Parameter Estimation for Software Reliability Models

P.H.A. Meyfroyt

July 20, 2012
Summary

The subject of this report is the estimation of the parameters of software reliability models. The models we consider are the Goel-Okumoto model, the Yamada S-Shaped model and the Inflection S-Shaped model. For the Goel-Okumoto model and the Yamada S-Shaped model, we establish necessary and sufficient conditions with respect to the software failure data, which, if satisfied, assure that the Maximum Likelihood Method returns a unique, positive and finite estimation of the unknown parameters. For the Inflection S-Shaped model we establish sufficient conditions with respect to the software failure data, which, if satisfied, assure that the Maximum Likelihood Method returns at least one, positive and finite estimation of the unknown parameters. When we establish the conditions, we consider two types of data, grouped and ungrouped. In case of grouped data, we consider \( k \) predetermined time intervals, defined by \( (\ell_{i-1}, \ell_i) \) for \( i = 1, \ldots, k \). The failure data consists of the number of failures per time interval, denoted by \( y_i \) for \( i = 1, \ldots, k \). The total number of failures is denoted by \( n_k \). In case of ungrouped data we consider the exact failure times of the software, denoted by \( t_i \) for \( i = 1, \ldots, n \). The total number of failures in this case is denoted by \( n \). In Table 1 we state the conditions for the Goel-Okumoto model and the Yamada S-Shaped model. Recall that these conditions are necessary and sufficient for the existence of a unique, positive and finite solution of the Maximum Likelihood equations.

<table>
<thead>
<tr>
<th>Goel-Okumoto Model</th>
<th>Grouped data</th>
<th>( \ell_k n_k &gt; \sum_{i=1}^k y_i (\ell_i + \ell_{i-1}) )</th>
<th>Ungrouped data</th>
<th>( t_n &gt; \frac{2}{n} \sum_{i=1}^n t_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yamada S-Shaped Model</td>
<td>( n_k \ell_k &gt; \sum_{i=1}^k y_i \left[ \ell_i + \frac{\ell_{i-1}}{\ell_{i-1} + \ell_i} \right] )</td>
<td>( t_n &gt; \frac{3}{2n} \sum_{i=1}^n t_i )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Software failure conditions for the Goel-Okumoto and Yamada S-Shaped model

In Table 2 we state the conditions for the Inflection S-Shaped model. We assume one of the parameters, \( \psi \), to be known. Recall that these conditions are sufficient for the existence of at least one, positive and finite solution of the Maximum Likelihood equations.

| \( \psi < 1 \) | Grouped data | \( \ell_k n_k > \sum_{i=1}^k y_i (\ell_i + \ell_{i-1}) \) | Ungrouped data | \( t_n > \frac{2}{n} \sum_{i=1}^n t_i \) |
| \( \psi > 1 \) | \( \ell_k n_k < \sum_{i=1}^k y_i (\ell_i + \ell_{i-1}) \) | \( t_n < \frac{2}{n} \sum_{i=1}^n t_i \) |

Table 2: Software failure conditions for the Inflection S-Shaped model

In case of the Goel-Okumoto Model and the Yamada S-Shaped Model we study the relation between reliability growth and whether the Maximum Likelihood equation has a root. The results can be found in Table 3. Note that “Root \( \rightarrow \) Reliability Growth” only states that it is reasonable to assume reliability growth, but that it may not be the case.

<table>
<thead>
<tr>
<th>Goel-Okumoto</th>
<th>Grouped data</th>
<th>No Root ( \leftrightarrow ) No Reliability Growth</th>
<th>Root ( \leftrightarrow ) Reliability Growth</th>
<th>Ungrouped data</th>
<th>No Root ( \leftrightarrow ) No Reliability Growth</th>
<th>Root ( \rightarrow ) Reliability Growth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yamada S-Shaped</td>
<td>No Root ( \rightarrow ) No Reliability Growth</td>
<td>Root ( \leftrightarrow ) Reliability Growth</td>
<td>No Root ( \rightarrow ) No Reliability Growth</td>
<td>Root ( \leftrightarrow ) Reliability Growth</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Relation between Maximum Likelihood equation and reliability growth
The second part of the report introduces a recursive estimation method which allows us to estimate the unknown parameters in real time by using only the last few software failure observations. Often, this algorithm is able to return estimates when the Maximum Likelihood Method seems unstable or when the software failure data does not satisfy the right conditions for the Maximum Likelihood Method to have a solution. The recursive method can be used to estimate each parameter separately or all parameters simultaneously. The benefit of separate estimation is that it is less computationally demanding. In case of grouped data we are able to calculate the recursive formulas for the parameters of the Goel-Okumoto model and the Yamada S-Shaped model. In case of ungrouped data, we are not. Because of their complexity, we refer to Chapter 4 for the recursive formulas.
Preface

This report is the result of my graduation from the Master program Industrial and Applied Mathematics at the Eindhoven University of Technology. I would like to thank my supervisors Alessandro Di Bucchianico and Eduard Belitser, both from TU/e. I would like to thank Alessandro for arranging an office at the university for me to work on this project, his help with Mathematica, LaTeX and the chapters concerning the Maximum Likelihood Method. I would like to thank Eduard for his help on the chapters concerning the recursive estimation method. With their enthusiasm and willingness to help, they both made this final step to my masters degree a very pleasant experience. Finally, I would like to thank Marko Boon, also from TU/e, for checking the report thoroughly, resulting in many useful notes which have helped to improve the report.

P.H.A. Meyfroyt
Eindhoven, 20-7-2012
# Contents

1 Introduction .................................................. 6
   1.1 Problem Introduction .................................... 6
   1.2 Overview Report ........................................ 7

2 Software Reliability Models ................................. 8
   2.1 Assumptions .............................................. 8
   2.2 General Model ........................................... 10
   2.3 Exponential Models ....................................... 11
      2.3.1 Goel-Okumoto Model ................................. 11
   2.4 S-Shaped Models ......................................... 12
      2.4.1 Yamada S-Shaped Model .............................. 12
      2.4.2 Inflection S-Shaped Model ......................... 13
   2.5 Data ..................................................... 14

3 Maximum Likelihood Method ................................ 15
   3.1 General Likelihood Procedures ......................... 15
      3.1.1 Grouped Data ....................................... 16
      3.1.2 Ungrouped Data ..................................... 19
   3.2 Trend Test ............................................... 21
   3.3 Goel-Okumoto Model ..................................... 22
      3.3.1 Grouped Data ....................................... 22
      3.3.2 Ungrouped Data ..................................... 25
   3.4 Yamada S-Shaped Model .................................. 26
      3.4.1 Grouped Data ....................................... 27
      3.4.2 Ungrouped Data ..................................... 33
   3.5 Inflection S-Shaped Model ............................... 36
      3.5.1 Grouped Data ....................................... 36
      3.5.2 Ungrouped Data ..................................... 38

4 Recursive Method ............................................. 41
   4.1 General Recursive Method ............................... 41
      4.1.1 Separate Estimation ................................. 42
      4.1.2 Simultaneous Estimation ............................ 45
   4.2 Goel-Okumoto Model ..................................... 47
      4.2.1 Separate Estimation ................................. 47
      4.2.2 Simultaneous Estimation ............................ 47
4.3 Yamada S-Shaped Model .............................................. 48
   4.3.1 Separate Estimation ........................................... 48
   4.3.2 Simultaneous Estimation ..................................... 48
4.4 Examples ............................................................... 49
   4.4.1 Goel-Okumoto Model ........................................... 50
   4.4.2 Yamada S-Shaped Model ....................................... 52
4.5 Ungrouped Data ....................................................... 54
   4.5.1 Separate Estimation ........................................... 54
   4.5.2 Simultaneous Estimation ..................................... 55
5 Conclusions ............................................................... 56
   5.1 Maximum Likelihood Method .................................... 56
   5.2 Recursive Method ................................................ 57
6 Recommendations ....................................................... 60
   6.1 Maximum Likelihood Method .................................... 60
   6.2 Recursive Method ................................................ 61
Glossary ................................................................. 63
Chapter 1

Introduction

The amount of software-based consumer products and the lines of code that software contains, has grown enormously in the last few decades. This development makes it more difficult for software developers to meet the challenging requirements of meeting market windows and providing nearly fault-free software. The requirement to meet market windows may cause the software to be released while containing errors, which may incur high failure costs. On the other hand, extensive testing and debugging increases development costs and may cause the software to be outdated on the day of release. These conflicting goals, minimizing both development time and failure costs, make quality control one of the key engineering technologies in today’s industry. Quality control can provide software developers with a clear picture of the relation between time spent on testing and reliability. We define reliability to be the probability of failure-free operation for a specified period of time in a specified environment (Musa et al. (1987)). A commonly accepted metric for quantifying a products reliability is the number of faults you expect to find within a certain period of time. The mathematical relation that exists between time spent on testing software and the cumulative number of errors discovered, is referred to as a software reliability model. Several software reliability models have been proposed which have already proven to be successful tools. With the application of these models rises the problem of the estimation of unknown parameters.

1.1 Problem Introduction

The first well-known model to study software reliability was made by Jelinski and Moranda (Jelinski and Moranda (1972)) and allows us to estimate reliability by using the software failure data. The models we study in this report also rely on software failure data and the mathematical framework used to establish these models, is identical to that of the Jelinski-Moranda model. The models we use are the Goel-Okumoto model, the Yamada S-Shaped model and the Inflection S-Shaped model. To fit these models to the software failure data, several parameters have to be estimated. A popular method to estimate these parameters is the Maximum Likelihood Method. In this method, parameters can be estimated by solving the Maximum Likelihood equations. Unfortunately, these equations do not always have a solution. In the first part of the report we study this issue and formulate the problem as follows:

What are the conditions software failure data has to satisfy, in order for the Maximum Likelihood Method to have a solution?
We derive the conditions for each model which, if satisfied, assure a unique solution of the Maximum Likelihood equations. When the conditions for the software failure data are not satisfied, or when the Maximum Likelihood Method seems unstable, we would still like to be able to estimate the model parameters. This is why in the second part of the report we present another method to estimate the unknown parameters. This method is a recursive estimation algorithm and is essentially a recursive application of the Maximum Likelihood Method. Often, this algorithm is able to provide an estimate in situations when the Maximum Likelihood Method fails. We would like to point out that the proposed recursive method is, at the moment, still a work in progress and we are not able to apply the method to all types of data or software reliability models.

Note that the goal of the report is to establish conditions on the software failure data and to propose a relatively new recursive method. We do not go into detail about how to apply the Maximum Likelihood Method or the recursive method in practice. When using these algorithms in practice, the assumptions on which these methods are based, may not apply. The effect of assumptions not being applicable to the situation of the reader lies out of the scope of this report.

1.2 Overview Report

In Chapter 2 we give a description of the modelling of software reliability growth. This chapter starts with the assumptions which form the basis of our model (Section 2.1). After the assumptions we give a description of how the software failure process is modelled (Section 2.2). After introducing the general model we give descriptions of exponential models (Section 2.3.1) and S-shaped models (Section 2.4). Chapter 2 concludes with a description of the types of data we use throughout the report (Section 2.5). In Chapter 3 we study the Maximum Likelihood Method. First we describe general likelihood procedures (Section 3.1). After introducing the Maximum Likelihood Method we discuss the connection between the existence of a solution of the Maximum Likelihood equations and trend tests (Section 3.2). Then we apply the Maximum Likelihood Method to the Goel-Okumoto model, Yamada S-Shaped model and Inflection S-Shaped model in Sections 3.3, 3.4 and 3.5, respectively. In Chapter 4 we introduce a recursive estimation algorithm. First we give a general description of the method in Section 4.1. In this section we also make a distinction between separate estimation and simultaneous estimation of the parameters. After the general description we apply the method to both the Goel-Okumoto Model (Section 4.2) and the Yamada S-Shaped model (Section 4.3) for grouped data. Section 4.4 contains plots of the estimation process of the recursive algorithm. In these graphs we also plot the estimation process of the Maximum Likelihood Method for comparison. In Section 4.5 we give a brief description of the application of the recursive method to ungrouped data. In Chapter 5 we draw conclusions from the results we gathered during our research. In Chapter 6 we state several suggestions for further research. After Chapter 6 we included a glossary which defines all the notations we use throughout the report.
Chapter 2

Software Reliability Models

In this chapter we model the software failure process. In Section 2.1 we first state the assumptions which form the framework in which we model the software failure process. In Section 2.2 we derive the general mathematical model of the process. After explaining the general model we introduce three specific models which we use throughout the report. We first introduce an exponential model, the Goel-Okumoto Model, in Section 2.3.1. Then we introduce two S-shaped models in Section 2.4. After an introduction about S-Shaped models we study two of these models, the Yamada S-Shaped Model and the Inflection S-Shaped Model. These models are the subject of Subsection 2.4.1 and 2.4.2, respectively. Note that Sections 2.3.1 and 2.4 are adapted from Pham (2006). Finally, in Section 2.5, we give a description of two types of data that we consider in this report.

2.1 Assumptions

First we state the assumptions which form the framework in which we develop the model of the software failure process. All assumptions are taken from Goel (1985). When reading these assumptions, the reader should keep in mind that software and the testing of software is highly dependent on the environment. This leads to the fact that not all assumptions may be applicable in the situation of interest to the reader. This is why before applying a model, the reader should take care in determining whether the model, including its underlying assumptions, is appropriate for his or her situation. For assumptions which apply to a specific popular model, we recommend Kharchenko et al. (2002).

A Detected Fault Is Immediately Corrected.
We assume that, when a failure occurs, the fault causing the failure is removed from the test path. This means that a fault can either be physically removed from the code or the fault has no further effect on the remainder of the test.

The Total Number Of Faults In The Program Is Finite
Because a software program can have an infinite number of states, there exists a possibility that the software generates an infinite amount of errors. For the physical process that is being modelled this assumption is obviously true. From a mathematical point of view this assumption is an important restriction when developing the model. Because of the importance we like to state it here explicitly.
No Fault Occurs At The Start Of The Test
Again, this is an assumption which may seem obvious but is useful from a mathematical point of view. When we model the number of faults discovered at time \( t \) as \( N(t) \), this implies \( N(0) = 0 \).

No New Faults Are Introduced During The Fault Removal Process
In practice this may not always be the case. When faults are discovered and corrected, this may have a negative effect on the remainder of the test. If these newly introduced faults are just a small fraction of the total amount of faults in the program, this effect will have little impact on the model.

Failure Rate Decreases With Test Time
Like stated before, when a fault is discovered it is either physically removed from the code or it has no further effect on the remaining test cases. If the fault is physically removed, the failure rate decreases explicitly. If the fault has no further effect, the failure rate decreases implicitly since a smaller portion of the code is subject to testing.

Failure Rate Is Proportional To The Number Of Remaining Faults
In other words, this assumption states that all faults have an equal probability of being detected between failures. However, this may not always be the case. If a crucial fault is discovered which may have effect on the remainder of the test path, this path may be tested more thoroughly and faults in this specific path are more likely to be discovered.

Reliability Is A Function Of The Number Of Remaining Faults
When developing the model, we assume that during the use of the program, all remaining faults have an equal probability to occur. However, the usage of the program may not be uniform. If some portions of the program are more likely to be executed, faults in these portions are more likely to occur. If information is available about the usage of the program, this can be incorporated in the analysis of the reliability. Otherwise, uniform usage is the only reasonable assumption.

Test Effort Is Used As A Basis For Failure Rate
Throughout this report we assume failure rate is proportional to test effort. If we would assume failure rate is proportional to calendar time, results could be deceiving because it neglects the intensity of the testing process. Test effort can, for example, be measured in lines of code tested, number of functions tested or number of test cases executed.

Testing is Representative of the Operational Usage
Because we are interested in the reliability of the software during operational usage, the reliability is determined with respect to the way the user is expected to operate the software. This implies that during testing, functions are tested with the same proportion as they are operated by the user. For this assumption to hold, it is important that information about operational usage is incorporated in the test.
2.2 General Model

In this section we establish the general model of the software failure process. We model the number of observed failures up to time $t$ as a pure birth counting process $(N(t))_{t \geq 0}$, or more specifically, a non-homogeneous Poisson process. Therefore we start this section with the definition of a pure birth counting process and the properties it has to satisfy to be a non-homogeneous Poisson process. This definition and the properties are taken from Ross (2007) and Feller (1968), respectively.

**Definition 2.1** A stochastic process $(N(t))_{t \geq 0}$ is said to be a pure birth counting process if the following conditions hold:

1) $N(0) = 0$

2) $N(t)$ is integer valued

3) If $s < t$, then $N(s) \leq N(t)$

4) For $s < t$, $N(t) - N(s)$ equals the number of events that occur in the interval $(s, t]$

The pure birth counting process possesses independent increments if the number of events that occur in disjoint time intervals are independent. The increments are called stationary increments if the distribution of the number of events that occur in any time interval depends only on the length of the interval.

**Definition 2.2** A pure birth counting process $(N(t))_{t \geq 0}$ is a non-homogeneous Poisson process (NHPP) with intensity function $\lambda(t)$, for all $t \geq 0$, if it satisfies the following properties:

1) $N(0) = 0$

2) $(N(t))_{t \geq 0}$ has independent increments. This implies that for any $t_i < t_j < t_k < t_\ell$ the random variables $N(t_j) - N(t_i)$ and $N(t_\ell) - N(t_k)$ are independent.

3) The random variable $N(t_j) - N(t_i)$ has a Poisson distribution with mean $\Lambda(t_j) - \Lambda(t_i)$, for all $0 \leq t_i < t_j$. This implies that

$$
\mathbb{P}[N(t_j) - N(t_i) = k] = e^{-(\Lambda(t_j) - \Lambda(t_i))} \frac{(\Lambda(t_j) - \Lambda(t_i))^k}{k!}
$$

for all $k = 0, 1, \ldots$, where $\Lambda(t) = \int_0^t \lambda(x) \, dx$ is the mean value function of the non-homogeneous Poisson process $[N(t), t \geq 0]$.

Note that the times between events of a NHPP are neither independent nor identically distributed. Only the increments are independent although not identically distributed. We model the number of failures up to time $t$ as a pure birth counting process $(N(t))_{t \geq 0}$ which follows a non-homogeneous Poisson distribution. This implies that the number of failures which occur during two different time intervals are independent. This is a property which we use extensively throughout the report. The quantity $\Lambda(t)$ describes the expected number of failures up to time $t$. Because of the underlying assumptions about the failures and number of faults in the software, we assume $\Lambda(t)$ to be a bounded, strictly increasing function satisfying the boundary conditions $\Lambda(0) = 0$ and $\lim_{t \to \infty} \Lambda(t) = E[N]$, where $N = \lim_{t \to \infty} N(t)$.
2.3 Exponential Models

Over the years, several exponential models, based on different assumptions, have been proposed. The Musa Exponential Growth Model proposed in Musa and Okumoto (1984) incorporates the relation between execution time and calendar time. The Hyperexponential Growth Model (Ohba (1984)) and the Yamada-Osaki Exponential Growth Model (Yamada and Osaki (1985)) both incorporate the existence of different modules in the software. We recommend Pham (2006) for a good overview of all assumptions. The exponential model we consider in this report is the Goel-Okumoto Model.

2.3.1 Goel-Okumoto Model

This model was first introduced in Goel and Okumoto (1979) and is based on the following assumptions:

1) All faults in the software are mutually independent from the failure detection point of view.

2) The number of failures detected at any time is proportional to the current number of faults in the software. This means that the probability of the failures or faults actually occurring, i.e., being detected, is constant.

3) The isolated faults are removed prior to future test occasions.

4) Each time a software failure occurs, the software error which caused it is immediately removed and no new errors are introduced.

These assumptions lead to the following differential equation

\[
\frac{\partial}{\partial t} \Lambda(t) = b(a - \Lambda(t)).
\]  

(2.1)

In this equation, \( a \) represents the expected total number of faults in the software before testing. Parameter \( b \) stands for the failure detection rate or the failure intensity of a fault and \( \Lambda(t) \) for the expected number of failures detected at time \( t \). Solving (2.1) for \( \Lambda(t) \), we obtain the following mean value function

\[
\Lambda(t) = a(1 - e^{-bt}).
\]  

(2.2)

The corresponding intensity function is

\[
\lambda(t) = ab e^{-bt}.
\]  

(2.3)

The remaining number of faults at time \( t \) equals

\[
\Lambda(\infty) - \Lambda(t) = ae^{-bt}.
\]  

(2.4)
2.4 S-Shaped Models

In case of an S-Shaped model, the software reliability growth curve resembles an S-shape. This is a result of the error detection rate reaching a maximum during the test, after which it decreases exponentially. This can be explained by stating that some faults are covered by other faults at the beginning of the test phase, and before these faults are removed, the covered faults remain undetected. The shape can also be explained by stating that the software testing process usually involves a learning process were testers become familiar with the software and their testing skills gradually improve. See Ohba (1984) for a more elaborate justification of this particular shape. S-Shaped models are based on the following assumptions:

1) The error detection rate differs among faults.

2) Each time a software failure occurs the software error which caused it is immediately removed and no new errors are introduced.

These assumptions lead to the following differential equation

\[
\frac{\partial}{\partial t} \Lambda(t) = b(t)(a - \Lambda(t)).
\]  

(2.5)

In this equation, \(a\) represents the expected total number of faults in the software before testing. Parameter \(b\) stands for the failure detection rate or the failure intensity of a fault and \(\Lambda(t)\) for the expected number of failures detected at time \(t\). Solving (2.5) for \(\Lambda(t)\), we obtain the following mean value function

\[
\Lambda(t) = a\left(1 - e^{-\int_0^t b(u)du}\right).
\]  

(2.6)

The corresponding intensity function is

\[
\lambda(t) = ab(t)e^{-\int_0^t b(u)du}.
\]  

(2.7)

2.4.1 Yamada S-Shaped Model

The Yamada S-Shaped model was first introduced in Yamada et al. (1983). The model is based on the following assumptions:

1) All faults in the software are mutually independent from the failure detection point of view.

2) The probability of failure detection at any time is proportional to the current number of faults in the software.

3) The proportionality of failure detection is constant.

4) The initial error content of the software is a random variable.

5) A software system is subject to failures at random times caused by errors present in the system.

6) The time between the \((i - 1)\)th and the \(i\)th failure, depends on the time of the \((i - 1)\)th failure.
7) Each time a failure occurs, the error which caused it, is immediately removed and no other errors are introduced.

The Yamada S-Shaped Model is defined by taking

\[ b(t) = \frac{b^2 t}{1 + bt}, \]  

(2.8)

where the parameter \( b \) represent the failure-detection rate. Substituting (2.8) into (2.6) gives

\[ \Lambda(t) = a(1 - (1 + bt)e^{-bt}). \]

The intensity function is

\[ \lambda(t) = ab^2te^{-bt}. \]

The remaining number of faults at time \( t \) equals

\[ \Lambda(\infty) - \Lambda(t) = a(1 + bt)e^{-bt}. \]  

(2.9)

### 2.4.2 Inflection S-Shaped Model

The Inflection S-shaped model, see also Ohba (1984), is based on the following assumptions:

1) Some faults are not detectable before some other faults are removed.

2) The probability of failure detection at any time is proportional to the current number of detectable faults in the software.

3) Failure rate of each detectable fault is constant and identical.

4) The isolated faults can be entirely removed.

The Inflection S-Shaped Model is defined by taking

\[ b(t) = \frac{b}{1 + \beta e^{-bt}}, \]  

(2.10)

where the parameters \( b \) and \( \beta \) represent the failure-detection rate and the inflection factor, respectively. The inflection factor is defined as follows: \( \beta = \frac{1 - r}{r} \). Here, \( r \) represents the proportion of independent errors present in the software. Note that if all errors are independent, which implies \( r = 1 \), the Inflection S-Shaped Model equals the Goel-Okumoto Model.

Substituting (2.10) into (2.6), we obtain

\[ \Lambda(t) = a\frac{1 - e^{-bt}}{1 + \beta e^{-bt}}. \]

The corresponding intensity function is

\[ \lambda(t) = ab\frac{(1 + \beta)e^{-bt}}{(1 + \beta e^{-bt})^2}. \]

The remaining number of faults at time \( t \) equals

\[ \Lambda(\infty) - \Lambda(t) = a\frac{(1 + \beta)e^{-bt}}{1 + \beta e^{-bt}}. \]  

(2.11)
2.5 Data

In this report we consider two different forms in which software failure data can be presented, grouped and ungrouped data. In case of ungrouped data, the data consists of the exact times that a failure occurs. The point in time failure $i$ occurs, is in this report denoted by $t_i$. Ungrouped data is also called exact data or point-time data. In this case, the data can be presented by either the failure times or the time between failures. Ungrouped data can either be time truncated or failure truncated. In case of time truncated data, the software is tested up to a predetermined point in time $t$. If we define $T_n$ to be the random variable representing the last observed failure time, this implies $T_n < t$. When the data is failure truncated, the software is tested up to a predetermined number of failures, say $n$. This implies that the test ends at $t = T_n$. In this report, when we apply our methods to ungrouped data, we use failure truncated data. When the failure process is modelled using ungrouped data, the time between failures is treated as a random variable and the time between failures is expected to increase as faults are being removed from the software. When, for example, exact failure times cannot be observed, data can be presented as grouped data. Grouped data consists of failure counts in predetermined time intervals which do not have to be of the same length. This is why this type of data is also called interval data or failure count data. The amount of failures occurring in interval $i$, is in this report denoted by $y_i$. Note that, because the time intervals are specified beforehand, the bounds of the intervals do not correspond to failure times and the data is always time truncated. This implies that the intervals are not random variables but given by predetermined end points in $(0, \infty)$. The number of failures in each interval is treated as a random variable and the number of failures per unit of time is expected to decrease as faults are being removed from the software.

In Table 2.1 we give an example of presenting failure time data both in grouped and ungrouped form. The predetermined time intervals used to create the grouped data have an equal length of one hour, starting at 9:00. The ungrouped data is denoted in number of minutes.

<table>
<thead>
<tr>
<th>$i$</th>
<th>Failure Time</th>
<th>Grouped Data ($y_i$)</th>
<th>Ungrouped Data ($t_i$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9:00</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>9:34</td>
<td>2</td>
<td>34</td>
</tr>
<tr>
<td>3</td>
<td>9:59</td>
<td>1</td>
<td>59</td>
</tr>
<tr>
<td>4</td>
<td>10:18</td>
<td>0</td>
<td>78</td>
</tr>
<tr>
<td>5</td>
<td>12:22</td>
<td>2</td>
<td>202</td>
</tr>
<tr>
<td>6</td>
<td>12:27</td>
<td>0</td>
<td>207</td>
</tr>
<tr>
<td>7</td>
<td>15:44</td>
<td>0</td>
<td>404</td>
</tr>
<tr>
<td>8</td>
<td>15:59</td>
<td>2</td>
<td>419</td>
</tr>
<tr>
<td>9</td>
<td>16:32</td>
<td>1</td>
<td>452</td>
</tr>
<tr>
<td>10</td>
<td>17:57</td>
<td>1</td>
<td>537</td>
</tr>
</tbody>
</table>

Table 2.1: Example Grouped and Ungrouped Data
Chapter 3

Maximum Likelihood Method

In this chapter we determine the parameters of software reliability models by using the Maximum Likelihood Method. In Section 3.1 we give a general description of this method. In Subsections 3.1.1 and 3.1.2 we give a general description of how to apply the method to the grouped data case and the ungrouped data case, respectively. In Section 3.2 we give a short description of the Laplace trend test which we use to establish a condition which implies a trend in the data. We then compare this condition with the condition which guarantees a solution of the Maximum Likelihood equations. Then we apply the Maximum Likelihood Method to the Goel-Okumoto Model (Section 3.3), Yamada S-Shaped Model (Section 3.4) and Inflection S-Shaped Model (Section 3.5). We especially go into detail about the conditions the software failure data has to satisfy for this method to work and the comparison with the conditions from Section 3.2. The conditions and the corresponding proofs for the Goel-Okumoto Model and Yamada S-Shaped Model are largely due to Knafl (1992), Knafl and Morgan (1996) and Hossain and Dahiya (1993).

3.1 General Likelihood Procedures

Assume that $X_1, X_2, \ldots, X_n$ form a random sample from a discrete or continuous distribution for which the probability function or the probability distribution function is denoted by $f(x|\theta)$. Here, $\theta$ is a real valued vector, belonging to some space $\Omega$, consisting of one or more parameters. For an observed vector $x = (x_1, \ldots, x_n)$, the joint probability function or joint probability distribution function $f_n(x|\theta)$ is regarded as a function of $\theta$ and is called the likelihood function. A reasonable estimate of $\theta$ would be any vector $\theta \in \Omega$ which maximises the probability that we would observe $x = (x_1, \ldots, x_n)$ and thus maximises the likelihood function. The vector $\hat{\theta}$ that maximises the likelihood function for the observed vector $x$ is called the Maximum Likelihood estimator and is regarded as the best estimator for the real parameter(s) of $f(x|\theta)$. Most of the times, it is difficult to maximise the likelihood function and the maximisation can be simplified by maximising the logarithm of the likelihood function. This is allowed because the value for $\hat{\theta}$ which maximises the logarithm of the likelihood function, also maximises the likelihood function. The logarithm of the likelihood function is called the log-likelihood function. In order for the underlying failure model to be identifiable, we assume throughout this report that the number of data points used in the estimation is at least the number of unknown parameters which are being estimated.
3.1.1 Grouped Data

In case of grouped data, if we want to use the Maximum Likelihood Method to estimate the model parameters, we first have to derive the log-likelihood function of observing the failure counts \(y_i\) in predetermined time intervals \([\ell_{i-1}, \ell_i)\) for \(i = 1, \ldots, k\). In Theorem 3.1 we state the log-likelihood function followed by a proof of the statement.

**Theorem 3.1** Let \(0 = \ell_0 < \ell_1 < \cdots < \ell_k\) be fixed. Then the log-likelihood function of a non-homogeneous Poisson process with mean value function \(\Lambda(t)\) and \(y_i\) events in the interval \([\ell_{i-1}, \ell_i)\), is given by

\[
L = \sum_{i=1}^{k} [y_i \ln (\mu (\ell_i)) - \ln (y_i!)] - \Lambda (\ell_k),
\]

where \(\mu (\ell_i) = \Lambda (\ell_i) - \Lambda (\ell_{i-1})\).

**Proof:** To establish the log-likelihood function, we first derive the joint probability function of the observed data set. Define \(N_i = N(\ell_i)\), where \(N_0 = N(\ell_0) = 0\). By Property 3 of Definition 2.2, we have for \(i = 1, \ldots, k\)

\[
P (N_i - N_{i-1} = n_i - n_{i-1}) = \frac{(\Lambda (\ell_i) - \Lambda (\ell_{i-1}))^{n_i - n_{i-1}}}{(n_i - n_{i-1})!} e^{-(\Lambda (\ell_i) - \Lambda (\ell_{i-1}))}.
\]

If we use the following notations: \(Y_i = N_i - N_{i-1}\) and \(y_i = n_i - n_{i-1}\), we can transform (3.2) into

\[
P (Y_i = y_i) = \frac{(\Lambda (\ell_i) - \Lambda (\ell_{i-1}))^{y_i}}{y_i!} e^{-(\Lambda (\ell_i) - \Lambda (\ell_{i-1}))}.
\]

Using Property 2 of Definition 2.2 and \(\mu (\ell_i) = \Lambda (\ell_i) - \Lambda (\ell_{i-1})\), we can write

\[
P (Y_1 = y_1, \ldots, Y_k = y_k) = \prod_{i=1}^{k} \frac{(\Lambda (\ell_i) - \Lambda (\ell_{i-1}))^{y_i}}{(y_i)!} e^{-(\Lambda (\ell_i) - \Lambda (\ell_{i-1}))} = \prod_{i=1}^{k} \frac{(\mu (\ell_i))^{y_i}}{y_i!} e^{-\mu (\ell_i)}.
\]

Because observing the increments \(N_1, N_2 - N_1, \ldots, N_k - N_{k-1}\) equals observing the cumulative failure counts \(N_1, N_2, \ldots, N_k\), it follows that

\[
P (Y_1 = y_1, \ldots, Y_k = y_k) = P (N_1 = n_1, N_2 = n_2, \ldots, N_k = n_k).
\]

Thus we can write

\[
P (N_1 = n_1, \ldots, N_k = n_k) = \prod_{i=1}^{k} \frac{(\mu (\ell_i))^{y_i}}{y_i!} e^{-\mu (\ell_i)}.
\]

Expression (3.3) is the joint probability function of the observed data and thus the likelihood function. To calculate the log-likelihood function we take the logarithm of (3.3) which results in (3.1). Note that \(\Lambda (\ell_k)\) in (3.1) is the result of the telescoping sum \(\sum_{i=1}^{k} \mu (\ell_i) = \Lambda (\ell_k) - \Lambda (\ell_0) = \Lambda (\ell_k)\).\qed
All mean value functions that we consider in this report are of the form $a\bar{\Lambda}(t)$. Here, $a = \lim_{t \to \infty} \Lambda(t)$ and $\Lambda(t)$ is the mean value function $\Lambda(t)$ with parameter $a$ factored out. Because $\bar{\Lambda}(t)$ is a strictly increasing function and $t > 0$, we know that $0 < \bar{\Lambda}(t) < 1$. Recalling that $\mu(\ell_i) = \Lambda(\ell_i) - \Lambda(\ell_{i-1})$, we can rewrite (3.1) as

$$L = \sum_{i=1}^{k} \left[ y_i \ln(a) + y_i \ln(\bar{\Lambda}(\ell_i) - \bar{\Lambda}(\ell_{i-1})) - \ln(y_i!) \right] - a\bar{\Lambda}(\ell_k).$$  \hspace{1cm} (3.4)

Estimating a parameter with this method implies maximising the log-likelihood function (3.4) with respect to the parameter of interest. Maximising the log-likelihood function can be done by calculating its derivative and finding the root with respect to the desired parameter. In some situations, the derivative of the log-likelihood function may have several or no roots in which the method may lead to wrong or no parameter estimations. This is why, in Sections 3.3, 3.4 and 3.5 we give conditions for the Goel-Okumoto model, Yamada S-Shaped model and Inflection S-Shaped model respectively, which the observed software failure data has to satisfy for the derivative of the log-likelihood function to have a unique, positive root. In case of two-parameter models, differentiating (3.4) and setting the expressions equal to zero leads to the following set of equations:

$$\frac{\partial L}{\partial a} = \sum_{i=1}^{k} \left[ \frac{y_i}{a} \right] - \bar{\Lambda}(\ell_k) = 0$$  \hspace{1cm} (3.5)

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{k} y_i \left[ \frac{\partial}{\partial b} \left( \frac{\bar{\Lambda}(\ell_i) - \bar{\Lambda}(\ell_{i-1})}{\Lambda(\ell_i) - \Lambda(\ell_{i-1})} \right) \right] - a \frac{\partial}{\partial b} \bar{\Lambda}(\ell_k) = 0$$  \hspace{1cm} (3.6)

Notice that we can explicitly solve (3.5) with respect to $a$ by writing:

$$a = \frac{n_k}{\Lambda(\ell_k)}. \hspace{1cm} (3.7)$$

If we insert this expression for $a$ into (3.6) we get

$$\sum_{i=1}^{k} y_i \left[ \frac{\partial}{\partial b} \left( \frac{\bar{\Lambda}(\ell_i) - \bar{\Lambda}(\ell_{i-1})}{\Lambda(\ell_i) - \Lambda(\ell_{i-1})} \right) \right] - n_k \frac{\partial}{\partial b} \bar{\Lambda}(\ell_k) = 0. \hspace{1cm} (3.8)$$

Note that because $0 < \bar{\Lambda}(t) < 1$, our estimate of $a$ is never less than the number of faults that are detected. Also, we do not need to worry about a singularity in the denominator of our estimate for $a$. Thus, if we have an estimate of $b$, we also have an estimate of $a$. This is why in the remainder of the report, we focus on the estimation of $b$. To estimate $b$, we have to solve (3.8). For the models used in this report, this equation has to be solved by computer and we do not go into detail about how to find the root. We do go into detail about which conditions the software failure data has to satisfy for (3.8) to have a solution. These conditions are due to Knafl (1992) and Knafl and Morgan (1996) and rely on the fact that (3.8) is a monotonically decreasing function. When we derive these conditions, the left-hand side of (3.8) is used extensively and we refer to it as $\Phi(b)$. To simplify the analysis of $\Phi(b)$ we transform it into a single sum using $n_k = \sum_{i=1}^{k} y_i$. Then the expression looks as follows

$$\Phi(b) = \sum_{i=1}^{k} y_i \left[ \frac{\partial}{\partial b} \left( \frac{\bar{\Lambda}(\ell_i) - \bar{\Lambda}(\ell_{i-1})}{\Lambda(\ell_i) - \Lambda(\ell_{i-1})} \right) - \frac{\partial}{\partial b} \frac{\bar{\Lambda}(\ell_k)}{\Lambda(\ell_k)} \right]. \hspace{1cm} (3.9)$$
In case of estimating the parameters for the Inflection S-Shaped model we use $\Phi(\psi)$ to estimate the parameter $\psi$. Analogously to (3.9), we can write
\[
\Phi(\psi) = \sum_{i=1}^{k} y_i \left[ \frac{\partial}{\partial \psi} \left( \frac{\bar{\Lambda}(t_i) - \bar{\Lambda}(t_{i-1})}{\Lambda(t_i) - \Lambda(t_{i-1})} \right) - \frac{\partial}{\partial \psi} \frac{\bar{\Lambda}(t_k)}{\Lambda(t_k)} \right]. \tag{3.10}
\]

During this research we found another way of transforming $\Phi(b)$ and $\Phi(\psi)$ into a single sum. We did not end up using it because the expression was not useful for the estimation of the parameters. The theory does resemble the paper Zhao and Xie (1996) in which the asymptotic of Maximum Likelihood estimators of software failure models are studied. To obtain a single sum, we add a virtual data point at infinity and thus rewrite $\Phi(b)$ by using:
\[
y_{k+1} = a - n_k = \frac{n_k}{\Lambda(t_k)} - n_k = n_k \left( \frac{1 - \bar{\Lambda}(t_k)}{\Lambda(t_k)} \right) \tag{3.11}
\]
\[
\bar{\Lambda}(t_{k+1}) = \lim_{t \to \infty} \bar{\Lambda}(t) = 1
\]
\[
\frac{\partial}{\partial b} \bar{\Lambda}(t_{k+1}) = \lim_{t \to \infty} \frac{\partial}{\partial b} \bar{\Lambda}(t) = 0
\]

Note that the last limit is a property of the models used in this report and may not be true for other models. The term that is being added to the sum in (3.8) can be rewritten as follows
\[
-n_k \frac{\partial}{\partial b} \frac{\bar{\Lambda}(t_k)}{\Lambda(t_k)} = n_k \left( \frac{1 - \bar{\Lambda}(t_k)}{\Lambda(t_k)} \right) \left( \frac{0 - \partial}{\partial b} \frac{\bar{\Lambda}(t_k)}{\Lambda(t_k)} \right) = y_{k+1} \frac{\partial}{\partial b} \frac{\bar{\Lambda}(t_{k+1}) - \bar{\Lambda}(t_k)}{\Lambda(t_{k+1}) - \Lambda(t_k)}.
\]

Thus we can rewrite $\Phi(b)$ to
\[
\Phi(b) = \sum_{i=1}^{k+1} y_i \left[ \frac{\partial}{\partial b} \left( \frac{\bar{\Lambda}(t_i) - \bar{\Lambda}(t_{i-1})}{\Lambda(t_i) - \Lambda(t_{i-1})} \right) - \frac{\partial}{\partial b} \frac{\bar{\Lambda}(t_k)}{\Lambda(t_k)} \right]. \tag{3.12}
\]

All equations mentioned in this subsection also apply to the Inflection S-Shaped model. Because this model has one extra parameter, $\psi$, we also need one extra equation. Looking at the derivation of (3.9), it is obvious we can write down a similar expression for $\psi$:
\[
\Phi(\psi) = \sum_{i=1}^{k} y_i \left[ \frac{\partial}{\partial \psi} \left( \frac{\bar{\Lambda}(t_i) - \bar{\Lambda}(t_{i-1})}{\Lambda(t_i) - \Lambda(t_{i-1})} \right) - \frac{\partial}{\partial \psi} \frac{\bar{\Lambda}(t_k)}{\Lambda(t_k)} \right]. \tag{3.13}
\]

It is not obvious whether we can write (3.12) in the same form as (3.11). For this we first have to calculate $\lim_{t \to \infty} \frac{\partial}{\partial \psi} \Lambda(t)$ and see whether it equals zero just like $\lim_{t \to \infty} \frac{\partial}{\partial \psi} \Lambda(t)$. If we take $\Lambda(t)$ to be the mean value function of the Inflection S-Shaped model, we see
\[
\lim_{t \to \infty} \frac{\partial}{\partial \psi} \Lambda(t) = \lim_{t \to \infty} \frac{e^{bt} - 1}{(e^{bt} + \psi)^2} = 0.
\]

This implies we can use the method of adding a virtual data point at infinity and thus rewrite (3.12) into
\[
\Phi(\psi) = \sum_{i=1}^{k+1} y_i \left[ \frac{\partial}{\partial \psi} \left( \frac{\bar{\Lambda}(t_i) - \bar{\Lambda}(t_{i-1})}{\Lambda(t_i) - \Lambda(t_{i-1})} \right) \right]. \tag{3.13}
\]

Note that we cannot use (3.11) and (3.13) to estimate $b$ or $\psi$ because we do not know $y_{k+1} = n_k \left( \frac{1 - \bar{\Lambda}(t_k)}{\Lambda(t_k)} \right)$. 

18
3.1.2 Ungrouped Data

In case of ungrouped data, if we want to use the Maximum Likelihood Method to estimate the model parameters, we first have to derive the log-likelihood function of observing the exact failure times \( t_i \) for \( i = 1, \ldots, n \). In Theorem 3.2 we state the log-likelihood function followed by a proof of the statement. The theorem and proof are largely taken from Rigdon and Basu (2000).

**Theorem 3.2** The log-likelihood function of a non-homogeneous Poisson process with mean value function \( \Lambda(t) \) and the observed event times \( 0 = t_0 < t_1 < \cdots < t_n \), is given by

\[
L = \sum_{i=1}^{n} \ln(\lambda(t_i)) - \Lambda(t_n).
\]  
(3.14)

**Proof:** To establish the log-likelihood function, we first derive the joint probability density function for observing exact failure data. From Rigdon and Basu (2000) we use the following relation

\[
f(t_1, t_2, \ldots, t_n) = f_1(t_1)f_2(t_2|t_1)f_3(t_3|t_1, t_2)\cdots f_n(t_n|t_1, t_2, \ldots, t_{n-1}).
\]  
(3.15)

Another relation from the same book states that, because of independent increments, we have for an arbitrary \( k \)

\[
P(T_k > t_k|T_1 = t_1, T_2 = t_2, \ldots, T_{k-1} = t_{k-1}) = P(N(t_{k-1}, t_k] = 0) = e^{-(\Lambda(t_k) - \Lambda(t_{k-1}))}.
\]

To determine the probability density function we first write

\[
P(T_k < t_k|T_{k-1} = t_{k-1}) = 1 - e^{-(\Lambda(t_k) - \Lambda(t_{k-1}))}.
\]

When we differentiate this function we get

\[
f(t_k|t_{k-1}) = \lambda(t_k) e^{-(\Lambda(t_k) - \Lambda(t_{k-1}))}.
\]  
(3.16)

Equations (3.16) in combination with (3.15), gives

\[
f(t_1, t_2, \ldots, t_n) = f_1(t_1)f_2(t_2|t_1)f_3(t_3|t_1, t_2)\cdots f_n(t_n|t_1, t_2 \ldots t_{n-1})
\]

\[
= f_1(t_1)f_2(t_2|t_1)f_3(t_3|t_2)\cdots f_n(t_n|t_{n-1})
\]

\[
= \left[\lambda(t_1) e^{-\Lambda(t_1)}\right] \left[\lambda(t_2) e^{-(\Lambda(t_2) - \Lambda(t_1))}\right] \cdots \left[\lambda(t_k) e^{-(\Lambda(t_k) - \Lambda(t_{k-1}))}\right]
\]

\[
= e^{-\Lambda(t_n)} \prod_{i=1}^{n} \lambda(t_i).
\]

The last expression represents the likelihood function of the observed data. If we take the logarithm of this expression we get (3.14). Note that \( \Lambda(t_n) \) in (3.14) is the result of the telescoping sum \( \sum_{i=1}^{n} (\Lambda(t_i) - \Lambda(t_{i-1})) = \Lambda(t_n) \). □
Analogously to the rewriting of the mean value functions in the previous chapter, all the intensity functions we consider in this report can be written as $a \bar{\lambda}(t)$. Again, the bar on top of $\lambda(t)$ denotes that $a$ is factored out. Using this notation, we can rewrite (3.14) in the following way

$$L = n \ln(a) + \sum_{i=1}^{n} \ln(\bar{\lambda}(t_i)) - a \bar{\Lambda}(t_n).$$

(3.17)

Again, we estimate the parameters by maximising the log-likelihood function (3.17). This is done by differentiating (3.17) and finding the root with respect to the parameter of interest. In case of estimating the parameters for the Goel-Okumoto model and the Yamada S-Shaped model, differentiating (3.17) and setting the expressions equal to zero leads to the following set of equations:

$$\frac{\partial L}{\partial a} = \frac{n}{a} - \bar{\Lambda}(t_n) = 0$$

(3.18)

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} \frac{\partial}{\partial b} \bar{\lambda}(t_i) - a \bar{\Lambda}(t_n) = 0$$

(3.19)

Notice that we can solve (3.18) with respect to $a$ explicitly by writing

$$a = \frac{n}{\bar{\Lambda}(t_n)}.$$  

(3.20)

Inserting the expression for $a$ into (3.19), we obtain

$$\sum_{i=1}^{n} \frac{\partial}{\partial b} \bar{\lambda}(t_i) - \frac{n}{\bar{\Lambda}(t_n)} \frac{\partial}{\partial b} \bar{\Lambda}(t_n) = 0.$$  

(3.21)

Just like in the case of grouped software failure data, because $0 < \bar{\Lambda}(t) < 1$, our estimate of $a$ is never less than the number of faults that are detected. Also, it is not possible to have a singularity in the denominator of our estimation of $a$. Thus, if we have an estimate of $b$, we also have an estimate of $a$. This is why also in case of ungrouped data, we focus on the estimation of $b$. Also note that, because all mean value functions used in this report are monotonically increasing, $\lambda(t)$ is never equal to zero. This implies that the denominator in the summation never causes a singularity. To estimate $b$, we have to solve (3.21). Just like in the grouped data case we refer to the left-hand side of (3.21) by $\Phi(b)$.

$$\Phi(b) = \sum_{i=1}^{n} [ \frac{\partial}{\partial b} \bar{\lambda}(t_i) - n \frac{\partial}{\partial b} \bar{\Lambda}(t_n) ]$$

(3.22)

Because the Inflection S-Shaped model contains one extra parameter, $\psi$, we need one more equation. All steps which were used during the derivation of (3.22) analogously apply to parameter $\psi$. Thus, as estimate of parameter $\psi$ we can use the root of the following function

$$\Phi(\psi) = \sum_{i=1}^{n} [ \frac{\partial}{\partial \psi} \bar{\lambda}(t_i) - n \frac{\partial}{\partial \psi} \bar{\Lambda}(t_n) ]$$

(3.23)
3.2 Trend Test

In this chapter we state different conditions which give an indication whether a data set contains a trend. The reason to include this chapter in the report is to show the close relation between the indication of a trend in the software failure data, which would imply reliability growth or decrease, and the condition which guarantees a solution of the Maximum Likelihood equations. While modelling the software failure process, we assume that the reliability of software increases as faults are found and repaired. Thus, when we want to apply our model to software failure data, we should first verify whether our data indicates reliability growth. Intuitively, this implies that time between failures increases. In a theoretical plot of the expected number of failures until time \( t \) versus \( t \), \( \Lambda(t) = E(N(t)) \), reliability growth would result in a graph which increases but where the increments are a decreasing function of \( t \). This is equivalent to saying that the second derivative of \( \Lambda(t) \) is negative, or saying that \( \Lambda(t) \) is concave. But, in practice, this notion is too strict. A reliability plot which contains local fluctuations may still show overall reliability growth. This is why we state another trend test, the Laplace test (see Rigdon and Basu (2000) Section 4.3). The null hypothesis of the test states that the fault detection process is a homogeneous Poisson process. The alternative states that there is either reliability growth or decrease. In case of ungrouped data, assume that the failure times, \( T_1 < \cdots < T_n \), are failure truncated. Under the null hypothesis and conditioned on the event \( T_n = t_n \), it is known that \( T_1 < \cdots < T_{n-1} \) are distributed as \( n - 1 \) order statistics from a uniform distribution on the interval \((0, t_n)\). Thus, the random variable \( S = \sum_{i=1}^{n} T_i \) has mean \( \frac{(n-1)t_n}{2} \) and variance \( \frac{(n-1)t_n^2}{12} \). Then, under the null hypothesis and by the Central Limit Theorem, we have

\[
U_L = \frac{\sum_{i=1}^{n} T_i - (n+1)t_n/2}{\sqrt{(n-1)t_n^2/12}} \rightarrow Z \sim N(0,1),
\]

(3.24)

where \( N(0,1) \) denotes the standard normal distribution. In case of reliability growth, we expect faults to be more frequently present in the beginning of the time interval \((0, t_n)\). Because this implies that failure times are relatively small, we can state that negative values of \( U_L \) indicate reliability growth. Testing for reliability growth is thus possible using as rejection region \( U_L < -z_\alpha \), where \( z_\alpha \) denotes the \((1-\alpha)\)th percentile of the standard normal distribution. Note that we only test for reliability growth. When \( U_L > -z_\alpha \), we do not reject the null hypothesis but there is still the possibility of reliability decrease. In case of grouped data, we again use a version of the Laplace test which is studied in Kanoun et al. (1991). Assume that all test intervals have the same length \( \ell \). Thus, the observation intervals are given by \([\ell_{i-1}, \ell_i) = [(i-1)\ell, i\ell)\), for all \( i = 1, \ldots, k \), where \( Y_i = N(i\ell) - N((i-1)\ell) \). Under the null hypothesis and conditioned on the event \( \sum_{i=1}^{k} Y_i = nk \), we have

\[
U_L = \frac{\sum_{i=1}^{k} (i-1)Y_i - nk(k-1)/2}{\sqrt{nk(k^2-1)/12}} \rightarrow Z \sim N(0,1),
\]

(3.25)

where \( N(0,1) \) denotes the standard normal distribution. Again, \( U_L < 0 \) is an indication for reliability growth. Testing for reliability growth is thus possible using as rejection region \( U_L < -z_\alpha \), where \( z_\alpha \) denotes the \((1-\alpha)\)th percentile of the standard normal distribution. A generalisation of this test, in case of different interval lengths, is possible using weighted failure counts, but out of the scope of this report.
3.3 Goel-Okumoto Model

In this section we derive the condition for the existence of Maximum Likelihood estimators for the parameters of the Goel-Okumoto model. Recall from Section 3.1 that we only have to find the condition such that we can find an estimate for \( b \) which is the root of \( \Phi(b) \). For both cases, grouped data (Subsection 3.3.1) and ungrouped data (Subsection 3.3.2), we derive the explicit expression for \( \Phi(b) \) and establish the condition for \( \Phi(b) \) to have a unique root. We especially go into detail about proving this condition. Recall that the mean value function and the intensity function of the Goel-Okumoto model are given by

\[
\Lambda(t) = a \left(1 - e^{-bt}\right) \tag{3.26}
\]

and

\[
\lambda(t) = ab e^{-bt}, \tag{3.27}
\]

respectively.

3.3.1 Grouped Data

To establish \( \Phi(b) \) we insert (3.26) into (3.9) and simplify the expression. This gives us

\[
\Phi(b) = \sum_{i=1}^{k} y_i \left[ \frac{\ell_i e^{-b\ell_i} - \ell_{i-1} e^{-b\ell_{i-1}}}{e^{-b\ell_{i-1}} - e^{-b\ell_i}} - \frac{\ell_k e^{-b\ell_k}}{1 - e^{-b\ell_k}} \right]. \tag{3.28}
\]

Note that (3.28) no longer contains the parameter \( a \), only parameter \( b \). Because there is no closed form of the equation, the roots have to be found numerically. Before stating the conditions which have to hold for (3.28) to have a root, we first state two lemmas which we use to prove the correctness of the condition.

**Lemma 3.3** For all real \( a \) and \( b \) we have

\[
\lim_{x \downarrow 0} \frac{b}{e^{bx} - 1} - \frac{a}{e^{ax} - 1} = \frac{1}{2} (a - b).
\]

**Proof:** Expanding denominators into into Taylor series, we see that

\[
\lim_{x \downarrow 0} \frac{b}{e^{bx} - 1} - \frac{a}{e^{ax} - 1} = \lim_{x \downarrow 0} \frac{b}{\left(1 + bx + \frac{1}{2} (bx)^2 + O(x^3)\right) - 1} - \frac{a}{\left(1 + ax + \frac{1}{2} (ax)^2 + O(x^3)\right) - 1}
\]

\[
= \lim_{x \downarrow 0} \frac{1}{x \left(1 + \frac{1}{2} bx + O(x^2)\right)} - \frac{1}{x \left(1 + \frac{1}{2} ax + O(x^2)\right)}
\]

\[
= \lim_{x \downarrow 0} \frac{a - b}{2 \left(1 + \frac{1}{2} (a + b) x + O(x^2)\right)}
\]

\[
= \frac{1}{2} (a - b).
\]

\[\square\]
Lemma 3.4 Let \( g(x) = \frac{x^2 e^x}{(e^x - 1)^2} \), then:

a. \( g(x) \) is a strictly decreasing function for \( x > 0 \).

b. \( \lim_{x \downarrow 0} g(x) = 1 \)

Proof:

a. The derivative of \( g(x) \) can be written as \( \frac{d}{dx} g(x) = \frac{(e^x - 1)(2 - x) - 2x}{(e^x - 1)^3} \). This implies that \( g(x) \) is a decreasing function for \( x > 0 \) if \( t(x) = (e^x - 1)(2 - x) - 2x < 0 \) for \( x > 0 \). The fact that \( \frac{d}{dx} t(0) = 0 \) and \( \frac{d^2}{dx^2} t(x) = -xe^x < 0 \) for \( x > 0 \), implies that \( \frac{d}{dx} t(x) < 0 \) for \( x > 0 \). Thus \( t(x) \) is decreasing for \( x > 0 \), while starting at \( t(0) = 0 \), which implies \( t(x) \) is negative for \( x > 0 \) and thus \( g(x) \) is a decreasing function for \( x > 0 \).

b. \[
\lim_{x \downarrow 0} g(x) = \lim_{x \downarrow 0} \frac{x^2 e^x}{(e^x - 1)^2} = \left( \lim_{x \downarrow 0} \frac{x}{e^x - 1} \right)^2 = \left( \lim_{x \downarrow 0} \frac{1}{1 + O(x)} \right)^2 = 1
\]

The following theorem is due to Knafl (1992) and Hossain and Dahiya (1993). The proof is a slightly simplified and corrected version of the proof given in Hossain and Dahiya (1993).

Theorem 3.5 In case of estimating the parameters of the Goel-Okumoto model using grouped data, a necessary and sufficient condition for the existence of a unique, positive and finite solution of the Maximum Likelihood equations is given by

\[
\ell_k n_k > \sum_{i=1}^{k} y_i (\ell_i + \ell_{i-1}). \tag{3.29}
\]

Proof: Note that \( a \) is positive and finite if and only if \( b \) is positive and finite. Therefore, we only need to show that condition (3.29) is the necessary and sufficient condition for the existence of a unique, positive and finite root of (3.28). We first show that \( \lim_{b \to \infty} \Phi(b) < 0 \) and that \( \Phi(b) \) is a decreasing function. Then we show that (3.29) holds, if and only if \( \lim_{b \downarrow 0} \Phi(b) > 0 \) and thus (3.28) has a positive root. To show that \( \lim_{b \to \infty} \Phi(b) < 0 \), we first
rewrite $\Phi(b)$ by using the properties of a telescoping sum and multiplying with $e^{bl_i}$

\[ \Phi(b) = \sum_{i=1}^{k} y_i \left[ -\frac{\ell_{i-1} e^{b(l_i - l_{i-1})} - \ell_i}{e^{b(l_i - l_{i-1})} - 1} - \frac{\ell_k}{e^{bl_k} - 1} \right] \]

\[ = \sum_{i=1}^{k} y_i \left[ -\ell_{i-1} \frac{e^{b(l_i - l_{i-1})} - 1}{e^{b(l_i - l_{i-1})} - 1} + \frac{\ell_i}{e^{b(l_i - l_{i-1})} - 1} - \frac{\ell_k}{e^{bl_k} - 1} \right] \]

\[ = \sum_{i=1}^{k} y_i \left[ \frac{\ell_{i-1} (e^{b(l_i - l_{i-1})} - 1)}{e^{b(l_i - l_{i-1})} - 1} + \frac{\ell_i - \ell_{i-1}}{e^{b(l_i - l_{i-1})} - 1} - \frac{\ell_k}{e^{bl_k} - 1} \right] \]

\[ = \sum_{i=1}^{k} y_i \left[ \frac{\ell_i - \ell_{i-1}}{e^{b(l_i - l_{i-1})} - 1} - \frac{\ell_k}{e^{bl_k} - 1} - \ell_{i-1} \right] \]

Note that

\[ \lim_{b \to \infty} \Phi(b) = -\sum_{i=1}^{k} y_i \ell_{i-1} < 0. \]

Next, we show that $\Phi(b)$ is a decreasing function. Calculating the derivative of $\Phi(b)$ with respect to $b$, we obtain

\[ \frac{\partial}{\partial b} \Phi(b) = -\sum_{i=1}^{k} y_i \left[ (\ell_i - \ell_{i-1})^2 \frac{e^{b(l_i - l_{i-1})}}{(e^{b(l_i - l_{i-1})} - 1)^2} - \frac{\ell_k^2 e^{b(l_i - l_{i-1})}}{(e^{b(l_i - l_{i-1})} - 1)^2} \right]. \]

Lemma 3.4 states that, because $\ell_k \geq \ell_i \geq \ell_i - \ell_{i-1}$, the expression between square brackets is strictly positive and thus $\frac{\partial}{\partial b} \Phi(b) < 0$, which implies $\Phi(b)$ is a decreasing function. The fact that $\lim_{b \to \infty} \Phi(b) < 0$ and $\Phi(b)$ is a decreasing function implies that $\Phi(b)$ has a positive root if and only if $\lim_{b \to 0} \Phi(b) > 0$. Lemma 3.3 yields

\[ \lim_{b \to 0} \Phi(b) = \sum_{i=1}^{k} y_i \left[ \frac{1}{2} (\ell_k - (\ell_i - \ell_{i-1})) - \ell_{i-1} \right] = \frac{1}{2} \left[ n_k \ell_k - \sum_{i=1}^{k} y_i (\ell_i + \ell_{i-1}) \right]. \]

This expression is positive if and only if $n_k \ell_k > \sum_{i=1}^{k} y_i (\ell_i + \ell_{i-1})$. □

To make a statement about reliability growth we use the theory of Section 3.2. If we assume the intervals to be of the same length, we can rewrite (3.29) the following way

\[ \sum_{i=1}^{k} y_i (i + (i - 1)) < n_k k, \]

\[ 2 \sum_{i=1}^{k} y_i i - n_k (k + 1) < 0, \]

\[ \sum_{i=1}^{k} y_i i - n_k (k + 1) / 2 < 0. \]

The last expression equals the numerator of (3.25). Thus, if (3.28) does not have a root, (3.25) is positive, which implies there is no reliability growth.
3.3.2 Ungrouped Data

To establish $\Phi(b)$, we substitute (3.27) and (3.26) into (3.22). We obtain

$$\Phi(b) = \frac{n}{b} - \frac{nt_n}{e^{bt_n} - 1} - \sum_{i=1}^{n} t_i.$$  \hspace{1cm} (3.30)

Before stating the conditions which have to hold for (3.30) to have a root, we first state a lemma. We use this lemma to prove the correctness of the condition.

**Lemma 3.6** $\lim_{x \downarrow 0} \frac{e^x - (1 + x)}{x(e^x - 1)} = \frac{1}{2}$

**Proof:**

\[
\lim_{x \downarrow 0} g(x) = \lim_{x \downarrow 0} \frac{e^x - (1 + x)}{x(e^x - 1)} = \lim_{x \downarrow 0} \frac{1 + x + \frac{1}{2}x^2 + O(x^3) - (1 + x)}{x(1 + x + \frac{1}{2}x^2 + O(x^3) - 1)} = \lim_{x \downarrow 0} \frac{\frac{1}{2}x^2 + O(x^3)}{x^2 + O(x^3)} = \frac{1}{2}
\]

\[\square\]

The following theorem is due to Hossain and Dahiya (1993) and Knafl and Morgan (1996) and the proof is a slightly simplified version of Hossain and Dahiya (1993).

**Theorem 3.7** In case of estimating the parameters of the Goel-Okumoto model using ungrouped data, a necessary and sufficient condition for the existence of a unique, positive and finite solution of the Maximum Likelihood equations is given by

$$t_n > 2\sum_{i=1}^{n} t_i.$$ \hspace{1cm} (3.31)

**Proof:** Again we only have to prove (3.31) ensures a unique, positive and finite estimate for $b$. When we have found such an estimate for $b$, this automatically gives us a suitable estimate for $a$. The proof goes analogously to the proof given in Subsection 3.3.1. We first show that $\lim_{b \to \infty} \Phi(b) < 0$ and $\Phi(b)$ is a decreasing function. Then we show that if and only if (3.31) holds, $\lim_{b \downarrow 0} \Phi(b) > 0$ and thus, if and only if (3.31) holds, (3.30) has a positive root. We start with determining $\lim_{b \to \infty} \Phi(b)$.

\[
\lim_{b \to \infty} \Phi(b) = \lim_{b \to \infty} \frac{n}{b} - \frac{nt_n}{e^{bt_n} - 1} - \sum_{i=1}^{n} t_i = -\sum_{i=1}^{n} t_i
\]

This implies

$$\lim_{b \to \infty} \Phi(b) = -\sum_{i=1}^{n} t_i < 0.$$
Next, we show $\Phi(b)$ is a decreasing function. Differentiating $\Phi(b)$ and simplifying the expression, we get

$$
\frac{\partial}{\partial b} \Phi(b) = -\frac{n}{b^2} \left( 1 - \left( \frac{bt_n}{e^{bt_n} - 1} \right)^2 e^{bt_n} \right).
$$

According to Lemma 3.4, the function $g(b) = \left( \frac{bt_n}{e^{bt_n} - 1} \right)^2 e^{bt_n}$ is decreasing and $\lim_{b \to 0} g(b) = 1$. Because $g(b) < 1$, the derivative of $\Phi(b)$ with respect to $b$ is negative and thus $\Phi(b)$ is a decreasing function. Now we have shown that $\lim_{b \to \infty} \Phi(b) < 0$ and $\Phi(b)$ is a decreasing function. This implies that $\Phi(b)$ has a positive root if and only if $\lim_{b \to 0} \Phi(b) > 0$. If we rewrite $\Phi(b)$ and use Lemma 3.6 we get

$$
\lim_{b \downarrow 0} \Phi(b) = \lim_{b \downarrow 0} \frac{n}{b} - \frac{nt_n}{e^{bt_n} - 1} - \sum_{i=1}^{n} t_i
$$

$$
= \lim_{b \downarrow 0} -\sum_{i=1}^{n} t_i + n t_n \left( \frac{e^{bt_n} - (1 + bt_n)}{bt_n (e^{bt_n} - 1)} \right)
$$

$$
= -\sum_{i=1}^{n} t_i + \frac{nt_n}{2}
$$

Clearly, for the last expression to be positive, condition (3.31) must be satisfied. □

To make a statement about reliability growth we use the theory of Section 3.2. We can rewrite (3.31) as $\sum_{i=1}^{n} t_i - nt_n/2 < 0$. Recall, that the numerator of (3.24) equals $\sum_{i=1}^{n} t_i - (n + 1) t_n/2$. Because $\sum_{i=1}^{n} t_i - (n + 1) t_n/2 < \sum_{i=1}^{n} t_i - nt_n/2 < 0$, we can state that if (3.30) has a root there is an indication for reliability growth. But, note that we only reject the null hypothesis if $\sum_{i=1}^{n} t_i - (n + 1) t_n/2 < -z_\alpha$, which may not be the case, even when (3.30) has a root.

### 3.4 Yamada S-Shaped Model

In this section we derive the condition for the existence of Maximum Likelihood estimators for the parameters of the Yamada S-Shaped model. Recall from Section 3.1 that we only need a condition such that we can find an estimate for $b$ which is the root of $\Phi(b)$. For both cases, grouped data (Subsection 3.4.1) and ungrouped data (Subsection 3.4.2), we derive an explicit expression for $\Phi(b)$ and establish the necessary and sufficient condition for $\Phi(b)$ to have a unique root. We especially go into detail about proving this condition. Recall that the mean value function and the intensity function of the Yamada S-Shaped model are given by

$$
\Lambda(t) = a \left( 1 - (1 + bt) e^{-bt} \right)
$$

and

$$
\lambda(t) = ab^2 te^{-bt},
$$

respectively.
3.4.1 Grouped Data

To establish \( \Phi(b) \) we insert (3.32) into (3.9) and simplify the expression, we get

\[
\Phi(b) = \sum_{i=1}^{k} y_i \left[ \frac{b \ell_i^2 e^{-b \ell_i} - b \ell_{i-1}^2 e^{-b \ell_{i-1}}}{(1 + b \ell_{i-1}) e^{-b \ell_{i-1}} - (1 + b \ell_i) e^{-b \ell_i}} - \frac{b \ell_k^2 e^{-b \ell_k}}{1 - (1 + b \ell_k) e^{-b \ell_k}} \right]
\] (3.34)

Just like in the previous chapter, to obtain \( b \) from (3.34), we have to use numerical methods. The following lemmas are used to prove the condition which has to be satisfied for (3.34) to have a unique, positive and finite root.

**Lemma 3.8** For \( 0 < s < t < u \) we have

\[
\lim_{x \downarrow 0} \frac{t^2 x}{(1 + sx) e^{(t-s)x} - (1 + tx)} + \frac{s^2 xe^{(t-s)x}}{(1 + tx) - (1 + sx) e^{(t-s)x}} - \frac{u^2 x}{e^{ux} - (1 + ux)} = \frac{2}{3} \left[ u - \left( t + \frac{s^2}{s+t} \right) \right]
\]

**Proof:** The proof of this lemma goes analogously to the proof of Lemmas 3.4 and 3.6: we replace the exponential by the appropriate Taylor approximation. Because this lemma involves a relatively large expression and the proof only works if we replace \( e^{\alpha x} \) by \( T = 1 + \alpha x + \frac{1}{2} (\alpha x)^2 + \frac{1}{6} (\alpha x)^3 + O(x^4) \), we only give a sketch of the proof in which we neglect \( O(x^4) \). Define \( g(x) \) to be the left-hand side of the expression in Lemma 3.8. After substituting the Taylor approximations, \( \lim_{x \downarrow 0} g(x) \) can be rewritten as follows

\[
\lim_{x \downarrow 0} g(x) = \lim_{x \downarrow 0} -s + \frac{2u}{3 + ux} + \frac{s^2 + st - 2t^2 - s(s-t)^2}{3(s+t) + (t^2 + st - 2s^2)x + s(s-t)^2 x^2}
\]

\[
= -s + \frac{2}{3}u + \frac{s^2 + st - 2t^2}{3(s+t)}
\]

\[
= \frac{2}{3} \left[ u - \left( t + \frac{s^2}{s+t} \right) \right]
\]

\( \square \)
Lemma 3.9 The function
\[
\Phi(b) = \frac{b (e^{bt} s^2 - e^{bs} t^2)}{-e^{bt}(1 + bs) + e^{bs}(1 + bt)} - \frac{b (e^{bu} u^2 - e^{bu} v^2)}{-e^{bu}(1 + bu) + e^{bu}(1 + bv)}
\]
is decreasing for \(u < s < t < v\).

Proof: Define
\[
f(i, j) = \frac{b (e^{bi} i^2 - e^{bj} j^2)}{-e^{bi}(1 + bi) + e^{bj}(1 + bj)}.
\]
We can show \(\Phi(b)\) is decreasing by showing that
\[
\frac{\partial}{\partial b} f(s, t) - \frac{\partial}{\partial b} f(u, v) < 0.
\]
This can be done by showing that, for \(s < t\)
\[
\frac{\partial^2}{\partial s \partial b} f(s, t) < 0
\]
and
\[
\frac{\partial^2}{\partial t \partial b} f(s, t) > 0.
\]
Because \(f(s, t)\) is symmetric in \(s\) and \(t\), we only have to show that, for \(t < s\),
\[
\frac{\partial^2}{\partial s \partial b} f(s, t) < 0
\]
and, for \(t > s\),
\[
\frac{\partial^2}{\partial t \partial b} f(s, t) > 0.
\]
If we calculate \(\frac{\partial^2}{\partial s \partial b} f(s, t)\), we get
\[
t \left(2 e^{3bs} + e^{b(s+2t)} (2 + b(1 + bs)(s - t)(4 + bs(3 + bs) - b(1 + bs)t)) + e^{b(2s+t)} \left(-4 + b(s - t) \left(-4 + b \left(bs^2(1 + bt) - t(1 + bt) - s(5 + bt(2 + bt))\right)\right)\right)\right) \right) \right)
\]
Because the denominator, \((e^{bt}(1 + bs) - e^{bs}(1 + bt))^3\), is less than zero for \(t < s\) and greater than zero for \(t > s\), we only need to show that the numerator is positive for all \(t\). We first prove that the numerator is positive for \(t < s\). Thus,
\[
t \left(2 e^{3bs} + e^{b(s+2t)} (2 + b(1 + bs)(s - t)(4 + bs(3 + bs) - b(1 + bs)t)) + e^{b(2s+t)} \left(-4 + b(s - t) \left(-4 + b \left(bs^2(1 + bt) - t(1 + bt) - s(5 + bt(2 + bt))\right)\right)\right)\right) > 0.
\]
If we divide both sides of the inequality by \( te^{b(s+2t)} \), we get

\[
2e^{2b(s-t)} + (2 + b(1+bs)(s-t)(4+bs(3+bs)-b(1+bs))) + e^{b(s-t)} \left( -4 + b(s-t) \right) \left( -4 + b \left( b(1+bt) - t(1+bt) - s(5 + bt(2 + bt)) \right) \right) > 0.
\]

We can rewrite the polynomials which occur in the inequality above and substitute \( x = b(s-t) \). This gives us

\[
2e^{2x} + (2 + (4 + 6bs + 2b^2 s^2) x + (1 + 2bs + b^2 s^2) x^2) - e^x \left( 4 + (4 + 6bs + 2b^2 s^2) x - (1 + 4bs + b^2 s^2) x^2 + (1 + bs)x^3 \right) > 0.
\]

To further simplify the expression, we replace \( bs \) by \( \alpha \).

Note that \( 0 < t < s \) and thus \( 0 < x < bs \). Define the left-hand side of the inequality above to be \( G(x) \). We prove \( G(x) \) is positive by showing that

\[
\frac{d}{dx} G(x) = \frac{d^2}{dx^2} G(x) = \frac{d^3}{dx^3} G(x) = 0 \quad \text{and} \quad \frac{d^4}{dx^4} G(x) > 0.
\]

If we calculate the derivatives, the first statement is easy to verify.

\[
\begin{align*}
G(x) &= 2e^{2x} + (2 + (4 + 6\alpha + 2\alpha^2) x + (1 + 2\alpha + \alpha^2) x^2) - e^x \left( 4 + (4 + 6\alpha + 2\alpha^2) x - (1 + 4\alpha + \alpha^2) x^2 + (1 + \alpha)x^3 \right) \\
\frac{d}{dx} G(x) &= 4e^{2x} + 2(1 + \alpha)(2 + x + \alpha(1+x)) - e^x \left( 8 - \alpha^2 (-2 + x^2) + \alpha(6 + (-2 + x)x(1+x)) + x(2 + x(2 + x)) \right) \\
\frac{d^2}{dx^2} G(x) &= 8e^{2x} + 2(1 + \alpha)^2 - e^x \left( 10 + x(2 + x)(3 + x) - \alpha^2(-2 + x(2 + x)) + \alpha(4 + x(-4 + x(2 + x))) \right) \\
\frac{d^3}{dx^3} G(x) &= e^x \left( -16 + 16e^x - x \left( -\alpha^2(4 + x) + (4 + x)^2 + \alpha x(5 + x) \right) \right)
\end{align*}
\]

Now we shall prove \( \frac{d^4}{dx^4} G(x) > 0 \) for \( x > \alpha \). For this to be true we must show that

\[
-16 + 16e^x - x \left( -\alpha^2(4 + x) + (4 + x)^2 + \alpha x(5 + x) \right) > 0,
\]

which can be rewritten to

\[
e^x > 1 + \left( 1 - \frac{\alpha^2}{2} \right) x + \frac{1}{2} + \frac{\alpha(5 - \alpha)}{16} x^2 + \frac{1 + \alpha}{16} x^3.
\]

(3.35)

For this inequality to be true we must prove it for all \( \alpha \). The easiest way to do this, is by proving the inequality for the value of \( \alpha \) which maximises the right-hand side of (3.35), define this value to be \( \alpha_{\max} \). Then we have

\[
e^x > 1 + \left( 1 - \frac{\alpha_{\max}^2}{2} \right) x + \left( \frac{1}{2} + \frac{\alpha_{\max}(5 - \alpha_{\max})}{16} \right) x^2 + \frac{1 + \alpha_{\max}}{16} x^3 > 1 + \left( 1 - \frac{\alpha^2}{2} \right) x + \frac{1}{2} + \frac{\alpha(5 - \alpha)}{16} x^2 + \frac{1 + \alpha}{16} x^3.
\]
Setting the derivative of the right-hand side of (3.35) equal to 0 and solving to $\alpha$ gives

$$\alpha_{\max} = \frac{5x + x^2}{2(4 + x)}.$$  

If we insert $\alpha_{\max}$ into (3.35) we get

$$e^x > \frac{256 + 320x + 192x^2 + 73x^3 + 14x^4 + x^5}{64(4 + x)},$$

which implies

$$(4 + x)e^x > 4 + 5x + 3x^2 + \frac{73}{64}x^3 + \frac{7}{32}x^4 + \frac{1}{64}x^5.$$  

If we use the Taylor approximation of $(4 + x)e^x$ we can write

$$(4 + x)e^x > 4 + 5x + 3x^2 + \frac{7x^3}{6} + \frac{x^4}{3} + \frac{3x^5}{40} > 4 + 5x + 3x^2 + \frac{73}{64}x^3 + \frac{7}{32}x^4 + \frac{1}{64}x^5.$$  

This inequality holds because

$$\frac{7x^3}{6} + \frac{x^4}{3} + \frac{3x^5}{40} > \frac{73}{64}x^3 + \frac{7}{32}x^4 + \frac{1}{64}x^5.$$  

This finishes the first half of the proof. Now, we prove that the numerator of $\frac{\partial^2}{\partial h^2}f(s, t)$ is positive for $t > s$. To prove this, we again use the function $G(x)$. Because we would like to work with $x > 0$, we replace all $x$ with $-x$ and multiply $G(x)$ with $e^{2x}$. We get the following expression

$$2 + e^{2x} \left(2 - (4 + 6\alpha + 2\alpha^2)x + (1 + 2\alpha + \alpha^2)x^2\right) - e^x \left(4 - (4 + 6\alpha + 2\alpha^2)x - (1 + 4\alpha + \alpha^2)x^2 - (1 + \alpha)x^3\right)$$

Define this expression to be $H_0(x)$.

$$H_0(x) = 2 + e^{2x} \left(2 - (4 + 6\alpha + 2\alpha^2)x + (1 + 2\alpha + \alpha^2)x^2\right) - e^x \left(4 - (4 + 6\alpha + 2\alpha^2)x - (1 + 4\alpha + \alpha^2)x^2 - (1 + \alpha)x^3\right)$$

Because $H_0(s) = 0$, we can say that $H_0(x) > 0$ if

$$\frac{d}{dx} H_0(x) = e^x \left(x(6 + 2e^x(-3 + x) + x(4 + x)) + \alpha^2(2 + x(4 + x) + 2e^x(-1 + (-1 + x)x)) + \alpha(e^x(-6 + 4(-2 + x)x) + (3 + x)(2 + x(4 + x)))\right) > 0.$$  

The inequality $\frac{d}{dx} H_0(x) > 0$ holds if

$$H_1(x) = x(6 + 2e^x(-3 + x) + x(4 + x)) + \alpha^2(2 + x(4 + x) + 2e^x(-1 + (-1 + x)x)) + \alpha(e^x(-6 + 4(-2 + x)x) + (3 + x)(2 + x(4 + x))) > 0.$$
Because $H_1(s) = 0$, we can say that $H_1(x) > 0$ if

$$
\frac{d}{dx} H_1(x) = 6 + 2\alpha^2(2 + x) + x(8 + 3x) + \alpha(14 + x(14 + 3x)) + 2e^x (-3 + (-1 + x)x + \alpha^2 (-2 + x + x^2) + \alpha (-7 + 2x^2)) > 0.
$$

The inequality $\frac{d}{dx} H_1(x) > 0$ holds if

$$
H_2 = 6 + 2\alpha^2(2 + x) + x(8 + 3x) + \alpha(14 + x(14 + 3x)) + 2e^x (-3 + (-1 + x)x + \alpha^2 (-2 + x + x^2) + \alpha (-7 + 2x^2)) > 0.
$$

Because $H_2(s) = 0$, we can say that $H_2(x) > 0$ if

$$
\frac{d}{dx} H_2(x) = 2 (4 + 3x + \alpha(7 + \alpha + 3x) + e^x (-4 - \alpha(7 + \alpha) + x + \alpha(4 + 3\alpha)x + (x + \alpha x)^2)) > 0.
$$

The inequality $\frac{d}{dx} H_2(x) > 0$ holds if

$$
H_3 = 4 + 3x + \alpha(7 + \alpha + 3x) + e^x (-4 - \alpha(7 + \alpha) + x + \alpha(4 + 3\alpha)x + (x + \alpha x)^2)) > 0.
$$

Because $H_3(s) = 0$, we can say that $H_3(x) > 0$ if

$$
\frac{d}{dx} H_3(x) = 3 + 3\alpha + e^x (-3 + x(3 + x) + \alpha(-3 + 2x(4 + x)) + \alpha^2(2 + x(5 + x))) > 0.
$$

The inequality $\frac{d}{dx} H_3(x) > 0$ holds if

$$
H_4 = 3 + 3\alpha + e^x (-3 + x(3 + x) + \alpha(-3 + 2x(4 + x)) + \alpha^2(2 + x(5 + x))) > 0.
$$

Because $H_4(s) = 2\alpha^2$, we can say that $H_4(x) > 0$ if

$$
\frac{d}{dx} H_4(x) = e^x (x(5 + x) + \alpha(5 + 2x(6 + x)) + \alpha^2(7 + x(7 + x))) > 0.
$$

This last inequality is obviously true. □
The following theorem is largely due to Knafl (1992). A formal proof that \( \Phi(b) \) is a decreasing function was not provided in Knafl (1992) but is included in this report.

**Theorem 3.10** In case of estimating the parameters of the Yamada S-Shaped model using grouped data, a necessary and sufficient condition for the existence of a unique, positive and finite solution of the Maximum Likelihood equations is given by

\[
n_k \ell_k > \sum_{i=1}^{k} y_i \left[ \ell_i + \frac{\ell_i^2}{\ell_{i-1} + \ell_i} \right]. \tag{3.36}
\]

**Proof:** This proof is also similar to that of the grouped data case of the Goel-Okumoto Model. Notice that, again, \( \alpha \) is positive and finite if and only if \( b \) is positive. Therefore we only need to prove that (3.36) is a necessary and sufficient condition for the existence of a unique root of (3.34). We first show that \( \lim_{b \to \infty} \Phi(b) < 0 \) and \( \Phi(b) \) is a decreasing function. Then, we show that if and only if (3.36) holds, \( \lim_{b \to 0} \Phi(b) > 0 \) and thus, if and only if (3.36) holds (3.34) has a positive root. We start with rewriting \( \Phi(b) \) the following way

\[
\Phi(b) = \sum_{i=1}^{k} y_i \left[ \frac{b \ell_i^2 e^{-b \ell_i} - b \ell_{i-1}^2 e^{-b \ell_{i-1}}}{(1 + b \ell_{i-1}) e^{-b \ell_{i-1}} - (1 + b \ell_i) e^{-b \ell_i}} - \frac{b \ell_k^2 e^{-b \ell_k}}{1 - (1 + b \ell_k) e^{-b \ell_k}} \right] - \sum_{i=1}^{k} y_i \left[ \frac{b \ell_k^2 e^{-b \ell_k}}{1 - (1 + b \ell_k) e^{-b \ell_k}} \right] = \sum_{i=1}^{k} y_i \left[ \frac{b \ell_i^2}{(1 + b \ell_{i-1}) e^{(\ell_i - \ell_{i-1})b} - (1 + b \ell_i) e^{(\ell_i - \ell_{i-1})b}} + \frac{b \ell_{i-1}^2 e^{-b \ell_{i-1}}}{(1 + b \ell_i) e^{-b \ell_i} - (1 + b \ell_{i-1}) e^{-b \ell_{i-1}}} \right] - \sum_{i=1}^{k} y_i \left[ \frac{b \ell_k^2}{e^{b \ell_k} - (1 + b \ell_k)} \right]
\]

Now, we prove that \( \lim_{b \to \infty} \Phi(b) < 0 \). It is obvious that

\[
\lim_{b \to \infty} \Phi(b) = \lim_{b \to \infty} \sum_{i=1}^{k} y_i \left[ \frac{b \ell_{i-1}^2 e^{(\ell_i - \ell_{i-1})b}}{(1 + b \ell_i) - (1 + b \ell_{i-1}) e^{(\ell_i - \ell_{i-1})b}} \right] - \sum_{i=1}^{k} y_i \ell_{i-1}
\]

This implies

\[
\lim_{b \to \infty} \Phi(b) = - \sum_{i=1}^{k} y_i \ell_{i-1} < 0.
\]
To show that $\Phi(b)$ is a decreasing function we rewrite $\Phi(b)$ the following way

$$\Phi(b) = \sum_{i=1}^{k} y_i \left[ \frac{bl_i^2 e^{-bl_i} - bl_{i-1}^2 e^{-bl_{i-1}}}{1 + bl_{i-1}} - \frac{bl_k^2 e^{-bl_k}}{1 + bl_k} \right]$$

Applying Lemma 3.9 to this expression shows us that $\Phi(b)$ is a decreasing function. Now that we have shown that $\Phi(b)$ is a decreasing function and $\lim_{b \to \infty} \Phi(b) < 0$, the only thing left to prove is that $\lim_{b \downarrow 0} \Phi(b) > 0$ if and only if (3.36) holds. Because of Lemma 3.8 we can write

$$\lim_{b \downarrow 0} \Phi(b) = \frac{2}{3} \left( \ell_k n_k - \sum_{i=1}^{k} y_i \left[ \ell_i + \frac{\ell_{i-1}^2}{\ell_{i-1} + \ell_i} \right] \right),$$

which is positive if $\ell_k n_k > \sum_{i=1}^{k} y_i \left[ \ell_i + \frac{\ell_{i-1}^2}{\ell_{i-1} + \ell_i} \right]$.

To make a statement about reliability growth we use the theory of Section 3.2. If we assume the intervals to be of the same length, we can rewrite (3.36) to

$$\sum_{i=1}^{k} y_i \left[ \left( i + \frac{(i-1)^2}{i+(i-1)} \right) - n_k \right] < 0.$$

If we rewrite the numerator of (3.25) we get

$$\sum_{i=1}^{k} i y_i - \frac{n_k (1 + k)}{2} < 0,$$

$$\sum_{i=1}^{k} 2t y_i - n_k (1 + k) < 0,$$

$$\sum_{i=1}^{k} i + (i-1) y_i - n_k k < 0.$$

Note that $\sum_{i=1}^{k} (i + \frac{(i-1)^2}{i+(i-1)}) y_i - n_k k < \sum_{i=1}^{k} i + (i-1) y_i - n_k k$. Thus, reliability growth implies (3.36) has a root. Note that when the numerator is negative, (3.25) does not have to be smaller than $-z_\alpha$, which would imply we still do not reject our null hypothesis.

### 3.4.2 Ungrouped Data

We calculate $\Phi(b)$ by substituting (3.32) and (3.33) into (3.22). We get

$$\Phi(b) = \frac{-nb t_n^2 e^{-bt_n}}{1 - (1 + bl_n) e^{-bl_n}} - \sum_{i=1}^{n} t_i + \frac{2n}{b}. \quad (3.37)$$

Before we give the condition which has to be satisfied for (3.37) to have a positive root, we give Lemma 3.11 and Lemma 3.12.
Lemma 3.11 \( g(x) = \frac{2}{x} - \frac{1}{x(e^x - (1+x))} \) is a decreasing function for \( x > 0 \).

Proof: The derivative of \( g(x) \) can be written as \( \frac{d}{dx} g(x) = \frac{e^x(x^3 - x^2 + 4x + 4) - (x^2 + 4x + 2) - 2e^{2x}}{x^2(e^x - (1+x))^2} \).

For \( g(x) \) to be decreasing for \( x > 0 \), the denominator of the derivative must be negative for \( x > 0 \). Thus, \( t(x) = e^x(x^3 - x^2 + 4x + 4) - (x^2 + 4x + 2) - 2e^{2x} < 0 \) for \( x > 0 \). In the proof of Lemma 3.4 we used the fact that a function \( f(x) \) is negative for \( x > 0 \) if \( f(0) = 0 \) and \( \frac{d}{dx} f(x) < 0 \) for \( x > 0 \). To prove this lemma, we use the same way of reasoning, repeatedly. Because \( t(0) = 0 \) we need to show that \( \frac{d}{dx} t(x) = e^x(x^3 + 2x^2 + 8x + 2 - (2x + 4) - 4e^{2x} < 0 \) for \( x > 0 \). This implies, because \( \frac{d}{dx} t(0) = 0 \), we need to show that \( \frac{d^2}{dx^2} t(x) = e^x(x^3 + 5x^2 + 6x + 10) - 2 - 8e^{2x} < 0 \) for \( x > 0 \). This implies, because \( \frac{d^2}{dx^2} t(0) = 0 \) we need to show that \( \frac{d^3}{dx^3} t(x) = e^x(x^3 + 8x^2 + 16x + 16) - 16e^{2x} < 0 \) for \( x > 0 \). Again, note that \( \frac{d^4}{dx^4} t(0) = 0 \). The only thing left to prove is \( e^x(x^3 + 8x^2 + 16x + 16) - 16e^{2x} < 0 \) for \( x > 0 \). This inequality can be rewritten as \( e^x > \frac{1}{16}x^3 + \frac{1}{2}x^2 + x + 1 \) for \( x > 0 \). This last inequality is obviously true because when we use the Taylor expansion of \( e^x \) we can write \( e^x > \frac{1}{6}x^3 + \frac{1}{2}x^2 + x + 1 \).

\[ \text{Lemma 3.12} \lim_{x \downarrow 0} g(x) = \frac{2(e^x - (1+x)) - x^2}{x(e^x - (1+x))} = \frac{2}{3}. \]

Proof:

\[
\lim_{x \downarrow 0} g(x) = \lim_{x \downarrow 0} \frac{2(e^x - (1+x)) - x^2}{x(e^x - (1+x))} = \lim_{x \downarrow 0} \frac{2\left(1 + x + \frac{1}{2}x^2 + \frac{1}{6}x^3 + O(x^4) - (1 + x)\right) - x^2}{x\left(1 + x + \frac{1}{2}x^2 + \frac{1}{6}x^3 + O(x^4) - (1 + x)\right)} = \lim_{x \downarrow 0} \frac{2\left(\frac{1}{2}x^2 + \frac{1}{6}x^3 + O(x^4)\right) - x^2}{\frac{1}{2}x^2 + \frac{1}{6}x^3 + O(x^4)} = \frac{2}{3}.
\]

The following theorem is a special case of the proof concerning the Gamma Family of software reliability models in Knafl and Morgan (1996) and the proof given in Zhao and Xie (1996).

**Theorem 3.13** In case of estimating the parameters of the Yamada S-Shaped model using ungrouped data, a necessary and sufficient condition for the existence of a unique, positive and finite solution of the Maximum Likelihood equations is given by

\[
\frac{2}{3}t_n > \frac{\sum_{i=1}^{n} t_i}{n}.
\]

**Proof:** The following proof is similar to the proof we gave for the condition in case of grouped data. We prove that \( \lim_{b \to \infty} \Phi(b) > 0 \) and that \( \Phi(b) \) is decreasing. Finally, we prove that, if
and only if (3.38) holds, \( \lim_{b \downarrow 0} \Phi(b) < 0 \). We calculate \( \lim_{b \to \infty} \Phi(b) \) the following way

\[
\lim_{b \to \infty} \Phi(b) = \lim_{b \to \infty} \frac{-nbt^2_n e^{-bt_n}}{1 - (1 + bt_n) e^{-bt_n}} + \frac{2n}{b} - \sum_{i=1}^{n} t_i
\]

\[
= \lim_{b \to \infty} \frac{-nbt^2_n}{e^{bt_n} - (1 + bt_n)} + \frac{2n}{b} - \sum_{i=1}^{n} t_i
\]

\[
= -\sum_{i=1}^{n} t_i.
\]

This implies that \( \lim_{b \to \infty} \Phi(b) < 0 \). We can see that \( \Phi(b) \) is a decreasing function if we rewrite \( \Phi(b) \) in the following way

\[
\Phi(b) = \frac{-nbt^2_n e^{-bt_n}}{1 - (1 + bt_n) e^{-bt_n}} + \frac{2n}{b} - \sum_{i=1}^{n} t_i
\]

\[
= \frac{-nbt^2_n}{e^{bt_n} - (1 + bt_n)} + \frac{2n}{b} - \sum_{i=1}^{n} t_i
\]

\[
= nt_n \left[ \frac{2 (e^{bt_n} - (1 + bt_n)) - (bt_n)^2}{(bt_n)(e^{bt_n} - (1 + bt_n))} \right] - \sum_{i=1}^{n} t_i.
\]

Thus, \( \Phi(b) \) is decreasing if the part between square brackets is decreasing. The fact that this is decreasing follows from Lemma 3.11. Now that we have shown that \( \lim_{b \to \infty} \Phi(b) < 0 \) and \( \Phi(b) \) is a decreasing function, the only way \( \Phi(b) \) can have a unique, positive and finite root is if \( \lim_{b \downarrow 0} \Phi(b) > 0 \). We show this is the case if and only if (3.38) is satisfied. If we rewrite \( \Phi(b) \) the same way as before and use Lemma 3.12, we can calculate \( \lim_{b \downarrow 0} \Phi(b) \) as follows

\[
\lim_{b \downarrow 0} \Phi(b) = \lim_{b \downarrow 0} \frac{-nbt^2_n e^{-bt_n}}{1 - (1 + bt_n) e^{-bt_n}} + \frac{2n}{b} - \sum_{i=1}^{n} t_i
\]

\[
= \lim_{b \downarrow 0} \frac{-nbt^2_n}{e^{bt_n} - (1 + bt_n)} + \frac{2n}{b} - \sum_{i=1}^{n} t_i
\]

\[
= \lim_{b \downarrow 0} nt_n \left[ \frac{2 (e^{bt_n} - (1 + bt_n)) - (bt_n)^2}{(bt_n)(e^{bt_n} - (1 + bt_n))} \right] - \sum_{i=1}^{n} t_i
\]

\[
= -\sum_{i=1}^{n} t_i + \frac{2}{3} nt_n.
\]

The last expression is positive if and only if (3.38) is satisfied.

To make a statement about reliability growth we use the theory of Section 3.2. We can rewrite (3.38) to

\[
\sum_{i=1}^{n} t_i - \frac{2nt_n}{3} < 0.
\]

For \( n > 3 \) we know that \( \frac{2nt_n}{3} > \frac{(n+1)t_n}{2} \), which implies \( \sum_{i=1}^{n} t_i - \frac{2nt_n}{3} < \sum_{i=1}^{n} t_i - \frac{(n+1)t_n}{2} \). Note that the right-hand side of the inequality equals the numerator of (3.24). Thus, reliability growth implies (3.37) has a root. Again, note that we only reject the null hypothesis if (3.24) is smaller than \( -z_\alpha \), which may not be the case when \( \sum_{i=1}^{n} t_i - (n+1)t_n/2 < 0 \).
3.5 Inflection S-Shaped Model

In this section we derive conditions such that the parameters of the Inflection S-Shaped model can be estimated with the Maximum Likelihood Method. Recall from Section 3.1 that we only need a condition such that we can find an estimate for $b$ and $\psi$ which are roots of $\Phi(b)$ and $\Phi(\psi)$, respectively. For both cases, grouped data (Subsection 3.5.1) and ungrouped data (Subsection 3.5.2), we derive the explicit expression for $\Phi(b)$ and $\Phi(\psi)$ and establish the conditions for these functions to have at least one root. We especially go into detail about proving these conditions. Recall that the mean value function and the intensity function of the Inflection S-Shaped model are given by

$$\Lambda(t) = a \frac{1 - e^{-bt}}{1 + \psi e^{-bt}}$$

and

$$\lambda(t) = \frac{ab(1 + \psi)e^{bt}}{(\psi + e^{bt})^2},$$

respectively.

3.5.1 Grouped Data

We first calculate $\Phi(b)$. If we insert (3.39) into (3.9) and simplify the expression, we get

$$\Phi(b) = \sum_{i=1}^{k} y_i \left[ -\ell_i - 1 + \frac{\ell_i - \ell_i - 1}{e^{b(\ell_i - \ell_i - 1)} - 1} + \frac{\ell_i - 1}{\psi + e^{b\ell_i - 1}} + \frac{\ell_i \psi}{e^{b\ell_i} + \psi} - \frac{\psi \ell_k}{e^{b\ell_k} + \psi} - \frac{\ell_k}{e^{b\ell_k} - 1} \right].$$

Again, $\Phi(b)$ no longer contains $a$ and cannot be written in closed form. We have to find the roots numerically. Now we state the condition which, assuming $\psi$ is known, has to be satisfied for $\Phi(b)$ to have at least one root.

**Theorem 3.14** In case of estimating the parameters $a$ and $b$ of the Inflection S-Shaped model using grouped data and $\psi > 1$ known, a sufficient condition for the existence of at least one, positive and finite solution of the Maximum Likelihood equations is given by

$$\sum_{i=1}^{k} y_i [\ell_i + \ell_i - 1] > n_k \ell_k.$$  \hfill (3.42)

If $\psi < 1$, the sufficient condition is given by

$$\sum_{i=1}^{k} y_i [\ell_i + \ell_i - 1] < n_k \ell_k.$$  \hfill (3.43)

**Proof:** To prove the theorem, we first show that $\lim_{b \to \infty} \Phi(b) < 0$. Then we show that if Theorem 3.15 holds $\lim_{b \to 0} \Phi(b) > 0$, which implies that $\Phi(b)$ has at least one positive root. It is easy to see that

$$\lim_{b \to \infty} \Phi(b) = -\sum_{i=1}^{k} y_i \ell_i - 1 < 0$$  \hfill (3.44)
and
\[ \lim_{b \downarrow 0} \Phi(b) = \sum_{i=1}^{k} y_i \left[ -\ell_{i-1} + \frac{1}{2} (\ell_k - (\ell_i - \ell_{i-1})) + \frac{\ell_{i-1}}{1 + \psi} + \frac{\ell_{i+1}}{1 + \psi} - \frac{\ell_k}{1 + \psi} \right]. \]

With the use of Lemma 3.3 we see that
\[ \lim_{b \downarrow 0} \Phi(b) = -\sum_{i=1}^{k} y_i \left[ \frac{(\psi - 1)(\ell_k - \ell_i - \ell_{i-1})}{2(\psi + 1)} \right]. \] (3.45)

In order for this expression to be positive, we have to distinguish between two different cases, \( \psi > 1 \) and \( \psi < 1 \). If \( \psi > 1 \), it is easy to see that (3.45) is positive if
\[ \sum_{i=1}^{k} y_i [\ell_k - \ell_i - \ell_{i-1}] < 0, \]
which implies
\[ \sum_{i=1}^{k} y_i [\ell_i + \ell_{i+1}] > n_k \ell_k. \]

Analogously, if \( \psi < 1 \), it is obvious that (3.45) is positive if
\[ \sum_{i=1}^{k} y_i [\ell_k - \ell_i - \ell_{i-1}] > 0, \]
which implies
\[ \sum_{i=1}^{k} y_i [\ell_i + \ell_{i-1}] < n_k \ell_k. \]

A condition that ensures \( \frac{\partial}{\partial b} \Phi(b) < 0 \) would be a sufficient, but not necessary condition, for \( \Phi(b) \) to have a unique root. If we calculate \( \frac{\partial}{\partial b} \Phi(b) \) we get
\[ \frac{\partial}{\partial b} \Phi(b) = -\sum_{i=1}^{k} y_i \left[ \psi e^{b_{i-1}} e^{b_{i-1}} b_{i-1} \left( e^{b_{i-1} + \psi} \right)^2 + \psi e^{b_{i-1}} e^{b_{i-1}} b_{i+1} \left( e^{b_{i+1} + \psi} \right)^2 + \left( \ell_i - \ell_{i-1} \right)^2 e^{b_{i-1}} e^{b_{i-1}} b_{i-1} - \left( e^{b_{i-1} + \psi} \right)^2 \right]. \]

If we want \( \frac{\partial}{\partial b} \Phi(b) < 0 \), this would imply that
\[ n_k \left[ \frac{e^{b_{i-1}} e^{b_{i-1}} b_{i-1}}{e^{b_{i-1} + \psi}^2} + \frac{e^{b_{i-1}} e^{b_{i-1}} b_{i+1}}{e^{b_{i+1} + \psi}^2} \right] < \sum_{i=1}^{k} y_i \left[ \psi e^{b_{i-1}} e^{b_{i-1}} b_{i-1} \left( e^{b_{i-1} + \psi} \right)^2 + \psi e^{b_{i-1}} e^{b_{i-1}} b_{i+1} \left( e^{b_{i+1} + \psi} \right)^2 + \left( \ell_i - \ell_{i-1} \right)^2 e^{b_{i-1}} e^{b_{i-1}} b_{i-1} - \left( e^{b_{i-1} + \psi} \right)^2 \right]. \]

We have not yet found a condition which guarantees this inequality and further research is needed. Now we study \( \Phi(\psi) \). If we insert (3.39) into (3.12), we get
\[ \Phi(\psi) = -\sum_{i=1}^{k} y_i \left( \frac{1}{\psi + e^{b_{i-1}}} + \frac{1}{e^{b_{i-1} + \psi}} - \frac{1}{e^{b_{i+1} + \psi}} - \frac{1}{1 + \psi} \right). \] (3.46)

It is easy to see that
\[ \lim_{\psi \to \infty} \Phi(\psi) = 0 \]
and
\[ \lim_{\psi \downarrow 0} \Phi(\psi) = \sum_{i=1}^{k} y_i \left[ 1 + \frac{1}{e^{b \ell_k}} - \frac{1}{e^{b \ell_{i-1}}} - \frac{1}{e^{b \ell_i}} \right]. \]

Several simulations have led us to believe that a necessary condition for \( \Phi(\psi) \) to have a root is \( \lim_{\psi \downarrow 0} \Phi(\psi) > 0 \). For \( \lim_{\psi \downarrow 0} \Phi(\psi) \) to be positive, the following condition has to be satisfied
\[ n_k \left( 1 + \frac{1}{e^{b \ell_k}} \right) > \sum_{i=1}^{k} y_i \left[ \frac{1}{e^{b \ell_{i-1}}} + \frac{1}{e^{b \ell_i}} \right]. \]

Unfortunately, this is not a sufficient condition. When \( \lim_{\psi \downarrow 0} \Phi(\psi) > 0 \), the function \( \Phi(\psi) \) may have the shape as depicted in Figures 3.1 and 3.2. Clearly, only the function shown in Figure 3.2 has a root. We have not found a condition which guarantees the existence of this root and further research is needed.

![Figure 3.1: Function \( \Phi(\psi) \) without root](image)

### 3.5.2 Ungrouped Data

First we study \( \Phi(b) \). If we insert (3.40) and (3.39) into (3.22) we get
\[ \Phi(b) = \sum_{i=1}^{n} \left[ \frac{1}{b - t_i} + 2 \frac{t_i \psi}{e^{b \ell_i} + \psi} - \frac{\psi t_n}{\psi + e^{b \ell_n}} - \frac{t_n}{e^{b \ell_n} - 1} \right]. \]  
(3.47)

The function \( \Phi(b) \) does not contain \( a \) and cannot be written in closed form. Thus, the roots have to be found numerically. We now state the condition which has to be satisfied for \( \Phi(b) \) to have at least one root, assuming that \( \psi \) is known.

**Theorem 3.15** In case of estimating the parameters \( a \) and \( b \) of the Inflection S-Shaped model using ungrouped data and \( \psi \) known, sufficient conditions for the existence of at least one,
positive and finite solution of the Maximum Likelihood equations are given by

\[ \sum_{i=1}^{n} t_i > \frac{nt_k}{2} \]

if \( \psi > 1 \), and

\[ \sum_{i=1}^{n} t_i < \frac{nt_k}{2} \]

if \( \psi < 1 \).

**Proof:** Calculating the limits of \( \Phi(b) \), gives us

\[ \lim_{b \to \infty} \Phi(b) = -\sum_{i=1}^{n} t_i \]  \hspace{1cm} (3.48)

and

\[ \lim_{b \downarrow 0} \Phi(b) = -\sum_{i=1}^{n} \frac{(\psi - 1)(t_k - 2t_i)}{2(\psi + 1)}. \]  \hspace{1cm} (3.49)

Because (3.48) is negative, we can say that, if (3.49) is positive, the function \( \Phi(b) \) has at least one root. We establish the condition for (3.49) to be positive for \( \psi > 1 \) and \( \psi < 1 \). If \( \psi > 1 \), it is easy to see that (3.49) is positive if

\[ -\sum_{i=1}^{n} t_k - 2t_i > 0, \]

which implies

\[ \sum_{i=1}^{n} t_i > \frac{nt_k}{2}. \]
If $\psi < 1$, it is easy to see that (3.49) is positive if

$$-\sum_{i=1}^{n} t_k - 2t_i > 0,$$

which implies

$$\sum_{i=1}^{n} t_i < \frac{nt_k}{2}.$$  \hfill $\Box$

Calculating the derivative of $\Phi(b)$ with respect to $b$, we obtain

$$\frac{\partial}{\partial b} \Phi(b) = \sum_{i=1}^{n} \left[ -\frac{1}{b^2} - \frac{2e^{bt_i}\psi t_i^2}{(e^{bt_i} + \psi)^2} + \frac{e^{bt_k}t_k^2}{(-1 + e^{bt_k})^2} + \frac{e^{bt_k}\psi t_k^2}{(e^{bt_k} + \psi)^2} \right].$$  \hfill (3.50)

We can guarantee a single root if $\frac{\partial}{\partial b} \Phi(b) < 0$. This would imply that

$$\frac{ne^{bt_n}t_n^2}{(-1 + e^{bt_n})^2} + \frac{ne^{bt_n}\psi t_n^2}{(e^{bt_n} + \psi)^2} - \frac{n}{b^2} < \sum_{i=1}^{n} \frac{2e^{bt_i}\psi t_i^2}{(e^{bt_i} + \psi)^2}.$$  \hfill (3.51)

We have not found a condition which guarantees this inequality and more research is needed. Now we study $\Phi(\psi)$. If we insert (3.40) and (3.39) into (3.23) we get

$$\Phi(\psi) = \sum_{i=1}^{n} \left[ \frac{1}{1 + \psi} - \frac{2}{\psi + e^{bt_i}} + \frac{1}{\psi + e^{bt_k}} \right].$$  \hfill (3.52)

If we calculate the limits we get

$$\lim_{\psi \downarrow 0} \Phi(\psi) = \sum_{i=1}^{n} \left[ 1 - \frac{2}{e^{bt_i}} + \frac{1}{e^{bt_k}} \right]$$  \hfill (3.53)

and

$$\lim_{\psi \to \infty} \Phi(\psi) = 0.$$  \hfill (3.54)

Again, simulations indicate that a necessary condition for $\Phi(\psi)$ to have a root is

$$\lim_{\psi \downarrow 0} \Phi(\psi) > 0.$$  \hfill \hfill (3.55)

This would imply

$$n(1 + \frac{1}{e^{bt_k}}) > \sum_{i=1}^{n} \frac{2}{e^{bt_i}}.$$  \hfill (3.55)

Unfortunately, this is not a sufficient condition. When $\lim_{\psi \downarrow 0} \Phi(\psi) > 0$, the function $\Phi(\psi)$ may have the shape as depicted in Figures 3.1 and 3.2. Clearly, only the function shown in Figure 3.2 has a root. We have not found a condition which guarantees this root and further research is needed.
Chapter 4

Recursive Method

Conventional estimation methods based on fixed size samples, such as the Maximum Likelihood Method, use calculations involving the complete data set. This is in contrast to an approach based on sequential methods. Such methods, also known as stochastic recursive identification algorithms, allow updating of the parameter estimates while using only the last few observations. Recursive algorithms take many forms and have numerous applications in biomedicine, economics and engineering, beginning with the seminal paper of Robbins and Monro (1951). In this chapter we present a recursive algorithm to estimate the unknown model parameters inspired by a recursive application of the Maximum Likelihood Method. Several simulations have shown that, often, this algorithm provides reasonable estimates in situations when the software failure data does not satisfy the conditions for the Maximum Likelihood Method to have a solution or exhibits a very unstable behaviour. In Section 4.1 we give a general description of the method. Note that the general description applies to grouped data. The application of the method to ungrouped data appears to be too complex, but we do describe concisely in Section 4.5 how to apply the method to ungrouped data. The general description of the method is followed by the application to the following models: the Goel-Okumoto model (Section 4.2) and the Yamada S-Shaped model (Section 4.3).

4.1 General Recursive Method

In this section we give a general description of the recursive algorithm. The algorithm is used to estimate the unknown model parameters which are represented by a vector $\theta$. The algorithm uses the last $m$ observations to update the estimations. The random vector containing the last $m$ observations, at time moment $i$, is denoted by $Z_i$. The realisation of $Z_i$ is denoted by $z_i$ and the estimation of $\theta$ at iteration $i$ by $\hat{\theta}_i$. The general Robbins-Monro recursive algorithm is as follows

$$\hat{\theta}_{i+1} = \hat{\theta}_i + \gamma_i G_i(z_i, \hat{\theta}_i),$$

(4.1)

with some initial $\hat{\theta}_0$ and a gain vector $G_i(z_i, \hat{\theta}_i)$. In this equality, $\gamma_i$ represents a positive, non-increasing sequence which satisfies the following conditions

$$\sum_{i=1}^{\infty} \gamma_i = \infty,$$

(4.2)
\[ \sum_{i=1}^{\infty} \gamma_i^2 < \infty. \] (4.3)

The gain vector is a \( \mathbb{R}^d \) valued vector which is supposed to shift \( \hat{\theta}_k \) towards the true value of \( \theta \), at least on average. This can be provided under certain conditions. Such conditions and various examples of gain vectors can be found in Belitser and Serra (2012), where they also study the convergence of recursive algorithms as the “information” in the model increases.

The gain vector we use originates from Fisher’s scoring algorithm, which is a form of the Newton-Raphson method used to solve Maximum Likelihood equations numerically. The recursion of Fisher’s scoring algorithm is as follows

\[ \hat{\theta}_{m+1} = \hat{\theta}_m + I^{-1}(z, \hat{\theta}_m) \nabla l(z, \hat{\theta}_m). \]

Function \( l(z, \theta) \) represents the log-likelihood function of \( z \). Fisher’s scoring algorithm is not recursive in time as both functions \( I(z, \theta) \) and \( \nabla l(z, \theta) \) are based on the complete data set \( z = (z_1, \ldots, z_m) \) whereas we want our algorithm to be recursive in time. This leads to the following choice of gain vector

\[ B_i(z_i, \theta) = I^{-1}(z_i, \theta) \nabla l_i(z_i, \theta). \] (4.4)

Here \( z_i \) denotes the last \( m \) observations and \( l_i(z_i, \theta) \) and \( I(z_i, \theta) \) the corresponding log-likelihood function and Fisher Information Matrix, respectively. The number of observations that have to be used to update the estimations, depends on the number of parameters that are being estimated. Clearly, the number of observations should at least be as big as the number of parameters, otherwise, the parameters are not identifiable and the Fisher Information Matrix not invertible. Using (4.4) and the appropriate number of observations, our recursive formula can be written as

\[ \hat{\theta}_{i+1} = \hat{\theta}_i + \gamma_i I^{-1}(z_i, \hat{\theta}_i) \nabla l(z_i, \hat{\theta}_i). \]

Notice that, besides the fact that we only use the last \( m \) observations, our recursive formula also differs from Fisher’s scoring algorithm by adding a non-increasing sequence to the recursive formula. The non-increasing sequence we use in our algorithm is defined by \( \gamma_i = \frac{C}{p^r} \).

Because the sequence has to satisfy (4.2) and (4.3), we take \( \frac{1}{2} \leq p \leq 1 \). In this report we do not elaborate on how to choose the constants \( p \) and \( C \) because on this aspect of the algorithm, further research is needed.

4.1.1 Separate Estimation

The recursive method is able to provide an estimate for the whole vector of parameters, i.e., it estimates all parameters simultaneously. We refer to this approach as simultaneous estimation. This approach can however be computationally demanding, especially if the dimension of the parameter vector is relatively high. Below we discuss a version of the method for estimating each parameter separately. We refer to this approach as separate estimation. In case of separate estimation we construct a gain vector for each of the parameters we want to
estimate. Then, with each recursion, we apply the gain vectors to each corresponding parameter, while we update the parameters used in the gain vectors with the previous estimations. Mathematically it can be described as follows. Say, we want to estimate two parameters, $a$ and $b$, and for both parameters we have an initial estimation, $a_0$ and $b_0$. Then, we calculate the new parameters according to

$$a_{k+1} = a_k + \gamma_k G_k(y_k, a_k, b_k)$$

and

$$b_{k+1} = b_k + \gamma_k H_k(y_k, a_k, b_k).$$

Here, $G_k$ and $H_k$ are both gain vectors. A benefit of separate estimation, compared with simultaneous estimation, is that it reduces the complexity of the computations. If we want to estimate a one-dimensional parameter $\theta$, the gain vector as given in (4.4) simplifies to

$$B_i(y_i, \theta) = \left(-E\left[\frac{\partial^2}{\partial \theta^2} l_i(y_i, \theta)\right]\right)^{-1} \left(\frac{\partial}{\partial \theta} l_i(y_i, \theta)\right).$$

(4.5)

Notice that the gradient and the Fisher Information Matrix of $l_i(z_i, \theta)$ reduces to the first derivative and the expectation of the second derivative, respectively. Also note that, $z_i$ is replaced by $y_i$. In case of estimating one parameter at the time, we only need the last data point to update the estimation. We now derive the explicit recursion formulas for the Goel-Okumoto model and Yamada S-Shaped model in case of separate estimation. To calculate the gain vector we first need the log-likelihood function of the last data point. We use the log-likelihood function as described in Section 3, formula (3.4). Rewriting this formula to the log-likelihood function of the last data point gives us

$$l_i(y_i, \theta) = y_i \ln (a) + y_i \ln \left(\bar{\Lambda}(\ell_i) - \bar{\Lambda}(\ell_{i-1})\right) - \ln (y_i!) - a \bar{\mu}(\ell_i)$$

$$= y_i \ln (a) + y_i \ln (\bar{\mu}(\ell_i)) - \ln (y_i!) - a \bar{\mu}(\ell_i).$$

To calculate the gain vector for the estimation of $a$, we first determine $-E\left[\frac{\partial^2}{\partial a^2} l_i(y_i, \theta)\right]$. We calculate

$$-E\left[\frac{\partial^2}{\partial a^2} l_i(y_i, \theta)\right] = E\left[\frac{y_i}{a^2}\right].$$

Because $E[y_i] = \mu(\ell_i)$ we are able to write

$$-E\left[\frac{\partial^2}{\partial a^2} l_i(y_i, \theta)\right] = \mu(\ell_i) a^2. \quad (4.6)$$

Then we calculate the gradient

$$\frac{\partial}{\partial a} l_i(y_i, \theta) = \frac{y_i - \mu(\ell_i)}{a}.\quad (4.6)$$
Now we can calculate the gain vector the following way

\[
B_i(y_i, \theta) = \left( -E \left[ \frac{\partial^2}{\partial a^2} l_i(y_i, \theta) \right] \right)^{-1} \left( \frac{\partial}{\partial a} l_i(y_i, \theta) \right)
\]

\[
= (\frac{\mu(\ell_i)}{\sigma^2})^{-1} \left( \frac{y_i - \mu(\ell_i)}{\sigma} \right)
\]

\[
= \left( \frac{a}{\mu(\ell_i)} \right) \left( \frac{y_i - \mu(\ell_i)}{\sigma} \right)
\]

\[
= \frac{y_i - \mu(\ell_i)}{\mu(\ell_i)}.
\]

As recursion for the estimation of \( a \) we get

\[
a_{i+1} = a_i + \gamma_i B_i(y_i, \theta)
\]

\[
= a_i + \gamma_i \frac{y_i - \mu(\ell_i)}{\mu(\ell_i)}.
\]

To establish the gain vector for the estimation of \( b \), we first determine

\[
- E \left[ \frac{\partial^2}{\partial b^2} l_i(y_i, \theta) \right].
\]

We calculate

\[
- E \left[ \frac{\partial^2}{\partial b^2} l_i(y_i, \theta) \right] = - E \left[ y_i \left( \frac{\partial^2}{\partial b^2} \mu(\ell_i) \right) \mu(\ell_i) - \left( \frac{\partial}{\partial b} \mu(\ell_i) \right)^2 \right] - \frac{\partial^2}{\partial b^2} \mu(\ell_i)
\]

\[
= - \mu(\ell_i) \left( \frac{\partial^2}{\partial b^2} \mu(\ell_i) \right) \mu(\ell_i) - \left( \frac{\partial}{\partial b} \mu(\ell_i) \right)^2 + \frac{\partial^2}{\partial b^2} \mu(\ell_i)
\]

which implies

\[
- E \left[ \frac{\partial^2}{\partial b^2} l_i(y_i, \theta) \right] = \left( \frac{\partial}{\partial b} \mu(\ell_i) \right)^2.
\]

(4.7)

Then we calculate the gradient

\[
\frac{\partial}{\partial b} l_i(y_i, \theta) = y_i \frac{\partial}{\partial b} \mu(\ell_i) - \frac{\partial}{\partial b} \mu(\ell_i).
\]

Now we can calculate the gain vector the following way

\[
B_i(y_i, \theta) = \left( -E \left[ \frac{\partial^2}{\partial b^2} l_i(y_i, \theta) \right] \right)^{-1} \left( \frac{\partial}{\partial b} l_i(y_i, \theta) \right)
\]

\[
= \left( \frac{\mu(\ell_i)}{\partial^2 \mu(\ell_i)} \right) \left( y_i \frac{\partial}{\partial b} \mu(\ell_i) - \frac{\partial}{\partial b} \mu(\ell_i) \right)
\]

\[
= \frac{y_i - \mu(\ell_i)}{\partial^2 \mu(\ell_i)}.
\]
As recursion for the estimation of $b$ we get
\[ b_{i+1} = b_i + \gamma_i \beta_i(y_i, \theta_i) \]
\[ = b_i + \gamma_i \frac{y_i - \mu(\ell_i)}{\bar{\mu}(\ell_i)}. \]

For the remainder of this chapter we do not explicitly derive the gain vectors for each model. Instead, we always refer to the following recursion formulas:
\[ a_{i+1} = a_i + \gamma_i \frac{y_i - \mu(\ell_i)}{\bar{\mu}(\ell_i)} \tag{4.8} \]
\[ b_{i+1} = b_i + \gamma_i \frac{y_i - \mu(\ell_i)}{\partial b \mu(\ell_i)} \tag{4.9} \]

### 4.1.2 Simultaneous Estimation

When we want to estimate two parameters simultaneously, we have to use at least the last two data points. We thus put $z_i = (y_{i-1}, y_i)$. Because of the independence of the number of events in disjoint intervals, the likelihood function looks as follows: $P(z_i) = P(y_{i-1})P(y_i)$. This implies that the log-likelihood function can be written as $l(z_i, \theta) = l(y_{i-1}, \theta) + l(y_i, \theta)$. This makes it easy to calculate the following entries of the Fisher Information Matrix of $l(z_i, \theta)$. Using (4.6), we compute
\[ -E \left[ \frac{\partial^2}{\partial a^2} l(z_i, \theta) \right] = -E \left[ \frac{\partial^2}{\partial a^2} l(y_{i-1}, \theta) \right] - E \left[ \frac{\partial^2}{\partial a^2} l(y_i, \theta) \right] \]
\[ = \frac{\bar{\mu}(\ell_i)}{a} + \frac{\mu(\ell_i)}{a}. \]

Using (4.7), we write
\[ -E \left[ \frac{\partial^2}{\partial b^2} l(z_i, \theta) \right] = -E \left[ \frac{\partial^2}{\partial b^2} l(y_{i-1}, \theta) \right] - E \left[ \frac{\partial^2}{\partial b^2} l(y_i, \theta) \right] \]
\[ = \frac{\left( \frac{\partial \mu(\ell_i)}{\partial b} \right)^2}{\mu(\ell_{i-1})} + \frac{\left( \frac{\partial \mu(\ell_i)}{\partial b} \right)^2}{\mu(\ell_i)}. \]

Using some simple calculations, we calculate
\[ -E \left[ \frac{\partial^2}{\partial a \partial b} l(z_i, \theta) \right] = (-E \left[ \frac{\partial^2}{\partial a \partial b} l(y_{i-1}, \theta) \right]) + (-E \left[ \frac{\partial^2}{\partial a \partial b} l(y_i, \theta) \right]) \]
\[ = \frac{\partial \mu(\ell_{i-1})}{\partial b} + \frac{\partial \mu(\ell_i)}{\partial b}. \]

Now we can construct the Fisher Information Matrix the following way
\[
I(z_i, \theta) = \begin{pmatrix}
-E \left[ \frac{\partial^2 l(z_i, \theta)}{\partial a^2} \right] & -E \left[ \frac{\partial^2 l(z_i, \theta)}{\partial a \partial b} \right] \\
-E \left[ \frac{\partial^2 l(z_i, \theta)}{\partial b^2} \right] & -E \left[ \frac{\partial^2 l(z_i, \theta)}{\partial b^2} \right]
\end{pmatrix}.
\]
The inverse of this matrix can be calculated with

\[ I^{-1}(z_i, \theta) = \frac{1}{\det(I(z_i, \theta))} \begin{pmatrix} -E \frac{\partial^2 l(z_i, \theta)}{\partial b^2} & E \frac{\partial^2 l(z_i, \theta)}{\partial a \partial b} \\ E \frac{\partial^2 l(z_i, \theta)}{\partial a \partial b} & -E \frac{\partial^2 l(z_i, \theta)}{\partial a^2} \end{pmatrix} \].

Notice that we are able to write

\[ I^{-1}(z_i, \theta) = \frac{1}{\det(I(z_i, \theta))} \begin{pmatrix} -E \frac{\partial^2 l(y_{i-1}, \theta)}{\partial b^2} & E \frac{\partial^2 l(y_{i-1}, \theta)}{\partial a \partial b} \\ E \frac{\partial^2 l(y_{i-1}, \theta)}{\partial a \partial b} & -E \frac{\partial^2 l(y_{i-1}, \theta)}{\partial a^2} \end{pmatrix} \\
+ \frac{1}{\det(I(z_i, \theta))} \begin{pmatrix} -E \frac{\partial^2 l(y_i, \theta)}{\partial b^2} & E \frac{\partial^2 l(y_i, \theta)}{\partial a \partial b} \\ E \frac{\partial^2 l(y_i, \theta)}{\partial a \partial b} & -E \frac{\partial^2 l(y_i, \theta)}{\partial a^2} \end{pmatrix} \]

If we define

\[ I(y_i, \theta) = \begin{pmatrix} -E \frac{\partial^2 l(y_i, \theta)}{\partial a^2} & -E \frac{\partial^2 l(y_i, \theta)}{\partial a \partial b} \\ -E \frac{\partial^2 l(y_i, \theta)}{\partial a \partial b} & -E \frac{\partial^2 l(y_i, \theta)}{\partial b^2} \end{pmatrix}, \]

we can rewrite the expansion of \( I^{-1}(z_i, \theta) \) as follows:

\[ I^{-1}(z_i, \theta) = \frac{\det(I(y_i, \theta))}{\det(I(z_i, \theta))} I^{-1}(y_i, \theta) + \frac{\det(I(y_{i-1}, \theta))}{\det(I(z_i, \theta))} I^{-1}(y_{i-1}, \theta). \]

This expansion of \( I^{-1}(z_i, \theta) \) is useful in the final step of calculating the gain vector. We first continue with calculating the gradient of the log-likelihood function. It is easy to see that

\[ \nabla l(z_i, \theta) = \nabla l_{i-1}(y_{i-1}, \theta) + \nabla l_i(y_i, \theta), \]

where

\[ \nabla l_j(y_j, \theta) = \left( \frac{y_j - \mu(\ell_j)}{y_j} \frac{\partial}{\partial \mu} \mu(\ell_j) - \frac{\partial}{\partial \mu} \mu(\ell_j) \right). \]

A simple multiplication shows that for \( j = i \) and \( j = i - 1 \) we have

\[ I^{-1}(y_j, \theta) \nabla l_j(y_j, \theta) = 0. \]

Thus, we get the following expression for the gain vector

\[ B(z_i, \theta) = \frac{\det(I(y_i, \theta))}{\det(I(z_i, \theta))} I^{-1}(y_i, \theta) \nabla l_{i-1}(y_{i-1}, \theta) + \frac{\det(I(y_{i-1}, \theta))}{\det(I(z_i, \theta))} I^{-1}(y_{i-1}, \theta) \nabla l_i(y_i, \theta). \]  \hspace{1cm} (4.10)

Looking closely to this expression, we can see that it resembles the sum of two gain vectors. When we compute \( B(z_i, \theta) \) for a specific model, we only derive the first part of this sum. The second part can be obtained by adjusting the indexes in the appropriate way.
4.2 Goel-Okumoto Model

In this section we apply the recursive method to the Goel-Okumoto model. We start with discussing separate estimation in Subsection 4.2.1 and continue with simultaneous estimation in Subsection 4.2.2. In Section 4.4 we give examples of applying the recursive method to simulated data. In Section 4.4 we give graphs of both separate and simultaneous recursive estimation of the parameters $a$ and $b$.

4.2.1 Separate Estimation

To calculate the gain vectors to estimate $a$ and $b$, we use (4.8) and (4.9). We get the following recursive formulas:

$$a_{k+1} = a_k - \gamma \left( a_k - \frac{e^{b_k(\ell_k+\ell_{k-1})}y_k}{e^{b_k\ell_k} - e^{b_k\ell_{k-1}}} \right)$$

$$= a_k (1 - \gamma) + \gamma \frac{e^{b_k(\ell_k+\ell_{k-1})}y_k}{e^{b_k\ell_k} - e^{b_k\ell_{k-1}}}$$

and

$$b_{k+1} = b_k + \gamma \left( -a_k + \frac{y_k}{e^{-b_k\ell_{-1+k}} (1 + b_k\ell_{-1+k}) - e^{-b_k\ell_k} (1 + b_k\ell_k)} \right).$$

The application of the recursive formula for the estimation of $a$ is show in Figures 4.1 and 4.2. The application of the recursive formula for the estimation of $b$ is shown in Figures 4.3 and 4.4.

4.2.2 Simultaneous Estimation

First we calculate $I(y_i, \theta)$:

$$I(y_i, \theta) = \left( \begin{array}{ccc} \frac{e^{-b_i\ell_{i-1}} - e^{-b_i\ell_i}}{a} & -e^{-b_i\ell_{i-1}}\ell_{i-1} + e^{-b_i\ell_i} \ell_i & ae^{-b_i\ell_{i-1+i}}(e^{b_i\ell_{i-1}} - e^{b_i\ell_{i-1+i}})^2 \\ -e^{-b_i\ell_{i-1}}\ell_{i-1} - e^{-b_i\ell_i} & -e^{-b_i\ell_{i-1}}\ell_{i-1} + e^{-b_i\ell_i} & \frac{e^{-b_i\ell_{i-1}} - e^{-b_i\ell_i}}{a} \end{array} \right).$$

Next, we calculate $\nabla l_i(y_i, \theta)$:

$$\nabla l_i(y_i, \theta) = \left( \begin{array}{cccc} -e^{-b_i\ell_{i-1}} + e^{-b_i\ell_i} & -e^{-b_i\ell_{i-1}} + e^{-b_i\ell_i} & \frac{y_i(e^{b_i\ell_{i-1}} - e^{b_i\ell_{i-1+i}})}{e^{b_i\ell_{i-1}}} + a(e^{-b_i\ell_{i-1}}\ell_{i-1} - e^{-b_i\ell_i}) \\ \end{array} \right).$$

The above expressions also provide formulas for $I(y_{i-1}, \theta)$ and $\nabla l_{i-1}(y_{i-1}, \theta)$. Now we can calculate the gain vector by using (4.10). Because of the complexity of this expression, we do not state it here. The application of the recursive formula for the estimation of $a$ is shown in Figures 4.5 and 4.6. The application of the gain vector for the estimation of $b$ is shown in Figures 4.7 and 4.8.
4.3 Yamada S-Shaped Model

In this section we apply the recursive method to the Yamada S-Shaped model. We start with discussing separate estimation in Subsection 4.3.1 and continue with simultaneous estimation in Subsection 4.3.2. In Section 4.4 we give examples of applying the recursive method to simulated data. In Section 4.4 we give graphs of both separate estimation and simultaneous estimation of the parameters $a$ and $b$.

4.3.1 Separate Estimation

First we determine the gain vectors by using (4.8) and (4.9). Using these formulas we get the following recursions:

\[ a_{k+1} = a_k - \gamma \left( a_k + \frac{y_i}{e^{-bl_{i-1}} (1 + bl_{i-1})} - e^{-bl_i} (1 + bl_i) \right) \]

\[ = a_k (1 - \gamma) + \gamma \frac{y_i}{e^{-bl_{i-1}} (1 + bl_{i-1})} - e^{-bl_i} (1 + bl_i) \]

and

\[ b_{k+1} = b_k + \gamma \frac{e^{b(l_{i-1}+\ell_i)} y_i - ae^{bl_i} (1 + bl_{i-1}) + ae^{bl_{i-1}} (1 + bl_i)}{ab \left( -e^{bl_{i-1}l^2_i} + e^{bl_i l^2_i} \right)} \]

The application of the recursive formula for the estimation of $a$ is shown in Figures 4.9 and 4.10. The application of the recursive formula for the estimation of $b$ is shown in Figures 4.11 and 4.12.

4.3.2 Simultaneous Estimation

First we calculate $I(y_i, \theta)$:

\[ I(y_i, \theta) = \left( \frac{e^{-bl_i (1+bl_{i-1})} - e^{-bl_{i-1}} (1+bl_i)}{a} \right) \left( \frac{y_i}{-e^{-bl_{i-1}l^2_i} + e^{bl_i l^2_i}} \right) \]

Next we calculate $\nabla l_i(y_i, \theta)$:

\[ \nabla l_i(y_i, \theta) = \left( \frac{y_i}{b \left( e^{bl_i l^2_i} - e^{bl_{i-1} l^2_{i-1}} \right)} \left( \frac{ae^{b(l_{i-1}+\ell_i)} y_i}{e^{bl_i (1+bl_{i-1})} - e^{bl_{i-1} (1+bl_i)}} \right) \right) \]

The above expressions also provide formulas for $I(y_{i-1}, \theta)$ and $\nabla l_{i-1}(y_{i-1}, \theta)$. Now we can calculate the gain vector by using (4.10). Because of the complexity of this expression we do not state it here. The application of the gain vector for the estimation of $a$ is shown in Figures 4.13 and 4.14. The application of the gain vector for the estimation of $b$ is shown in Figures 4.15 and 4.16.


4.4 Examples

In this section we illustrate the recursive method by showing plots of the estimation process. We also compare the process with the Maximum Likelihood Method. In the graphs, the Maximum Likelihood Method is depicted by the dashed line, the recursive method by the thin line and the thick, straight line is the real value of the parameter. In these examples, we take $a_0 = 2500$ and $b_0 = 0.005$. We simulated 100 data points and take $C = 1000$ and $p = 1$. The true value of the parameters are $a = 5000$ and $b = 0.01$. Note that the figures on the right-hand side of the page are close ups of the line representing the recursive method of the graph on the left.
4.4.1 Goel-Okumoto Model

Figures 4.1-4.4: Separate Estimation Parameter
Figure 4.5: Simultaneous Estimation Parameter $a$

Figure 4.6: Simultaneous Estimation Parameter $a$

Figure 4.7: Simultaneous Estimation Parameter $b$

Figure 4.8: Simultaneous Estimation Parameter $b$
4.4.2 Yamada S-Shaped Model

Figure 4.9: Separate Estimation Parameter $a$

Figure 4.10: Separate Estimation Parameter $a$

Figure 4.11: Separate Estimation Parameter $b$

Figure 4.12: Separate Estimation Parameter $b$
Figure 4.13: Simultaneous Estimation Parameter $a$

Figure 4.14: Simultaneous Estimation Parameter $a$

Figure 4.15: Simultaneous Estimation Parameter $b$

Figure 4.16: Simultaneous Estimation Parameter $b$
4.5 Ungrouped Data

In this section we briefly discuss the recursive method for ungrouped data. In case of simultaneous estimation of the parameters, the gain vectors are too complex to calculate. This is why we only give a brief note on simultaneous estimation of the parameters. Even in case of separate estimation, we were not able to write explicit expressions for all gain vectors. This is why, also in case of separate estimation of the parameters, we only give a short description and we do not go into detail about each model separately.

4.5.1 Separate Estimation

In case of using ungrouped data, the gain vector to estimate parameter $a$ is described by

$$B_i(t_i, \theta) = \left( -E\left[ \frac{\partial^2}{\partial a^2}l(t_i, \theta) \right] \right)^{-1} \left( \frac{\partial}{\partial a} l(t_i, \theta) \right).$$

(4.11)

To calculate this expression, we use the log-likelihood function for ungrouped data as described in Section 3, Formula (3.17). Rewriting this formula to the log-likelihood function of the last data point gives us

$$l(t_i, \theta) = \ln(\lambda(t_i))) - (\Lambda(t_i) - \Lambda(t_{i-1}))$$

$$= \ln(a + \ln(\lambda(t_i))) - a(\Lambda(t_i) - \Lambda(t_{i-1})).$$

To calculate the gain vector for the estimation of $a$, we first determine $-E\left[ \frac{\partial^2}{\partial a^2} l(t_i, \theta) \right]$:

$$-E\left[ \frac{\partial^2}{\partial a^2} l(t_i, \theta) \right] = E\left[ \frac{1}{a^2} \right]$$

$$= \frac{1}{a^2}.$$

Then we calculate the gradient

$$\frac{\partial}{\partial a} l(t_i, \theta) = \frac{1}{a} - (\bar{\Lambda}(t_i) - \bar{\Lambda}(t_{i-1})).$$

Now we can calculate the gain vector in the following way

$$B_i(t_i, \theta) = \left( -E\left[ \frac{\partial^2}{\partial a^2} l(t_i, \theta) \right] \right)^{-1} \left( \frac{\partial}{\partial a} l(t_i, \theta) \right)$$

$$= \left( \frac{1}{a^2} \right)^{-1} \left( \frac{1}{a} - (\bar{\Lambda}(t_i) - \bar{\Lambda}(t_{i-1})) \right)$$

$$= a - a(\Lambda(t_i) - \Lambda(t_{i-1})).$$

As recursion for the estimation of $a$, we get

$$a_{i+1} = a_i + \gamma_i B_i(t_i, \theta_i)$$

$$= a_i + \gamma_i (a_i - a_i(\Lambda(t_i) - \Lambda(t_{i-1})))$$

$$= a_i(1 + \gamma_i (1 - (\Lambda(t_i) - \Lambda(t_{i-1}))).$$
The gain vector to estimate parameter $b$ is described by

$$B_i(t_i, \theta) = \left( -E[\partial^2 b^2 l(t_i, \theta)] \right)^{-1} \left( \frac{\partial}{\partial b} l(t_i, \theta) \right).$$ (4.12)

To calculate this gain vector, we first determine $-E[\partial^2 b^2 l(t_i, \theta)]$:

$$-E[\partial^2 b^2 l(t_i, \theta)] = E[\partial^2 b^2 (\Lambda(t_i) - \Lambda(t_{i-1}) - \ln(\lambda(t_i)))]
= \int_{t_{i-1}}^{\infty} E[\partial^2 b^2 (\Lambda(t_i) - \Lambda(t_{i-1}) - \ln(\lambda(t_i)))] \lambda(t_i) e^{-(\Lambda(t_i) - \Lambda(t_{i-1}))} dt_i.$$

Unfortunately, for the models we study in this report, this expression is too complex to calculate. Thus, we cannot give an explicit expression for the gain vector for the estimation of $b$.

### 4.5.2 Simultaneous Estimation

We would like to end this section with a quick note on the simultaneous estimation of the parameters. In case of estimating the parameter simultaneously, to calculate the gain vector, we need to calculate the Fisher Information Matrix. If we let $\theta_l$ and $\theta_m$ be either $a$ or $b$, this requires us to calculate

$$E \left[ \partial^2 \theta_l \theta_m \log(l(a, b)) \right] = E[g(t_i, t_i)]
= \int_0^\infty \int_0^\infty g(t_{i-1}, t_i) f_{T_{i-1}, T_i}(t_{i-1}, t_i).$$

In this equation $f_{T_{i-1}, T_i}(t_{i-1}, t_i)$ is the two-dimensional probability density function, which represents the density of two consecutive events on $t_{i-1}$ and $t_i$. Unfortunately, we were not able to write this function in closed form. Thus we could not calculate the Fisher Information Matrix needed to construct the gain vector. Further research is needed to calculate the gain vector in case of simultaneous estimation using ungrouped data.
Chapter 5

Conclusions

In this report we studied two different methods to estimate parameters of software reliability models: the Maximum Likelihood Method and a recursive estimation method. To provide a clear picture of the conclusions we draw from each method, we divide this chapter into two sections. Conclusions which can be drawn from Chapter 3, concerning the Maximum Likelihood Method, can be found in Section 5.1. Conclusions which can be drawn from Chapter 4, concerning the recursive method, can be found in Section 5.2.

5.1 Maximum Likelihood Method

In Chapter 3 we studied the Maximum Likelihood Method. We especially went into details about the conditions which the software failure data has to satisfy, for the Maximum Likelihood equations to have a solution. We have shown that, in order to estimate the parameters $a$ and $b$, it is sufficient to estimate $b$. This is because $a$ can be expressed as a function of $b$, as shown in (5.1).

$$a = \frac{n}{\Lambda(t)}$$  \hspace{1cm} (5.1)

In this expression $n$ represents the total number of faults detected at time $t$. We also have shown that, for a reasonable estimate of $b$, this function returns a reasonable estimate of $a$. This implies that, if $b$ is unique, positive and finite, then the expression for $a$ is unique, positive and finite. Also 5.1 will never have a singularity or be less than the number of faults which are already detected. To estimate $b$ we used the following function

$$\Phi(b) = \frac{\partial}{\partial b} \log(L(x)),$$

where $L(x)$ is defined as the likelihood function of the observed failure data. The root of $\Phi(b)$ is used as an estimate for $b$. A similar function, $\Phi(\psi)$, is used to estimate parameter $\psi$ of the Inflection S-Shaped model. In case of the Goel-Okumoto Model and the Yamada S-Shaped Model we have derived the conditions which assures that $\Phi(b)$ has a unique, positive and finite root. These conditions are as follows:

<table>
<thead>
<tr>
<th>Model</th>
<th>Grouped data</th>
<th>Ungrouped data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goel-Okumoto Model</td>
<td>$n_k \ell_k &gt; \sum_{i=1}^{k} y_i (\ell_i + \ell_{i-1})$</td>
<td>$t_n &gt; \frac{2}{n} \sum_{i=1}^{n} t_i$</td>
</tr>
<tr>
<td>Yamada S-Shaped Model</td>
<td>$n_k \ell_k &gt; \sum_{i=1}^{k} y_i [\ell_i + \frac{\ell_i^2}{\ell_{i-1} + \ell_i}]$</td>
<td>$t_n &gt; \frac{3}{2n} \sum_{i=1}^{n} t_i$</td>
</tr>
</tbody>
</table>
In case of estimating the parameters \( a \) and \( b \) of the Inflection S-Shaped model and \( \psi \) known, we have shown that the sufficient set of conditions for the existence of at least one, positive and finite root of \( \Phi(b) \) is given by the following inequalities

\[
\begin{align*}
\psi < 1 & \quad \ell_k n_k > \sum_{i=1}^{k} y_i (\ell_i + \ell_{i-1}) \\
\psi > 1 & \quad \ell_k n_k < \sum_{i=1}^{k} y_i (\ell_i + \ell_{i-1})
\end{align*}
\]

A condition which assures that \( \frac{\partial}{\partial b} \Phi(b) < 0 \), would be a sufficient, but not a necessary condition, for a unique root of \( \Phi(b) \). Unfortunately, we were not able to find this condition. As estimate of \( \psi \) we use the root of \( \Phi(\psi) \). Several simulations have led us to believe that \( \lim_{\psi \to 0} \Phi(\psi) > 0 \) is a necessary, but not sufficient condition for \( \Phi(\psi) \) to have a root. Unfortunately, we were not able to find this condition.

In case of the Goel-Okumoto Model and the Yamada S-Shaped Model, we also made a connection between the conditions which assure a reasonable estimate and trends in the software failure data. The trend we considered was an increase in reliability. The implications of reliability growth and whether \( \Phi(b) \) has a root can be found in the table below. Note that “Root \( \rightarrow \) Reliability Growth” only states that it is reasonable to assume reliability growth, but that it may not be the case.

<table>
<thead>
<tr>
<th></th>
<th>Grouped data</th>
<th>Ungrouped data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goel-Okumoto</td>
<td>No Root ( \leftrightarrow ) No Reliability Growth</td>
<td>No Root ( \leftrightarrow ) No Reliability Growth</td>
</tr>
<tr>
<td></td>
<td>Root ( \leftrightarrow ) Reliability Growth</td>
<td>Root ( \rightarrow ) Reliability Growth</td>
</tr>
<tr>
<td>Yamada S-Shaped</td>
<td>No Root ( \rightarrow ) No Reliability Growth</td>
<td>No Root ( \rightarrow ) No Reliability Growth</td>
</tr>
<tr>
<td></td>
<td>Root ( \leftrightarrow ) Reliability Growth</td>
<td>Root ( \leftrightarrow ) Reliability Growth</td>
</tr>
</tbody>
</table>

### 5.2 Recursive Method

In Chapter 4 we studied a recursive estimation algorithm to estimate the parameters of the Goel-Okumoto Model and the Yamada S-Shaped Model. We created an algorithm which resembles Fisher’s scoring algorithm which is a form of the Newton-Raphson method used to solve Maximum Likelihood equations. We transformed Fisher’s algorithm to an algorithm which can be used to estimate parameters in real time. This implies that previous estimations of the parameters are updated by using the last few observations. The general form of the recursive formula is

\[
\hat{\theta}_{i+1} = \hat{\theta}_i + \gamma_i G_i(z_i, \hat{\theta}_i),
\]

where \( G_i(z_i, \hat{\theta}_i) \) is called the gain vector. This vector is created in such a way that it shifts the vector \( \hat{\theta}_i \) to the true value of \( \theta \). A benefit of the recursive algorithm is that it may pose an estimate when the software failure data does not satisfy the conditions mentioned in Table 5.1. In Figures 4.1 to 4.8 can be seen that the recursive algorithm also poses reasonable estimates when the Maximum Likelihood Method seems unstable. We proposed two versions of the algorithm. One to estimate the parameters separately and the other to estimate all parameters simultaneously. A benefit of estimating the parameters separately, is that the recursive formula in this case is less computationally demanding. We applied the method to grouped data because in case of ungrouped data the recursive formulas, except for the recursive formula for the separate estimation of \( a \), could not be calculated.
In case of estimating parameters separately, the recursive formulas to estimate $a$ and $b$ are as follows. In case of the Goel-Okumoto Model:

$$a_{k+1} = a_k (1 - \gamma) + \gamma \frac{e^{b_k (\ell_k + \ell_{k-1})} y_k}{e^{b_k \ell_k} - e^{b_k \ell_{k-1}}}$$

and

$$b_{k+1} = b_k + \gamma \left( -a_k + \frac{y_k}{e^{-b_k \ell_k} (1 + b_k \ell_k)} \right).$$

In case of the Yamada S-Shaped Model:

$$a_{k+1} = a_k (1 - \gamma) + \gamma \frac{y_i}{e^{-b_i \ell_{i-1}} (1 + b_i \ell_{i-1}) - e^{-b_i \ell_{i}}}$$

and

$$b_{k+1} = b_k + \gamma \frac{e^{b_i (\ell_{i+1}^2)} y_i - a e^{b_i} (1 + b_i \ell_{i-1}) + a e^{b_i \ell_{i-1}} (1 + b_i \ell_{i})}{ab (-e^{b_i \ell_{i-1}^2} + e^{b_i \ell_i^2})}.$$

When all parameters are estimated simultaneously, we must use a more complex gain vector to shift the estimate of $\theta$ to its true value. The general formula to calculate this vector is

$$B(z_i, \theta) = \frac{\det(I(y_i, \theta))}{\det(I(z_i, \theta))} I^{-1}(y_i, \theta) \nabla l_i(y_i, \theta) + \frac{\det(I(y_{i-1}, \theta))}{\det(I(z_i, \theta))} I^{-1}(y_{i-1}, \theta) \nabla l_i(y_i, \theta).$$

In case of the Goel-Okumoto Model, we can calculate this expression by using

$$I(y_i, \theta) = \begin{pmatrix} \frac{e^{-b_{i-1}^2 - b_{i}^2}}{a} & -e^{-b_{i-1}^2} \ell_{i-1} + e^{-b_{i}^2} \ell_{i} \\ -e^{-b_{i-1}^2} \ell_{i-1} + e^{-b_{i}^2} \ell_{i} & \frac{e^{-b_{i-1}^2} \ell_{i-1} + e^{-b_{i}^2} \ell_{i}}{e^{b_{i-1}^2} + e^{b_{i}^2}} \end{pmatrix}$$

and

$$\nabla l_i(y_i, \theta) = \begin{pmatrix} y_i (e^{b_{i}^2 \ell_{i-1}^2 - b_{i-1}^2}) + a (e^{-b_{i-1}^2} \ell_{i-1} - e^{-b_{i}^2} \ell_{i}) \\ y_i (e^{b_{i-1}^2 \ell_{i-1}^2 - b_{i}^2}) + a (e^{-b_{i-1}^2} \ell_{i-1} - e^{-b_{i}^2} \ell_{i}) \end{pmatrix}.$$
In case of ungrouped data, most of the gain vectors were too complex to calculate. When we use separate estimation of the parameters, the recursion for the estimation of \( a \) is

\[
a_{i+1} = a_i (1 + \gamma_i (1 - (\Lambda(t_i) - \Lambda(t_{i-1})))).
\]

The gain vector to estimate parameter \( b \) is described by

\[
B_i(t_i, \theta) = \left( -E[\frac{\partial^2}{\partial b^2} l(t_i, \theta)] \right)^{-1} \left( \frac{\partial}{\partial b} l(t_i, \theta) \right).
\]

To calculate this gain vector, we must first determine \(-E[\frac{\partial^2}{\partial b^2} l(t_i, \theta)]\):

\[
-E[\frac{\partial^2}{\partial b^2} l(t_i, \theta)] = \int_{t_{i-1}}^{\infty} \left[ \frac{\partial^2}{\partial b^2} (\Lambda(t_i) - \Lambda(t_{i-1}) - \ln(\lambda(t_i))) \right] \lambda(t_i)e^{-(\Lambda(t_i) - \Lambda(t_{i-1}))} dt_i.
\]

Unfortunately, we could not calculate a closed form of this expression for the models we use in this report. In case of estimating the parameter simultaneously, to calculate the gain vector, we need to calculate

\[
E \left[ \frac{\partial^2}{\partial \theta_l \theta_m} \log(l(a, b)) \right] = E [g(t_{i-1}, t_i)]
= \int_0^\infty \int_0^\infty g(t_{i-1}, t_i) f_{T_i-T_i}(t_{i-1}, t_i) dt_i.
\]

In this equation \( f_{T_i-T_i}(t_{i-1}, t_i) \) is the two-dimensional probability density function, which represents the density of two consecutive events on \( t_{i-1} \) and \( t_i \). Unfortunately, we were not able to write this function in closed form. Thus we could not calculate the Fisher Information Matrix needed to construct the gain vector. Further research is needed to calculate the gain vector in case of simultaneous estimation using ungrouped data.
Chapter 6

Recommendations

In this chapter we describe the aspects of this report which require further research. Because we studied two different methods to estimate parameters of software reliability models, the Maximum Likelihood Method and a recursive estimation method, we divide this chapter into two corresponding sections. Recommendations with respect to Chapter 3, concerning the Maximum Likelihood Method, can be found in Section 6.1. Recommendations with respect to Chapter 4, concerning the recursive method, can be found in Section 6.2.

6.1 Maximum Likelihood Method

This section states the aspects of the report which require further research concerning the Maximum Likelihood Method.

A necessary and sufficient condition, for both grouped and ungrouped data, to assure a unique, positive and finite root of $\Phi(b)$ in case of estimating the parameter $b$ of the Inflection S-Shaped model, when $\psi$ is known and not known.

In Chapter 3 we derived for both the Goel-Okumoto model and the Yamada S-Shaped model necessary and sufficient conditions for the existence of a unique, positive and finite root of $\Phi(b)$. For the Inflection S-Shaped model we only derived a sufficient condition for the existence of at least one root of $\Phi(b)$ in case $\psi$ is known. A condition which assures that $\frac{\partial \Phi}{\partial b}(b) < 0$, would be a sufficient, but not necessary condition, for a unique root of $\Phi(b)$. In case of grouped data, that condition would imply that

$$nk \left[ \frac{\ell_k^2 e^{b \ell_k}}{(e^{b \ell_k} - 1)^2} + \frac{\psi \ell_k^2 e^{b \ell_k}}{(e^{b \ell_k} + \psi)^2} \right] < \sum_{i=1}^{k} y_i \left[ \frac{\psi \ell_{i-1}^2 e^{b \ell_{i-1}}}{(e^{b \ell_{i-1}} + \psi)^2} + \frac{\psi \ell_i^2 e^{b \ell_i}}{(e^{b \ell_i} + \psi)^2} + \frac{(\ell_i - \ell_{i-1})^2 e^{b(\ell_i - \ell_{i-1})}}{(e^{b(\ell_i - \ell_{i-1})} - 1)^2} \right].$$

In case of ungrouped data that condition would imply that

$$n e^{b t_n} t_n^2 \left( -1 + e^{b t_n} \right)^2 + n e^{b u_i} u_i^2 \left( e^{b u_i} + \psi \right)^2 - \frac{n b^2}{\psi} < \sum_{i=1}^{n} 2 e^{b u_i} u_i^2 \left( e^{b u_i} + \psi \right)^2.$$

We have not yet found a condition which guarantees these inequalities and further research is needed. Further research is also needed to find the necessary conditions for a unique, positive and finite estimate of $b$, in case $\psi$ is known and not known.
A necessary and sufficient condition, for both grouped and ungrouped data, to assure a unique, positive and finite root of $\Phi(\psi)$ in case of estimating the parameter $\psi$ of the Inflection S-Shaped model.

Several simulations have led us to believe that a necessary condition for $\Phi(\psi)$ to have a root is $\lim_{\psi \to 0} \Phi(\psi) > 0$. In case of grouped data the following condition has to be satisfied

$$n_k \left(1 + \frac{1}{e^{bl_k}} \right) > \sum_{i=1}^{k} y_i \left[ \frac{1}{e^{bl_{i-1}}} + \frac{1}{e^{bl_i}} \right].$$

In case of ungrouped data, the following condition has to be satisfied

$$n(1 + \frac{1}{e^{bl_k}}) > \sum_{i=1}^{n} \frac{2}{e^{bl_i}}. \quad (6.1)$$

Unfortunately, this is not a sufficient condition. When $\lim_{\psi \to 0} \Phi(\psi) > 0$, the function $\Phi(\psi)$ may have the shape as depicted in Figures 3.1 and 3.2. Clearly, only the function shown in Figure 3.2 has a root. We have not found a necessary and sufficient condition which guarantees the existence of a unique, positive and finite root of $\Phi(\psi)$ and further research is needed.

6.2 Recursive Method

This section states the aspects of the report which require further research concerning the recursive method.

A method to choose the best values for $p$ and $C$.

In the examples in Section 4.4 we defined the non-increasing sequence in the recursive algorithm by choosing $p = 1$ and $C = 1000$. A small value for $p$ and a large value for $C$ gives the algorithm the ability to make large steps towards the true value of the parameter it is trying to estimate. But large steps can also overshoot the true value which can lead to a bad estimate of the parameter. A large value of $p$ and a small value for $C$ leads to small steps. When the initial estimate of the parameter is not close to the real value and the amount of data is small, this may also lead to a bad estimate. Further research is needed to be able to choose the values for $p$ and $C$ wisely.

The calculation of the gain vector for the separate estimation of $b$ in case of ungrouped data.

In case of ungrouped data and estimating the parameters separately, we were only able to calculate the gain vector for parameter $a$. In order to calculate the gain vector for the estimation of $b$ in this particular situation, we need to calculate

$$-E[\frac{\partial^2}{\partial b^2} l(t_i, \theta)] = \int_{t_{i-1}}^{\infty} \left[ \frac{\partial^2}{\partial b^2} (\Lambda(t_i) - \Lambda(t_{i-1}) - \ln(\lambda(t_i))) \right] \lambda(t_i) e^{-(\Lambda(t_i) - \Lambda(t_{i-1}))} dt_i.$$
Unfortunately, for the models used in this report, we were not able to calculate this expression. For the calculation of this expression, further research is needed.

**The calculation of the gain vector for simultaneous estimation of the parameters in case of ungrouped data.**

In case of estimating the parameters simultaneously, to calculate the gain vector, we need to calculate

$$E \left[ \frac{\partial^2}{\partial \theta_l \theta_m} \log(l(a,b)) \right] = E[ g(t_{i-1}, t_i) ]$$

$$= \int_0^\infty \int_0^\infty g(t_{i-1}, t_i) f_{T_{i-1}, T_i}(t_{i-1}, t_i).$$

In this equation $f_{T_{i-1}, T_i}(t_{i-1}, t_i)$ is the two-dimensional probability density function, which represents the density of two consecutive events on $t_{i-1}$ and $t_i$. Unfortunately, we were not able to write this function in closed form. Thus we cannot calculate the Fisher Information Matrix needed to construct the gain vector. Further research is needed to calculate the gain vector in case of simultaneous estimation using ungrouped data.
## Glossary

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>Random variable which represents the initial number of faults in the software</td>
</tr>
<tr>
<td>$n$</td>
<td>Realisation of the random variable $N$</td>
</tr>
<tr>
<td>$(N(t))_{t \geq 0}$</td>
<td>Stochastic process which represents the number of failures in the time interval $[0, t)$</td>
</tr>
<tr>
<td>$\Lambda(t)$</td>
<td>Mean value function of the stochastic process $(N(t))_{t \geq 0}$</td>
</tr>
<tr>
<td>$\lambda(t)$</td>
<td>Intensity function of the stochastic process $(N(t))_{t \geq 0}$</td>
</tr>
<tr>
<td>$\theta$</td>
<td>Vector of model parameters of the distribution of $N(t)$, e.g. $\theta = (\theta_1, \ldots, \theta_d)$</td>
</tr>
<tr>
<td>$n(t)$</td>
<td>Realisation of the random variable $N(t)$</td>
</tr>
<tr>
<td>$T_i$</td>
<td>Random variable which represents the cumulative time until failure $i$</td>
</tr>
<tr>
<td>$t_i$</td>
<td>Realisation of the random variable $T_i$</td>
</tr>
<tr>
<td>$X_i$</td>
<td>Random variable which represents the time between failure $i$ and $i-1$, thus $T_i - T_{i-1}$</td>
</tr>
<tr>
<td>$x_i$</td>
<td>Realisation of the random variable $X_i$, thus $t_i - t_{i-1}$</td>
</tr>
<tr>
<td>$\ell_i$</td>
<td>Predetermined point in time which represents the end of the time interval $[0, \ell_i)$</td>
</tr>
<tr>
<td>$N_i$</td>
<td>Random variable which represents the number of failures in the time interval $[0, \ell_i)$</td>
</tr>
<tr>
<td>$n_i$</td>
<td>Realisation of the random variable $N_i$</td>
</tr>
<tr>
<td>$Y_i$</td>
<td>Random variable which represents the number of failures in the time interval $[\ell_{i-1}, \ell_i)$ thus $N_i - N_{i-1}$</td>
</tr>
<tr>
<td>$y_i$</td>
<td>Realisation of the random variable $Y_i$, thus $n_i - n_{i-1}$</td>
</tr>
<tr>
<td>$\mu(\ell_i)$</td>
<td>Expected number of failures in the time interval $[\ell_{i-1}, \ell_i)$, thus $\Lambda(\ell_i) - \Lambda(\ell_{i-1})$</td>
</tr>
</tbody>
</table>
Bibliography


