subset(SurvData,SurvData$hospno != 777)
SurvData <- subset(SurvData,SurvData$hospno != 903)
SurvData <- subset(SurvData,SurvData$hospno != 927)

#Number of patients per hospital number
hospital.numbers <- unique(SurvData$hospno) #hospital numbers
hospital.numbers <- sort(hospital.numbers) #hospital numbers sorted
N.hospital <- length(hospital.numbers) #total number of hospitals

#Number of patients per hospital (for table)
N.patients <- matrix(NA,N.hospital,2)
for(i in 1:N.hospital){
  N.patients[i,1]<-nrow(subset(SurvData, SurvData$hospno == hospital.numbers[i] & SurvData$treatm==1))#perioperative
  N.patients[i,2]<-nrow(subset(SurvData, SurvData$hospno == hospital.numbers[i] & SurvData$treatm==2))#non
}

#Number of deaths per hospital (for table)
N.deaths <- matrix(NA,N.hospital,2)
for(i in 1:N.hospital){

```
Appendix C. R-code Data Reconstruction

```r
N.deaths[i,1]<-nrow(subset(SurvData, SurvData$hospno == hospital.numbers[i] & SurvData$treatm==1 & SurvData$survstat == 2))
N.deaths[i,2]<-nrow(subset(SurvData, SurvData$hospno == hospital.numbers[i] & SurvData$treatm==2 & SurvData$survstat == 2))
}

#Define end points of time intervals in months (for T=2)
TP <- #endpoints for the time intervals need to be inserted here
T <- length(TP)
N <- N.hospital

LongCountDataFrame <- function(TP,T,N,Data,hospital.numbers){
  #Define matrices to be filled
  counts.mat <- matrix(NA,N,T) #matrix for number of events/counts per hospital per time interval
  cens.mat <- matrix(NA,N,T) #matrix for number of censored observations per hospital per time interval
  intlength.mat <- matrix(NA,N,T) #matrix for interval lengths per hospital per time interval
  atriskbegin.mat <- matrix(NA,N,T) #matrix for number at risk at the beginning of the interval per hospital per time interval
  atrisk.mat <- matrix(NA,N,T) #matrix for number at risk per hospital per time interval (see formula thesis)
  pyrs.mat <- matrix(NA,N,T) #matrix with person years per hospital per time interval

  #Fill the counts.mat
  Events <- subset(Data, Data$survstat==2)# 2 means death
  help <- subset(Events,Events$survyrs <= TP[1])
  for(n in 1:N){
    counts.mat[n,1] <- nrow(subset(help, help$hospno == hospital.numbers[n]))
  }
  for(t in 2:T){
    help <- subset(Events, Events$survyrs <= TP[t])
    help <- subset(help, help$survyrs > TP[t-1])
    for(n in 1:N){
      counts.mat[n,t] <- nrow(subset(help, help$hospno == hospital.numbers[n]))
    }
  }

  #Fill the cens.mat
  Cens <- subset(Data, Data$survstat==1)
  help <- subset(Cens, Cens$survyrs <= TP[1])
  for(n in 1:N){
    cens.mat[n,1] <- nrow(subset(help, help$hospno == hospital.numbers[n]))
  }
  for(t in 2:T){
    help <- subset(Cens, Cens$survyrs <= TP[t])
    help <- subset(help, help$survyrs > TP[t-1])
    for(n in 1:N){
      cens.mat[n,t] <- nrow(subset(help, help$hospno == hospital.numbers[n]))
    }
  }

  #Fill interval length matrix
  int.length <- rep(NA,T)
  int.length[1] <- TP[1]
  for(t in 2:T){
    int.length[t] <- TP[t]-TP[t-1]
  }
  intlength.mat <- matrix(rep(int.length,N),N,T,byrow = TRUE)
```
# Fill matrix number at risk at beginning of interval
help <- subset(SurvData, SurvData$survyrs > 0)
for(n in 1:N){
  atriskbegin.mat[n,1] <- nrow(subset(help, help$hospno == hospital.numbers[n]))
}

for(t in 2:T){
  help <- subset(Data, Data$survyrs > TP[t-1])
  for(n in 1:N){
    atriskbegin.mat[n,t] <- nrow(subset(help, help$hospno == hospital.numbers[n]))
  }
}

# Fill the matrix for number at risk
for(t in 1:T){
  atrisk.mat[,t] <- atriskbegin.mat[,t] - (counts.mat[,t] + cens.mat[,t])/2
}

# Fill the matrix for person years
pyrs.mat <- atrisk.mat * intlength.mat

# Create longitudinal count data frame
datalong <- data.frame(Time = rep(TP,N),
                       Study = rep(1:N, rep(T,N)),
                       count = as.vector(t(as.matrix(counts.mat))),
                       pyrs = as.vector(t(as.matrix(pyrs.mat))))

# Total
Datalong <- LongCountDataFrame(TP, T, N, SurvData, hospital.numbers)
count.mat <- matrix(Datalong$count, N, T)

# Control
SurvDataA <- subset(SurvData, SurvData$treatm==2)
DatalongA <- LongCountDataFrame(TP, T, N, SurvDataA, hospital.numbers)
count.matA <- matrix(DatalongA$count, N, T)

# Perioperative
SurvDataB <- subset(SurvData, SurvData$treatm==1)
DatalongB <- LongCountDataFrame(TP, T, N, SurvDataB, hospital.numbers)
count.matB <- matrix(DatalongB$count, N, T)

/DataReconstruction.R
Bibliography

Aalen, O. (2008), *Survival and event history analysis a process point of view*, New York etc. : Springer.


Venn, J. (1880), ‘On a diagrammatic and mechanical representation of propositions and reasonings’, *Philosophical Magazine Series 5* 10(59), 1–18.


