Maximum likehood estimators for debugging models in software reliability with ungrouped data

Jongerius, S.C.

Award date:
2018

TU/e Technische Universiteit Eindhoven

Bachelor Thesis

Maximum Likelihood Estimators for Debugging Models in Software Reliability with Ungrouped Data

Author:
Sabine Jongerius
s.c.jongerius@student.tue.nl
0858438

Supervisor:
Dr. A. Di Bucchianico

31st July 2017

Contents

Notation

- LOC lines of code
- NHPP non-homogeneous Poisson process
- SRGM Software reliability growth model
- $N(t)$ number of faults until time t
- $\Lambda(t) = \int_0^t \lambda(x)dx$ mean value function of NHPP
- $\lambda(t)$ intensity function of NHPP
- $a\overline{\Lambda}(t) = \Lambda(t)$
- $a(t)$ error content function
- $b(t)$ error detection rate
- t_i detection time of the i-th fault

Chapter 1

Introduction

Current society depends largely on technological devices, for example computers, mobile phones but also aeroplanes and cars. Many of these devices require software that is made of thousands of lines of code. These are all (or largely) handwritten by several software engineers. This of course may give rise to a high probability at mistakes. Since the effects of software errors can be catastrophic (or maybe just really inconvenient), it is really important to test the software before bringing it to the market, to make it more reliable. In this process software reliability engineering has a big role. Companies do not want to test there software longer than necessary. Due to strategic reasons it is best to bring your software to market as fast as possible. But bringing flawed software to the market is probably even worse. This is why software reliability models are being developed and used during the testing process.

Software reliability growth models are statistical models to predict how many bugs there still are in the software after a period of testing. These models are aimed at monitoring the decrease of the faults in the software, with the effect of an increase of the reliability of the software. There are over 200 of such models. These models come with several assumptions. Some of the assumptions are software specific, so an appropriate model can be chosen. Other assumptions are more crucial and might not always be realistic. For example, perfect debugging models assume that no new mistakes are made during the correcting of the errors that are found. When there is an extensive debugging process, this might not be a realistic assumption. This is why imperfect debugging models are introduced. Kumar (2016) provides a nice overview of all developments in the area of software debugging over the last 30 years.

1.1 Problem Introduction

A software reliability engineer addresses several problems before adapting a software reliability growth model. First a choice must be made for a specific model. In this report we will focus on Non-homogenous Poisson Process (NHPP) debugging models, but there is a much broader range of models to choose from. The choice of model (narrowing the options to NHPP) is not trivial. One could use perfect, imperfect fault debugging or pure error generation models. When using software reliability models one must also be sure to fit appropriate error data (of fault counts) in order to get correct estimators for the different parameters. The most used method to estimate these parameters is the Maximum Likelihood Method. Unfortunately this method does not always give a solution. There should for example be a trend in the data (improvement), because models are created with the assumption that more and more mistakes are repaired and the software becomes more reliable. Only testing for a trend is not enough, more strict conditions can be derived to ensure the existence of the solutions for the maximum likelihood equations. These conditions are already there for most perfect debugging models but are not yet derived for imperfect debugging models and pure error generation models. Due to the fast development of the area of software reliability a lot of models have been proposed, but not thoroughly analysed. Adding parameters to a simple model might give rise to problems, for example identifiability of parameters. Then one definitely encounters problems with maximum likelihood estimation. The main goal of this report is to review identifiability of the parameters for the selected models. The other goal is to obtain conditions on existence of the solution of the maximum likelihood equations. This is done theoretically and with a numerical analysis.

1.2 Overview Report

In Chapter 2 we give a description of the modelling of software reliability growth with NHPP models. After the derivation of the basic model we give a description on possible ways to choose a model and provide information about data in this context. In Chapter 3 we study the Maximum Likelihood Method and present the main ideas for existence and uniqueness of the maximum likelihood Estimators. Chapter 4 gives an overview of perfect, imperfect and pure error generation models and provides the connection between them. Chapter 5 handles the Pure Error Generation Model. Chapter 5 also provides conditions on existence of the maximum likelihood estimators with a known parameter. Chapter 5 concludes with presentation of the Pure Imperfect Fault Debugging Model due to its similarity with the Pure Error Generation Model. In Chapter 6 we present the Delayed S-shaped Pure Error Generation Model. Chapter 7 describes Yamada Imperfect Debugging Model 1 with its assumptions and likelihood equations. Chapter 8 provides this for Yamada Imperfect Debugging Model 2. In Chapter 9 we present the Pham Nordmann Zhang Model with its likelihood equations and a property that differs from the perfect debugging models. Chapter 10 presents the Pham Exponential Imperfect Model. Chapter 11 provides an analysis of the 3-parameter models with a dataset, to support the findings of the theoretical research. In Chapter 12 we draw conclusions from the results gathered during our research. In Chapter 13 we state several suggestions for further research. Note that a Glossary with all used notations is included in the beginning of the report.

Chapter 2

Software reliability models based on NHPP

Several models are used to describe software debugging. In this report we model the number of observed failures up to time t as a birth counting process $(N(t))_{t \geq 0}$. A large class within the birth counting processes is the non-homogeneous Poisson process. We will investigate models based on this process in this report. NHPP based models are quite extensively researched and developed, still there remains a lot to discover. Although the belief that software reliability growth follows a NHPP is widely supported, there are some issues raised against this approach. Cai et al. (2008) describes two controlled software experiments where empirical observations and statistical hypothesis testing suggest that software reliability behaviour do not follow a NHPP in general. In Lin et al. (2010) the variance of NHPP -models and the independence of failures in disjoint time intervals were researched as concerns against NHPP. The report concludes that although the concern regarding independence is not refuted, NHPP is still workable from the applicable perspective.

Below we first describe the definition of a pure birth counting process and the properties it has to satisfy to be a non-homogeneous Poisson process. Then we describe the construction of the specific models. After that we elaborate on how the choice for a specific model should be made and how fault data is presented in software reliability.

2.1 Non-homogenous Poisson process

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, $N(t) > 0$*
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 $t_i < t_j < t_k < t_l$ the random variables $N(t_j) - N(t_i)$ and $N(t_l) - N(t_k)$ 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*

$$\mathbb{P}[N(t_j) - N(t_i) = k] = e^{-(\Lambda(t_j) - \Lambda(t_i))} \frac{(\Lambda(t_j) - \Lambda(t_i))^k}{k!} \tag{2.1}$$

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 an 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 disjoint time intervals are independent. This is an important property. The quantity $\Lambda(t)$ describes the expected number of failures up to time t. $\lambda(t)$ is the intensity function of the Poisson process. Note that in literature a lot of different concept can be denoted when speaking about "time between failure". Thompson (1981) describes the different possibilities and how to interpret them.

2.2 Model description

In this section we describe how software reliability growth models are build. The basic assumptions that hold for all models (perfect and imperfect) are:

- Failure observation phenomenon is modelled by NHPP
- A detected fault is corrected immediately
- No fault occurs at the start of the test
- Reliability is a function of the remaining faults

The generalized mean value function of the model is the solution of the following differential equations (Pham, 2000):

$$\frac{\partial}{\partial t}\Lambda(t) = b(t)\left[a(t) - \Lambda(t)\right] \tag{2.2}$$

with the initial condition $\Lambda(t_0) = \Lambda_0$, (for models in this report $\Lambda(t_0) = 0$) and using the technique of the integrating factor, we derive

$$\Lambda(t) = e^{-B(t)}\left[\Lambda_0 + \int_{t_0}^{t} a(x)b(x)\ e^{B(x)}dx\right] \tag{2.3}$$

where

$$B(t) = \int_{t_0}^{t} b(x)dx \tag{2.4}$$

In these equations all parameters have a separate meaning, listed in Table 2.1.

Table 2.1: Parameter descriptions

$a(t)$	Expected initial error content at time t, $a > 0$
a	Number of errors in the software at the start of software testing
$b(t)$	Time dependent rate of fault removal per remaining faults
b	Constant rate of fault detection/removal per remaining faults in the software.
c	Constant rate of error generation, $0 < c < 1$
d	Constant parameter in the logistic function

A constant error content function, the choice $a(t) = a$, yields a perfect debugging model, then no new faults are introduced during the debugging process. The choice $b(t) = b$ yields the simplest perfect debugging model in this category, the Goel-Okumoto model. In Table 4.1 the several widely used debugging models are described. Most models can be transformed into a simpler model if we take some parameters equal to 0. An overview of the connection between these models is given in Section 4.2. For the models that are elaborated in this report, the underlying assumptions are mentioned in the corresponding chapters.

2.3 Choice of model

With all various models in place, the first task of a software project manager is to select a suitable model to use. This has proven to be a very difficult task. Khoshgoftaar and Woodcock (1991) suggests to select some models and use the Akaike Information Criterion. In this report the choice for the specific models that are compared is not elaborated extensively. However Kharchenko et al. (2002) provides a more structured approach in the first selection of model(s) using a matrix (database) based on all underlying assumptions of the different models. A combination of both selection criteria seems a reasonable approach.

2.4 Data presentation

In software testing there are two ways to provide test data. The most straightforward type is ungrouped data (time domain data). Every fault that is detected is denoted by the exact time it was found. This is supported by the property of NHPP that no two detections occur at the same time.

Grouped data (interval domain data) is presented as a failure count of predetermined time intervals. These time frames do not have to be of the same length. It is obvious that ungrouped data can be converted to grouped data, if necessary, but not the other way around. This conversion is also called interval censoring.

Contrary to what is sometimes said in Software Reliability Literature, models may have data of both kind, grouped and ungrouped. For both data presentations the likelihood equations can be derived for the use of maximum likelihood estimators. In this report we will focus on ungrouped data.

Most of the existing software reliability models describe the failure rate as constant increasing or decreasing over time. In reality this may not be the case due to the learning process of the tester. This implies less testing of failures in the beginning of the process. Then when the testers are familiar with the software, the detection of faults increases. And when most of the software failures are eliminated the failure rate will decrease. This phenomenon is not incorporated in most models but Hanagal and Bhalerao (2016) introduced a model that deals with this.

Note that when we speak of time, we mean effective testing time.

Chapter 3

Maximum Likelihood Method

The parameters of the different software reliability models can be estimated in several ways. In this chapter we introduce the general theory concerning existence and uniqueness of maximum likelihood estimators. (Knafl, 1992), (Knafl and Morgan, 1996) and (Hossain and Dahiya, 1993) have derived conditions for the existence and uniqueness of maximum likelihood estimators for the perfect debugging models Goel-Okumoto and Yamada S-Shaped, that will be handled in section 4.1. For the Inflection S-Shaped model, which is made of three parameters, Meyfroyt (2012) succeeded to derive a sufficient condition for the existence of at least one root in case d was known. With the development of new, more complicated models, more research about existence and uniqueness of the maximum likelihood estimators is necessary.

Another way of estimating the parameters of software reliability models is with Least Square Estimation and especially Weighted Least Square Estimation, described by e.g. Koh (2008) and Kuhl et al. (1998). This is beyond the scope of this report.

3.1 Ungrouped data

We now present a theorem from Rigdon and Basu (2000) for ungrouped data. From this theorem we obtain the log-likelihood of non-homogenous Poisson Process models. With this we obtain the derivatives of the likelihood function, that we use to derive conditions on the existence of a solution of the maximum likelihood equations.

Theorem 3.1. *The log-likelihood of a non-homogeneous Poisson Process with mean value function $\Lambda(t)$ and the observed event times $0 = t_0 < t_1 < ... < t_n$, is given by*

$$L = \sum_{i=1}^{n} \ln(\lambda(t_i)) - \Lambda(t_n). \tag{3.1}$$

For all debugging models described in this report, it is possible to factor out a from the mean value function and its derivative. This gives a mean value function we denote with $a\overline{\Lambda}(t)$ and a corresponding intensity function $a\overline{\lambda}(t)$. When substituting $a\overline{\Lambda}(t)$ in the expression for the log-likelihood as in Theorem 3.1, we obtain

$$L = n\ln(a) + \sum_{i=1}^{n} \ln(\overline{\lambda}(t_i)) - a\overline{\Lambda}(t_n). \tag{3.2}$$

Taking the derivative with respect to a and setting the expression equal to zero gives:

$$\frac{\partial L}{\partial a} = \frac{n}{a} - \overline{\Lambda}(t_n) = 0 \tag{3.3}$$

We can solve (3.3) with respect to a explicitly :

$$a = \frac{n}{\overline{\Lambda}(t_n)}. \tag{3.4}$$

From this expression Meyfroyt (2012) derived some properties for the Goel-Okumoto Model. In order to estimate the other parameters, differentiating and setting the expressions equal to zero leads to the following expression (for all parameters other than a)

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} \frac{\frac{\partial}{\partial b}\overline{\lambda}(t_i)}{\overline{\lambda}(t_i)} - a\frac{\partial}{\partial b}\overline{\Lambda}(t_n) = 0. \tag{3.5}$$

Inserting the explicit expression for a, we obtain

$$\sum_{i=1}^{n} \frac{\frac{\partial}{\partial b}\overline{\lambda}(t_i)}{\overline{\lambda}(t_i)} - n\frac{\frac{\partial}{\partial b}\overline{\Lambda}(t_n)}{\overline{\Lambda}(t_n)} = 0. \tag{3.6}$$

3.2 Existence and uniqueness

A problem that rises with the Maximum Likelihood method is the existence of a positive, unique and finite solution to the likelihood equations. For most perfect debugging models some conditions on the data have been derived to ensure the existence of the solution (Knafl and Morgan (1996),Hossain and Dahiya (1993) and Meyfroyt (2012)). We mention such conditions for some well-known models and show that non-existence of ML estimators in these cases is not a weakness of the ML method, but reflects that the data does not exhibit reliability growth. If data does not exhibit reliability growth, these models are not useful since they assume reliability growth.

There are multiple ways to check if data shows a trend, or so called growth. The following hypothesis test is a widely used option. Under the null hypothesis of an HPP (homogeneous poisson process) and conditioned on the event $\{N(t) = n\}$, it follows that $T_1 < T_2 < \ldots < T_n$ are distributed as n order statistics from a uniform distribution on the interval $(0, t_n)$ (cf. Rigdon and Basu (2000)[Theorem 22]). Thus, the random variable $S = \sum_{i=1}^{n} T_i$ has mean $nt_n/2$ and variance $nt_n^2/12$. Therefore, by the Central Limit Theorem it follows that

$$U_L = \frac{\displaystyle\sum_{i=1}^{n} T_i - nt_n/2}{\sqrt{nt_n^2/12}} \longrightarrow Z \sim \mathcal{N}(0,1) \; .$$

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_α denotes the $(1 - \alpha)-$ percentile of the standard normal distribution. The interpretation of the test statistic (known as the Laplace test statistic) is thus the following: for small values of the test statistic the null hypothesis of HPP is rejected in favour of reliability growth.

Another way of checking whether data exhibits reliability growth is by performing a graphical trend analysis. An example of this is a running averages plot. A more sophisticated well-known trend plot is the Total Time on Test (TTT) plot (see e.g. Kvaløy and Linqvist (1998)). This plot was developed for non-repairable systems, in order to decide whether a certain distribution exhibits reliability growth. In Barlow and Davis (1977) the adaptation to repairable systems was first described. The TTT plot is defined as a plot of the points $(i/n, t_i/t_n)$ where t_i is the observation time of the i-th fault of a total of n found faults. Since this TTT plot goes through $(0, 0)$ and $(1, 1)$ it is easy to see if the graph is below the diagonal everywhere). That is a necessary but not sufficient condition for the notion of reliability growth.

Chapter 4

Debugging Models Overview

In this chapter we will describe several NHPP reliability growth models to model software failure testing. We give a brief description of the difference in perfect, imperfect and error generation models. Then we give an overview of the models. We conclude with connections between the different models, which provides more insight in the way models are extended.

4.1 Perfect and Imperfect Debugging Models

Since NHPP models are being used to describe the software debugging a wide range of models have been developed. At first only so-called perfect debugging models were used. These models assume that failures, when found, are immediately repaired (perfectly). They also assume that the software engineer that repairs the fault does not create any new mistakes in the software while repairing the found fault. To incorporate the possibility of imperfect removal of faults and generation of faults during the process, the so called *imperfect debugging models* were developed. Within this category there are two classes. The *imperfect fault debugging* models, where found faults are not completely removed with a certain probability. The possibility of creating new faults during removal of existing fault is called *error generation(or fault generation)*, and these models were developed as well. This concept of imperfect debugging was first introduced by Goel (1985). Kapur and Garg (1990) were the first to incorporate the notion of imperfect debugging in the Goel-Okumoto Model. Kapur et al. (2011) gives an overview of these three types (perfect, imperfect fault debugging, error generation) of models. They also describe debugging processes to investigate probabilities connected to these models. In that way for example a probability at imperfect repair can be reviewed for part of a software test. Then for the remaining testing phase, this estimate of the probability can be incorporated in the model and does not have to be estimated via the model. Table 4.1 provides an overview of the models treated in this report as well as the most fundamental perfect debugging models. Note that in literature the distinction between imperfect debugging and imperfect fault debugging was often wrongly made. Kapur et al. (2011) makes this distinction clear.

The existence of the solution of maximum likelihood estimators of perfect debugging models has been examined extensively as mentioned in Section 3.2. The conditions of the Goel-Okumoto are due to Hossain and Dahiya (1993) and Knafl and Morgan (1996) for grouped and ungrouped data. The conditions for the Yamada S-shaped for ungrouped data are due to Knafl and Morgan (1996) and proven by Zhao and Xie (1996). The conjecture (Knafl and Morgan, 1996) of conditions for the Yamada S-shaped for ungrouped data were proven by Meyfroyt (2012). We present the conditions for ungrouped data for these perfect debugging models. Note that these are the conditions for the derivative of the likelihood to b to have a root.

Theorem 4.1. *In case of estimating the parameters of the Goel-Okumoto model using ungrouped data, the observed event times $0 = t_0 < t_1 < ... < t_n$, a necessary and sufficient condition for the existence of a unique, positive and finite solution of the maximum likelihood equations is given by*

$$t_n > \frac{2\sum_{i=1}^{n} t_i}{n}. \tag{4.1}$$

Theorem 4.2. *In case of estimating the parameters of the Yamada S-shaped model using ungrouped data, the observed event times $0 = t_0 < t_1 < ... < t_n$, a necessary and sufficient condition for the existence of a unique, positive and finite solution of the maximum likelihood equations is given by*

$$t_n > \frac{\frac{3}{2}\sum_{i=1}^{n} t_i}{n} \tag{4.2}$$

Table 4.1: Overview of debugging models

Perfect debugging models

Goel-Okumoto

$a(t) = a$

$b(t) = b$

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

Yamada S-shaped

$a(t) = a$

$b(t) = \frac{b^2 t}{1+bt}$

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

Inflection S-shaped

$a(t) = a$

$b(t) = \frac{b}{1+de^{-bt}}$

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

Imperfect fault Debugging models

Pure Imperfect Fault Debugging Model

$\Lambda_r(t) = a(1 - e^{-bpt})$

$\Lambda_f(t) = (\frac{a}{p})(1 - e^{-bpt})$

Error Generation Models

Pure Error Generation Model

$a(t) = a + c\Lambda(t)$

$b(t) = b$

$\Lambda(t) = \frac{a}{1-c}(1 - e^{-b(1-c)t})$

Delayed S-shaped Pure Error Generation Model

$a(t) = a + c\Lambda(t)$

$b(t) = \frac{b^2 t}{1+bt}$

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

Yamada Imperfect Debugging Model 1

$a(t) = ae^{kt}$

$b(t) = b$

$\Lambda(t) = \frac{ab}{k+b}(e^{kt} - e^{-bt})$

Yamada Imperfect Debugging Model 2

$a(t) = a(1 + ct)$

$b(t) = b$

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

PNZ model

$a(t) = a(1 + ct)$

$b(t) = \frac{b}{1+de^{-bt}}$

$\Lambda(t) = \frac{a}{1+de^{-bt}}((1 - e^{-bt})(1 - \frac{c}{b}) + ct)$

Pham exponential imperfect model

$a(t) = ae^{kt}$

$b(t) = \frac{b}{1+de^{-bt}}$

$\Lambda(t) = \frac{ab}{b+k}(\frac{e^{(k+b)t}-1}{e^{bt}+d})$

For perfect debugging models another nice attribute can be derived. Namely the estimator of a, the estimated number of initial faults can never be larger than the total amount of faults in the software. This is because $\lim_{t\to\infty} \overline{\Lambda}(t) = 1$. So with the explicit derivation in (3.4) we see that then the estimate of a is always smaller than n, the total number of faults found.
The Error generation models will be handled in the following chapters. Chapter 5 handles the Pure Error Generation Model. Chapter 5 concludes with presentation of the Pure Imperfect Fault Debugging Model due to its similarity with the Pure Error Generation Model. In Chapter 6 we present the Delayed S-shaped Pure Error Generation Model. Chapter 7 describes Yamada Imperfect Debugging Model 1 with its assumptions. Chapter 8 provides this for Yamada Imperfect Debugging Model 2. In Chapter 9 we present the Pham Nordmann Zhang Model. Chapter 10 presents the Pham Exponential Imperfect Model.

4.2 Comparison Debugging Models

Finding conditions for existence of a solution of the maximum likelihood equations for imperfect or error generation models is not trivial. Most imperfect and error generation models are extensions of perfect debugging models. We now present an overview of the simplifications that can be done to obtain a perfect debugging model.

Pure Imperfect Fault Debugging Model	
Λ_f with $p = 1 - c$	similar to Pure Error Generation model
Pure Error Generation Model	
with $c = 0$	reduces to Goel-Okumoto model
Delayed S-shaped Pure Error Generation Model	
with $c = 0$	reduces to Yamada S-shaped
Inflection S-shaped model	
with $d = 0$	reduces to Goel-Okumoto model
Yamada imperfect debugging model 1	
with $k = 0$	reduces to Goel-Okumoto model
Yamada imperfect debugging model 2	
with $c = 0$	reduces to Goel-Okumoto model
PNZ model	
with $c = 0$	reduces to Inflection S-shaped model
with $d = 0$	reduces to Yamada Imperfect debugging model 2
with $c = 0$ and $d = 0$	reduces to Goel-Okumoto model
Pham exponential imperfect model	
with $k = 0$	reduces to Inflection S-shaped model
with $d = 0$	reduces to Yamada Imperfect debugging model 1
with $k = 0$ and $d = 0$	reduces to Goel-Okumoto model

Chapter 5

Pure Error Generation Model

Ohba and Chou (1989) stated that the assumptions of perfect debugging models might not suffice the complexity of the software testing processes. Therefore they suggested to extend the Goel-Okumoto Model with an extra parameter. This parameter $0 < c < 1$ is an error introduction rate. The model is made such that only when there are more errors found, more errors can be introduced. This model assumes that:

1. The fault generation rate at any time t is proportional to the fault removal rate at that time

We will use the notation of Chapter 2. Assume that the time-dependent fault content function and error detection rate are

$$a(t) = a + c\Lambda(t) \tag{5.1}$$

$$b(t) = b \tag{5.2}$$

respectively, where $a = a(0)$ is the parameter for the total number of initial faults that exist in the software before testing and b is the error detection rate. The mean value function is given by

$$\Lambda(t) = \frac{a}{1-c}(1 - e^{-b(1-c)t}) \tag{5.3}$$

$$\lambda(t) = abe^{-b(1-c)t} \tag{5.4}$$

Then, as described in Section 3.1, a can be factored out. This gives a mean value function, that we denote with $\overline{\Lambda}(t)$ and a corresponding intensity function $\overline{\lambda}(t)$.

5.1 Identifiability

From the mean value function $\Lambda(t)$ the question rises if $1-c$ is an effective contribution to this model, since this effectively is an extension of the Goel-Okumoto Model. We check this by replacing $1-c$ with $m(1-c)$, m being an arbitrary constant. If we put this in the mean value function, we obtain $\Lambda(t) = \frac{a^*}{m(1-c)}(1-e^{-b^* m(1-c)t})$. This is equal to the original model with $a = \frac{a^*}{m}$ and $b = b^* m$. This means that this model does not uniquely identify the three parameters. This gives rise to problems when estimating the parameters. It remains a possibility to estimate the model with a known c. For this conditions are derived in Section 5.4.

5.2 Likelihood equations

When substituting the explicit expression for a in (3.2), we can obtain the log-likelihood function expressed in all parameters except a.

$$a = \frac{n}{\overline{\Lambda}(t_n)}. \tag{5.5}$$

$$\sum_{i=1}^{n} \log\left(be^{-b(1-c)t_i}\right) + n\log(n) - \log\left(1 - e^{-b(1-c)t_n}\right) + \log(1-c) - n \tag{5.6}$$

Then we can derive the expressions of the derivatives of the likelihood function with respect to all other parameters with Formula (3.6) as follows

$$\frac{\partial L}{\partial b} = \frac{n}{b} - n\frac{(c-1)t_n e^{b(c-1)t_n}}{e^{b(c-1)t_n} - 1} - (1-c)\sum_{i=1}^{n} t_i \tag{5.7}$$

$$\frac{\partial L}{\partial c} = \sum_{i=1}^{n} bt_i - n\frac{e^{b(c-1)t_n}(b(c-1)t_n - 1) + 1}{(c-1)\left(e^{b(c-1)t_n} - 1\right)} \tag{5.8}$$

5.3 Property of estimator initial number of faults

For the Pure Error Generation model the nice property that a, the number of initial faults, is always smaller than the number of faults that are found, does not hold, since:

$$\lim_{t\to\infty} \overline{\Lambda}(t_n) = \frac{1}{1-c} \tag{5.9}$$

together with (5.5). This gives $0 < \overline{\Lambda}(t_n) < \frac{1}{1-c}$, with $0 < 1-c < 1$. So while for the Goel-Okumoto model $0 < \overline{\Lambda}(t_n) < 1$ and thus $0 < n < a$, as shown in Section 4.1. For the Pure Error Generation model $0 < (1-c)n < a$. This is in line with the assumption of this model. Errors are introduced with rate c and the repair of errors is done perfectly. So the estimator for a, initial faults in the system before testing, can be smaller than n, but is always bigger than $(1-c)n$. The faults that are introduced with rate c during the debugging process, are added to the initial amount of faults to be found. Note that the consequence is that more faults can be found than initially are present in the software.

5.4 Conditions on existence of Maximum Likelihood for c known, ungrouped data

In this section we will derive conditions on the existence of a solution of the maximum likelihood equations, with p known. We first present two useful lemma from Meyfroyt (2012).

Lemma 5.1. $\lim_{x\downarrow 0} \frac{e^x-(1+x)}{x(e^x-1)} = \frac{1}{2}$

This can be proven with Taylor expansion.

Lemma 5.2. *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) $\lim_{x\downarrow 0} g(x) = 1$

The proof is given in Meyfroyt (2012).

Similar to the derivation for Goel-Okumoto in Knafl and Morgan (1996) we can obtain necessary and sufficient conditions for the existence of a unique, positive and finite solution of the maximum likelihood equations with $0 < c < 1$ known. The idea of the proof is to show that $\frac{\partial L}{\partial b}$ is decreasing, positive close to zero and negative for $b \to \infty$ as is the case in Figure 11.2

Theorem 5.3. *In case of estimating the parameters of the Pure Error Generation model using ungrouped data with* $0 < c < 1$ *known, a necessary and sufficient condition for the existence of a unique, positive and finite solution of the maximum likelihood equations is given by,*

$$t_n > \frac{2(1-c)\sum_{i=1}^n t_i}{n}. \tag{5.10}$$

Proof. We only have to prove that (5.10) 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, because in Section 3.1 we derived an explicit expression for a dependent on b and c. We will show that $\lim_{b\to\infty} \frac{\partial L}{\partial b} < 0$ and that $\frac{\partial L}{\partial b}$ is a decreasing function. Then we will show that $\lim_{b\downarrow 0} \frac{\partial L}{\partial b} > 0$ if and only if (5.10) holds and thus $\frac{\partial L}{\partial b}$ has a positive root. We will start with determining $\lim_{b\to\infty} \frac{\partial L}{\partial b}$.

$$\begin{aligned}\lim_{b\to\infty} \frac{\partial L}{\partial b} &= \lim_{b\to\infty} \frac{n}{b} - n\frac{(c-1)t_n e^{b(c-1)t_n}}{e^{b(c-1)t_n}-1} - (1-c)\sum_{i=1}^n t_i \\ &= -(1-c)\sum_{i=1}^n t_i\end{aligned} \tag{5.11}$$

since $0 < c < 1$ and thus $1-c > 0$. This implies

$$\lim_{b\to\infty} \frac{\partial L}{\partial b} = -(1-c)\sum_{i=1}^n t_i < 0.$$

Next we show that $\frac{\partial L}{\partial b}$ is a decreasing function. Differentiating and simplifying $\frac{\partial L}{\partial b}$, we get

$$\frac{\partial^2 L}{\partial b^2} = -\frac{n}{b^2}\left(1 - \left(\frac{-(1-c)bt_n}{e^{-(1-c)bt_n}}\right)^2 e^{-b(1-c)t_n}\right) \tag{5.12}$$

According to Lemma 5.2 the function $g(b) = \left(\frac{-(1-c)bt_n}{e^{-(1-c)bt_n}}\right)^2 e^{-b(1-c)t_n}$ is decreasing and $\lim_{b\downarrow 0} g(b) = 1$. Because $g(b) < 1$, sinc $e^x \geq x$, the derivative of $\frac{\partial L}{\partial b}$ with respect to b is negative and thus $\frac{\partial L}{\partial b}$ is a decreasing function. Now we have to show that $\lim_{b\downarrow 0} \frac{\partial L}{\partial b} > 0$. If we rewrite $\frac{\partial L}{\partial b}$ and use Lemma 5.1 we obtain

$$\begin{aligned}\lim_{b\downarrow 0}\frac{\partial L}{\partial b} &= \lim_{b\downarrow 0}\frac{n}{b} - n\frac{(c-1)t_n e^{b(c-1)t_n}}{e^{b(c-1)t_n}-1} - (1-c)\sum_{i=1}^{n} t_i \\ &= \lim_{b\downarrow 0} -(1-c)\sum_{i=1}^{n} t_i + (1-c)nt_n\left(\frac{e^{(1-c)bt_n} - (1+(1-c)bt_n)}{(1-c)bt_n(e^{(1-c)bt_n}-1)}\right) \\ &= -(1-c)\sum_{i=1}^{n} t_i + \frac{nt_n}{2}\end{aligned} \tag{5.13}$$

For this last expression to be positive, (5.10) must be satisfied. □

5.5 Pure Imperfect Fault Debugging Model

In this section we will present the Pure Imperfect Fault Debugging Model. In this model it is assumed that on a removal attempt, a fault is removed perfectly with probability p (Kapur and Garg, 1990). In this model two mean value function are distinguished. $\Lambda_r(t)$ is the mean value function of the fault removal process and $\Lambda_f(t)$ is the mean value function of the fault detection process.

$$\Lambda_r(t) = a(1 - e^{-bpt}) \tag{5.14}$$

$$\Lambda_f(t) = \frac{a}{p}(1 - e^{-bpt}) \tag{5.15}$$

We present this model alongside the Pure Error Generation Model since the mean value function of the fault detection process is similar to the mean value function of the Pure Imperfect Fault Debugging Model with $p = 1 - c$. So when using the Pure Imperfect Fault debugging model, the same problems rise as with the Pure Error Generation Model.

Chapter 6

Delayed S-shaped Pure Error Generation Model

When investigating software failure data an S-shaped curve was observed often. Yamada et al. (1984) developed models that fitted this shape well. This model is extended with an error introduction rate (Kapur et al., 2005). This model assumes that:

1. The fault generation rate at any time t is proportional to the fault removal rate at that time
2. The error detection rate is a learning fault detection rate

We will use the notation of Chapter 2. Assume that the time-dependent fault content function and error detection rate are

$$a(t) = a + c\Lambda(t) \tag{6.1}$$

$$b(t) = \frac{b^2 t}{1 + bt} \tag{6.2}$$

respectively, where $a = a(0)$ is the parameter for the total number of initial faults that exist in the software before testing. The mean value function is given by

$$\Lambda(t) = \frac{a}{1-c}(1 - e^{-(1-c)bt}(bt+1)^{1-c}) \tag{6.3}$$

$$\lambda(t) = \frac{a}{1-c}\left(-b(c-1)e^{b(c-1)t}(bt+1)^{1-c} - b(1-c)e^{b(c-1)t}(bt+1)^{-c}\right) \tag{6.4}$$

This can be simplified to

$$\lambda(t) = ab^2te^{b(c-1)t}(bt+1)^{-c} \tag{6.5}$$

Then, as described in Section 3.1, a can be factored out. This gives a mean value function, that we denote with $\overline{\Lambda}(t)$ and a corresponding intensity function $\overline{\lambda}(t)$.

6.1 Identifiability

From the mean value function $\Lambda(t)$ the question rises if the error introduction rate, $1 - c$, is an effective contribution to this model, since this effectively is an extension of the Yamada S-shaped Model. We check this by replacing $1 - c$ with $m(1 - c)$, m being an arbitrary constant. If we put this is the mean value function, we obtain
$\Lambda(t) = \frac{a^*}{m(1-c)}(1 - e^{-m(1-c)b^*t}(b^*t+1)^{m(1-c)})$. Here it is not clear that problems rise with identifiability of parameters as with the Pure Error Generation Model (Section 5.1). However we cannot conclude that the parameters are identifiable. This is investigated further by numerical analysis in Section 11.5.

6.2 Likelihood equations

When substituting the explicit expression for a in (3.2), we can obtain the log-likelihood function expressed in all parameters except a.

$$a = \frac{n}{\overline{\Lambda}(t_n)}. \tag{6.6}$$

$$L = \sum_{i=1}^{n} \log\left(b^2 t_i e^{b(c-1)t_i}(bt_i+1)^{-c}\right) + n\log(1 - e^{b(c-1)t_n}(bt_n+1)^{1-c}) - \log(1-c) - n \tag{6.7}$$

Then we can derive the expressions of the derivatives of the likelihood function with respect to all other parameters with Formula (3.6).

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} \frac{b^2(c-1)t_i^2 + bt_i + 2}{b(bt_i+1)} - n\frac{b(c-1)t_n^2 e^{b(c-1)t_n}}{(bt_n+1)e^{b(c-1)t_n} - (bt_n+1)^c} \tag{6.8}$$

$$\frac{\partial L}{\partial c} = \sum_{i=1}^{n} bt_i - \log(bt_i+1) - n\frac{e^{-bt_n}\left(e^{bt_n}(bt_n+1)^c + (bt_n+1)e^{bct_n}(b(c-1)t_n - 1) - (c-1)(bt_n+1)e^{bct_n}\log(bt_n+1)\right)}{(c-1)\left((bt_n+1)e^{b(c-1)t_n} - (bt_n+1)^c\right)} \tag{6.9}$$

6.3 Property of estimator initial number of faults

For the Delayed S-shaped Pure Error Generation Model the nice property that a, the number of initial faults, is always smaller than the number of faults that are found, does not hold, since:

$$\lim_{t\to\infty} \overline{\Lambda}(t_n) = \frac{1}{1-c} \tag{6.10}$$

together with (5.5). This gives $0 < \overline{\Lambda}(t_n) < \frac{1}{1-c}$, with $0 < 1-c < 1$. So while for the Goel-Okumoto model $0 < \overline{\Lambda}(t_n) < 1$ and thus $0 < n < a$, as shown in Section 4.1. For the Pure Error Generation model $0 < (1-c)n < a$. This is in line with the assumption of this model. Errors are introduced with rate c and the repair of errors is done perfectly. So the estimator for a, initial faults in the system before testing, can be smaller than n, but is always bigger than $(1-c)n$. The faults that are introduced with rate c during the debugging process, are added to the initial amount of faults to be found. Note that the consequence is that more faults can be found than initially are present in the software.

6.4 Conditions on existence of Maximum Likelihood for c known, ungrouped data

In this section we will start deriving conditions on the existence of a solution of the maximum likelihood equations, with p known. We only have to find conditions that ensure 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. In order to find the conditions it suffices to show that $\lim_{b\to\infty}\frac{\partial L}{\partial b} < 0$ and that $\frac{\partial L}{\partial b}$ is a decreasing function. Then we will have to show that $\lim_{b\downarrow 0}\frac{\partial L}{\partial b} > 0$ holds under certain conditions and thus $\frac{\partial L}{\partial b}$ has a positive root. We will start with determining $\lim_{b\to\infty}\frac{\partial L}{\partial b}$. Note that $0 < c < 1$ and thus $0 < 1 - c < 1$.

$$\begin{aligned}
\lim_{b\to\infty}\frac{\partial L}{\partial b} &= \lim_{b\to\infty}\sum_{i=1}^{n}\frac{-(1-c)b^2t_i^2+bt_i+2}{b(bt_i+1)} - n\frac{-(1-c)bt_n^2e^{-b(1-c)t_n}}{(bt_n+1)e^{-b(1-c)t_n}-(bt_n+1)^c} \\
&= \lim_{b\to\infty}\sum_{i=1}^{n}\frac{c}{b(bt_i+1)} - \frac{c-2}{b} - (1-c)t_i + n\frac{(1-c)bt_n^2e^{-b(1-c)t_n}}{(bt_n+1)e^{-b(1-c)t_n}-(bt_n+1)^c} \\
&= \lim_{b\to\infty}\sum_{i=1}^{n}-(1-c)t_i + \frac{c}{b(bt_i+1)} - n\frac{c-2}{b} + n\frac{(1-c)bt_n^2}{(bt_n+1)-(bt_n+1)^ce^{b(1-c)t_n}} \\
&= -(1-c)\sum_{i=1}^{n}t_i
\end{aligned} \tag{6.11}$$

This implies

$$\lim_{b\to\infty}\frac{\partial L}{\partial b} = -(1-c)\sum_{i=1}^{n}t_i < 0,$$

since $0 < (1-c)$. Showing that $\frac{\partial L}{\partial b}$ is a decreasing function and that $\lim_{b\downarrow 0}\frac{\partial L}{\partial b} > 0$ holds under certain conditions is not trivial and we leave this for further research. Showing these properties might be slightly similar to the proof for the perfect debugging model Yamada S-shaped due to Knafl and Morgan (1996) and Zhao and Xie (1996), since it is the reduced Delayed S-shaped Pure Error Generation model with $c = 0$.

Chapter 7

Yamada Imperfect Debugging Model 1

Most perfect debugging models have a time dependent error detection rate $b(t)$. Yamada et al. (1992) studied fault generation debugging models with a constant error detection rate $b(t) = b$. He created two models, the first is the Yamada Imperfect Debugging Model 1. This model assumes that:

1. When detected errors are removed, it is possible to introduce new errors
2. The probability of finding an error in a program is proportional to the number of remaining errors in the program

We will use the notation of Chapter 2. Assume that the time-dependent fault content function and error detection rate are

$$a(t) = ae^{kt} \tag{7.1}$$

$$b(t) = b \tag{7.2}$$

respectively, where $a = a(0)$ is the parameter for the total number of initial faults that exist in the software before testing. The mean value function is given by

$$\Lambda(t) = \frac{ab}{k+b}(e^{kt} - e^{-bt}) \tag{7.3}$$

$$\lambda(t) = \frac{ab\left(be^{-bt} + ke^{kt}\right)}{b+k} \tag{7.4}$$

Then, as described in Section 3.1, a can be factored out. This gives a mean value function, that we denote with $\overline{\Lambda}(t)$ and a corresponding intensity function $\overline{\lambda}(t)$.

7.1 Likelihood equations

When substituting the explicit expression for a in (3.2), we can obtain the log-likelihood function expressed in all parameters except a.

$$a = \frac{n}{\overline{\Lambda}(t_n)}. \tag{7.5}$$

$$\sum_{i=1}^{n} \log\left(\frac{b(be^{-bt_i} + ke^{kt_i})}{b+k}\right) + n\log n - n\log\frac{b\left(e^{kt_n} - e^{-bt_n}\right)}{b+k} - n \tag{7.6}$$

Then we can derive the expressions of the derivatives of the likelihood function with respect to all other parameters with Formula (3.6).

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} \frac{e^{-bt_i}\left(b^3(-t_i) + b^2(1-kt_i) + k^2e^{t_i(b+k)} + 2bk\right)}{b(b+k)\left(be^{-bt_i} + ke^{kt_i}\right)} - n\frac{e^{-bt_n}\left(b^2t_n + k\left(e^{t_n(b+k)} + bt_n - 1\right)\right)}{b(b+k)\left(e^{kt_n} - e^{-bt_n}\right)} \tag{7.7}$$

$$\frac{\partial L}{\partial k} = \sum_{i=1}^{n} \frac{e^{-bt_i}\left(k^2t_ie^{t_i(b+k)} + b\left((kt_i+1)e^{t_i(b+k)} - 1\right)\right)}{(b+k)\left(be^{-bt_i} + ke^{kt_i}\right)} - n\frac{e^{-bt_n}\left(e^{t_n(b+k)}(bt_n + kt_n - 1) + 1\right)}{(b+k)\left(e^{kt_n} - e^{-bt_n}\right)} \tag{7.8}$$

From these likelihood equations and the mean value function, we can not see any direct problems concerning the identifiability of the parameters. The existence and uniqueness of the solution of the maximum likelihood Equations is not clear, and needs extensive theoretical research, due to the complexity of the likelihood equations.

7.2 Property of total number of faults

For the Yamada Imperfect Debugging model 1 the nice property that a, the number of initial faults, is always smaller than than the number of faults that are found, does not hold, since:

$$\lim_{t \to \infty} \overline{\Lambda}(t_n) = \infty \tag{7.9}$$

together with (7.5). So while for the Goel-Okumoto model $0 < \overline{\Lambda}(t_n) < 1$, as shown in Section 4.1 for the Yamada Imperfect Debugging Model 1 the mean value function may go to infinity. From the assumptions it is assumed that there can be more mistakes that have to be removed than there are initially visible in the software. So the estimator of a can be smaller than the number of faults found in the system.

Chapter 8

Yamada Imperfect Debugging Model 2

Most perfect debugging models have a time dependent error detection rate $b(t)$. Yamada et al. (1992) studied fault generation debugging models with a constant error detection rate $b(t) = b$. He created two models, the second model is the Yamada Imperfect Debugging Model 2. This model assumes that:

1. When detected errors are removed, it is possible to introduce new errors
2. The probability of finding an error in a program is proportional to the number of remaining errors in the program

We will use the notation of Chapter 2. Assume that the time-dependent fault content function and error detection rate are

$$a(t) = a(1 + ct) \tag{8.1}$$

$$b(t) = b \tag{8.2}$$

respectively, where $a = a(0)$ is the parameter for the total number of initial faults that exist in the software before testing. The mean value function is given by

$$\Lambda(t) = a((1 - e^{-bt})\left(1 - \frac{c}{b}\right) + ct) \tag{8.3}$$

$$\lambda(t) = a(c + b\left(1 - \frac{c}{b}\right)e^{-bt}) \tag{8.4}$$

Then, as described in Section 3.1, a can be factored out. This gives a mean value function, that we denote with $\overline{\Lambda}(t)$ and a corresponding intensity function $\overline{\lambda}(t)$.

8.1 Likelihood equations

When substituting the explicit expression for a in (3.2), we can obtain the log-likelihood function expressed in all parameters except a.

$$a = \frac{n}{\overline{\Lambda}(t_n)}. \tag{8.5}$$

$$\sum_{i=1}^{n} \log\left(b\left(1 - \frac{c}{b}\right)e^{-bt_i} + c\right) + n\log n - n\log(\left(1 - \frac{c}{b}\right)\left(1 - e^{-bt_n}\right) + ct_n) - n \tag{8.6}$$

Then we can derive the expressions of the derivatives of the likelihood function with respect to all other parameters with Formula (3.6).

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} \frac{-bt_i + ct_i + 1}{c\left(e^{bt_i} - 1\right) + b} - n\frac{b^2 t_n + c\left(-bt_n + e^{bt_n} - 1\right)}{b\left(c\left(-e^{bt_n}\right) + b\left(e^{bt_n}(ct_n + 1) - 1\right) + c\right)} \tag{8.7}$$

$$\frac{\partial L}{\partial c} = \sum_{i=1}^{n} \frac{e^{bt_i} - 1}{c\left(e^{bt_i} - 1\right) + b} - n\frac{e^{bt_n}(bt_n - 1) + 1}{c\left(-e^{bt_n}\right) + b\left(e^{bt_n}(ct_n + 1) - 1\right) + c} - \tag{8.8}$$

From these likelihood equations and the mean value function, we can not see any direct problems concerning the identifiability of the parameters. The existence and uniqueness of the solution of the maximum likelihood Equations is not clear, and needs extensive theoretical research, due to the complexity of the likelihood equations.

8.2 Property of total number of faults

For the Yamada Imperfect Debugging model 2 the nice property that a, the number of initial faults, is always smaller than than the number of faults that are found, does not hold, since:

$$\lim_{t\to\infty} \overline{\Lambda}(t_n) = \infty \tag{8.9}$$

together with (8.5). So while for the Goel-Okumoto model $0 < \overline{\Lambda}(t_n) < 1$, as shown in Section 4.1 for the Yamada Imperfect Debugging Model 2 the mean value function may go to infinity. From the assumptions it is assumed that there can be more mistakes that have to be removed than there are initially visible in the software. So the estimator of a can be smaller than the number of faults found in the system.

Chapter 9

Pham Nordmann Zhang Model

The Pham Nordman Zhang (PNZ) model is a model that integrates imperfect debugging with the learning phenomenon. Pham et al. (1999) found that inclusion of both imperfect debugging and a time-dependent fault-detection rate in this model, improves both the descriptive and the predictive properties of a model. They also found that this is worth the extra model-complexity and the increased number of parameters required for a better relative fit. This model assumes that:

1. The introduction rate is a linear time-dependent overall fault content function.
2. The fault detection rate function is non-decreasing time-dependent with an inflection S-shaped model.

We will use the notation of Chapter 2. Assume that the time-dependent fault content function and error detection rate are

$$a(t) = a(1 + ct) \tag{9.1}$$

$$b(t) = \frac{b}{1 + de^{-bt}} \tag{9.2}$$

respectively, where $a = a(0)$ is the parameter for the total number of initial faults that exist in the software before testing, and $\frac{b}{1+d}$ is the initial per fault visibility or failure intensity. The mean value function is given by

$$\Lambda(t) = \frac{a}{1 + de^{-bt}} \left[(1 - e^{-bt}) \left(1 - \frac{c}{d}\right) + ct \right] \tag{9.3}$$

$$\lambda(t) = \frac{a(c + b(1 - \frac{c}{d})\ e^{-bt})}{1 + de^{-bt}} + \frac{abde^{-bt} \left[(1 - \frac{c}{d}) \left(1 - e^{-bt}\right) + ct \right]}{(1 + de^{-bt})^2} \tag{9.4}$$

This can be simplified to

$$\lambda(t) = \frac{ae^{bt} \left(b \left(c \left(d^2 t - d - 1\right) + d(d+1)\right) + cd \left(e^{bt} + d\right)\right)}{d \left(e^{bt} + d\right)^2} \tag{9.5}$$

Then, as described in Section 3.1, a can be factored out. This gives a mean value function, that we denote with $\overline{\Lambda}(t)$ and a corresponding intensity function $\overline{\lambda}(t)$.

An interesting extension of this model is the discrete version, described by Edris and Shatnawi (2011). These discrete versions are mathematically complex. Due to the software reliability data being discrete, discrete models may provide a better fit than their continuous time counterparts.

9.1 Likelihood equations

When substituting the explicit expression for a in (3.2), we can obtain the log-likelihood function expressed in all parameters except a.

$$a = \frac{n}{\overline{\Lambda}(t_n)}. \tag{9.6}$$

$$L = n\log(n) - n\log((1+e^{-bt_n})(1-\frac{c}{d}) + ct_n) + n\log(1+de^{-bt_n}) - n + \sum_{i=1}^{n} bt_i + \log\left(b(c(d^2t_i - d - 1) + d(d+1)) + cd(e^{bt_i}+d)\right) - \log(d(e^{bt_i}+d)^2) \tag{9.7}$$

Then we can derive the expressions of the derivatives of the likelihood function with respect to all other parameters with Formula (3.6).

$$\begin{aligned}\frac{\partial L}{\partial b} = \sum_{i=1}^{n} &\frac{d(d+1)\left(bdt_i + e^{bt}(1-bt_i) + d\right) - c\left(d^3(-t_i)(bt_i+2) + d^2\left(-2te^{bt_i} + bt_i\left(t_ie^{bt_i}+1\right)+1\right) + d\left(bt_i + e^{bt_i}(1-bt_i)+1\right) + e^{bt_i}(1-bt_i)\right)}{\left(e^{bt_i}+d\right)\left(b\left(c\left(d^2t_i - d - 1\right) + d(d+1)\right) + cd\left(e^{bt_i}+d\right)\right)} \\ &-n\frac{t_ne^{bt_n}\left(c\left(d^2t_n - d - 1\right) + d(d+1)\right)}{\left(e^{bt_n}+d\right)\left(c\left(e^{bt_n}(dt_n-1)+1\right) + d\left(e^{bt_n}-1\right)\right)}\end{aligned} \tag{9.8}$$

$$\frac{\partial L}{\partial c} = \sum_{i=1}^{n} \frac{b\left(d^2t_i - d - 1\right) + d\left(e^{bt_i}+d\right)}{b\left(c\left(d^2t_i - d - 1\right) + d(d+1)\right) + cd\left(e^{bt_i}+d\right)} - n\frac{e^{bt_n}(dt_n-1)+1}{c\left(e^{bt_n}(dt_n-1)+1\right) + d\left(e^{bt_n}-1\right)} \tag{9.9}$$

$$\begin{aligned}\frac{\partial L}{\partial d} = \sum_{i=1}^{n} &\frac{b\left(c\left(d^2\left(t_ie^{bt_i}+2\right) + e^{bt_i} + d^3(-t_i) + 3d\right) - d^2\left(-e^{bt_i}+d+2\right)\right) - cd^2\left(e^{bt_i}+d\right)}{d\left(e^{bt_i}+d\right)\left(b\left(c\left(d^2t_i - d - 1\right) + d(d+1)\right) + cd\left(e^{bt_i}+d\right)\right)} \\ &-n\frac{c\left(d^2t_n\left(-e^{bt_n}\right) + 2d\left(e^{bt_n}-1\right) + e^{bt_n}\left(e^{bt_n}-1\right)\right) - d^2\left(e^{bt_n}-1\right)}{d\left(e^{bt_n}+d\right)\left(c\left(e^{bt_n}(dt_n-1)+1\right) + d\left(e^{bt_n}-1\right)\right)}\end{aligned} \tag{9.10}$$

From these likelihood equations and the mean value function, we can not see any direct problems concerning the identifiability of the parameters. The existence and uniqueness of the solution of the maximum likelihood Equations is not clear, and needs extensive theoretical research, due to the complexity of the likelihood equations.

9.2 Property of total number of faults

For the PNZ model the nice property that a, the number of initial faults, is always smaller than than the number of faults that are found, does not hold, since:

$$\lim_{t\to\infty} \overline{\Lambda}(t_n) = \infty. \tag{9.11}$$

together with (9.6). So while for the Goel-Okumoto model $0 < \overline{\Lambda}(t_n) < 1$, as shown in Section 4.1 for the PNZ model the mean value function may go to infinity. From the assumptions it is assumed that there can be more mistakes that have to be removed than there are initially visible in the software. So the estimator of a can be smaller than the number of faults found in the system.

Chapter 10

Pham Exponential Imperfect Model

This model is very similar to the PNZ model (Chapter 9). With this model the fault introduction rate is an exponential function of the testing time (Pham, 2000). This model assumes that:

1. The introduction rate is an exponential function of testing time
2. The error detection rate function is non-decreasing with an inflection S-shaped model

We will use the notation of Chapter 2. Assume that the time-dependent fault content function and error detection rate are

$$a(t) = ae^{kt} \tag{10.1}$$

$$b(t) = \frac{b}{1 + de^{-bt}} \tag{10.2}$$

respectively, where $a = a(0)$ is the parameter for the total number of initial faults that exist in the software before testing, and $\frac{b}{1+d}$ is the initial per fault visibility or failure intensity. The mean value function is given by

$$\Lambda(t) = \frac{ab}{b+k}\left(\frac{e^{(k+b)t} - 1}{e^{bt} + d}\right) \tag{10.3}$$

$$\lambda(t) = \frac{abe^{t(b+k)}}{e^{bt} + d} - \frac{ab^2 e^{bt}\left(e^{t(b+k)} - 1\right)}{(b+k)\left(e^{bt} + d\right)^2} \tag{10.4}$$

This can be simplified to

$$\lambda(t) = \frac{abe^{bt}\left(bde^{kt} + ke^{kt}\left(e^{bt} + d\right) + b\right)}{(b+k)\left(e^{bt} + d\right)^2} \tag{10.5}$$

Then, as described in Section 3.1, a can be factored out. This gives a mean value function, that we denote with $\overline{\Lambda}(t)$ and a corresponding intensity function $\overline{\lambda}(t)$.

10.1 Likelihood equations

When substituting the explicit expression for a in (3.2), we can obtain the log-likelihood function expressed in all parameters except a.

$$a = \frac{n}{\overline{\Lambda}(t_n)}. \tag{10.6}$$

$$L = n \log n - n \log(b(e^{(k+b)t_n} - 1)) + n \log((b+k)(e^{bt_n} + d)) - n + \sum_{i=1}^{n} bt_i + \log(b) + \log(bde^{kt_i} + ke^{kt_i}(e^{bt_i} + d) + b) - \log((b+k)(e^{bt_i} + d)^2) \tag{10.7}$$

Then we can derive the expressions of the derivatives of the likelihood function with respect to all other parameters with Formula (3.6).

$$\begin{aligned}\frac{\partial L}{\partial b} = \sum_{i=1}^{n} & \frac{b^3 t_i \left(d - e^{bt_i}\right)\left(de^{kt_i} + 1\right) + b^2\left(d\left(e^{t_i(b+k)} + kt_i + 1\right) + e^{bt_i}(1 - kt_i) + d^2 e^{kt_i}(2kt_i + 1)\right) + k^2 e^{kt_i}\left(e^{bt_i} + d\right)^2 + bk\left(e^{bt_i} + d\right)\left(de^{kt_i}(kt_i + 2) + 2\right)}{b(b+k)\left(e^{bt_i} + d\right)\left(bde^{kt_i} + ke^{kt_i}\left(e^{bt_i} + d\right) + b\right)} \\ & - n\frac{d\left(b^2 t_n e^{t_n(b+k)} + k\left((bt_n + 1)e^{t_n(b+k)} - 1\right)\right) + e^{bt_n}\left(b^2 t_n + k\left(e^{t_n(b+k)} + bt_n - 1\right)\right)}{b(b+k)\left(e^{bt_n} + d\right)\left(e^{t_n(b+k)} - 1\right)}\end{aligned} \tag{10.8}$$

$$\frac{\partial L}{\partial k} = \sum_{i=1}^{n} \frac{b^2 dte^{kt_i} + k^2 t_i e^{kt_i}\left(e^{bt_i} + d\right) + b\left((kt_i + 1)e^{t_i(b+k)} + 2dkt_i e^{kt_i} - 1\right)}{(b+k)\left(bde^{kt_i} + ke^{kt_i}\left(e^{bt_i} + d\right) + b\right)} - n\frac{e^{t_n(b+k)}(bt_n + kt_n - 1) + 1}{(b+k)\left(e^{t_n(b+k)} - 1\right)} \tag{10.9}$$

$$\frac{\partial L}{\partial d} = \sum_{i=1}^{n} \frac{b\left(e^{t_i(b+k)} + d\left(-e^{kt_i}\right) - 2\right) - ke^{kt_i}\left(e^{bt_i} + d\right)}{\left(e^{bt_i} + d\right)\left(bde^{kt_i} + ke^{kt_i}\left(e^{bt_i} + d\right) + b\right)} + n\left(\frac{1}{e^{bt_n} + d}\right) \tag{10.10}$$

From these likelihood equations and the mean value function, we can not see any direct problems concerning the identifiability of the parameters. The existence and uniqueness of the solution of the maximum likelihood Equations is not clear, and needs extensive theoretical research, due to the complexity of the likelihood equations.

10.2 Property of total number of faults

For the Pham Exponential Imperfect model the nice property that a, the number of initial faults, is always smaller than than the number of faults that are found, does not hold, since:

$$\lim_{t \to \infty} \overline{\Lambda}(t_n) = \infty. \tag{10.11}$$

together with (10.6). So while for the Goel-Okumoto model $0 < \overline{\Lambda}(t_n) < 1$, as shown in Section 4.1 for the Pham Exponential Imperfect model the mean value function may go to infinity. From the assumptions it is assumed that there can be more mistakes that have to be removed than there are initially visible in the software. So the estimator of a can be smaller than the number of faults found in the system.

Chapter 11

Results

In this chapter we will investigate the existence and uniqueness of the maximum likelihood Estimators for the Pure Error Generation Model, Yamada Imperfect Debugging Model 1, Yamada Imperfect Debugging Model 2 and the Delayed S-shaped Pure Error Generation Model. We will do this by visualizing the likelihood equation and the derivatives of the likelihood equation to the parameters for each distinct model for a given dataset. First we investigate the data and check the conditions for reliability growth. Then we will show the approach of this visualization by performing it on the perfect debugging models Goel-Okumoto and Yamada S-shaped. Subsequently we will investigate the different 3-parameter models. As shown in Section 3.1 a can be factored out and solved explicitly in terms of the other parameters in all researched models, so it suffices to look only at the other two parameters.

11.1 Dataset

The software data set was extracted from information about failures in the development of software for the real-time multi-computer complex of the US Naval Fleet Programming Center of the US Naval Tactical Data Systems (Goel and Okumoto, 1979). The time horizon is divided in four phases: the production phase, test phase, user phase and subsequent test phase. In this analysis we use the first 27 software failures that were found during production phase and the first test phase, like in Pham (2006)[Chapter 6]. This data set is ungrouped. We can check whether the data satisfies the necessary and sufficient condition for the existence of a unique, positive and finite solution of maximum likelihood equations for the models assuming perfect repair, namely the Goel-Okumoto model and Yamada S-Shaped model. Indeed, condition (4.1) is satisfied since $t_n = 337$, $\sum_{i=1}^{n} t_i = 2829$ and $(2\sum_{i=1}^{n} t_i)/n = 209.5$. From this it follows $337 > 209.5$ which satisfies the condition. The condition (4.2) is satisfied since $(\frac{3}{2}\sum_{i=1}^{n} t_i)/n = 157.2$ an thus $337 > 157.2$. We can also check the TTT plot of the data as described in Section 3.2. This is shown in Figure 11.1. Indeed the graph is below the diagonal so this gives us no reason to believe there is no reliability growth in this data set.

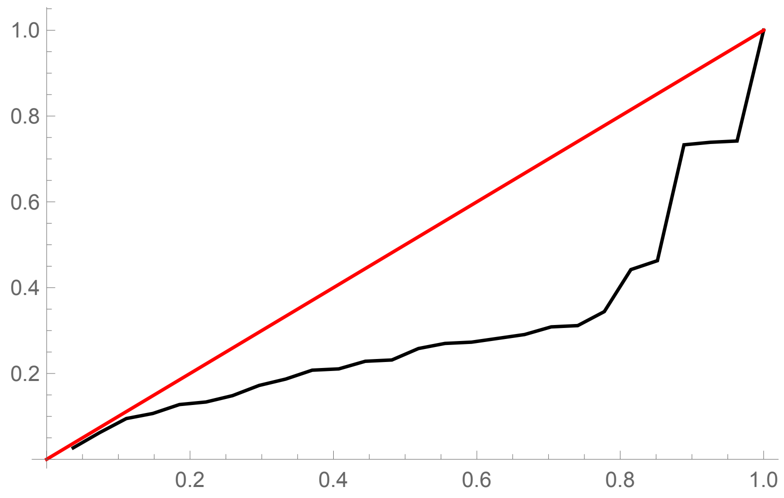
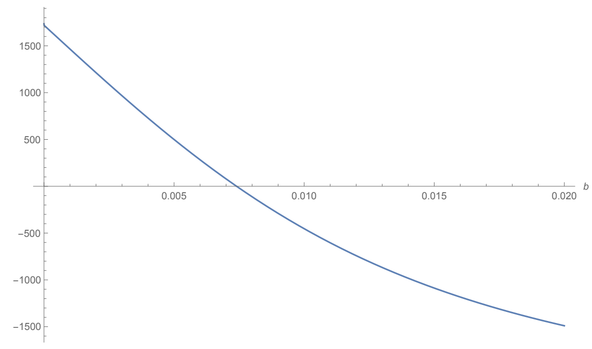


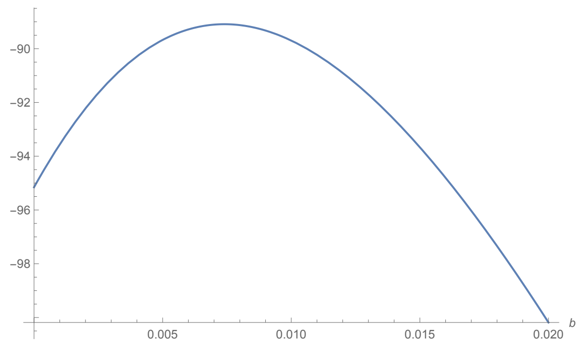
Figure 11.1: TTT plot US Naval data

11.2 Goel-Okumoto Model (perfect repair)

The existence and uniqueness of the maximum likelihood estimators for the Goel-Okumoto model has clear conditions (Knafl and Morgan, 1996). In the previous section we have shown that the software failure data set satisfies these conditions. This means that maximizing the likelihood function and finding the root of the derivative to b of the likelihood should give the same, unique estimate for b. This b subsequently uniquely defines an estimate for a. In Figure 11.2 we see that this is indeed the case. If we determine the root of the derivative of the log-likelihood to b we find $\hat{b} = 0.007402$. When we determine the maximum of the log-likelihood we get the same value for $\hat{b}$.



(a) Derivative of Log-Likelihood to b

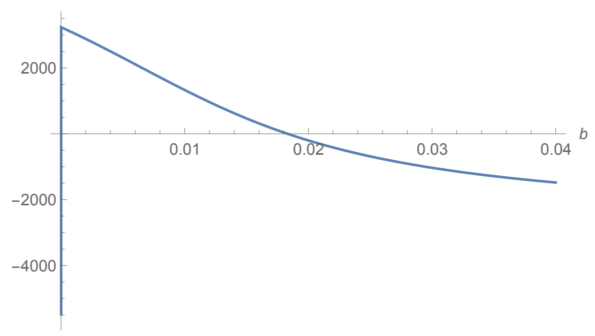


(b) Log-Likelihood

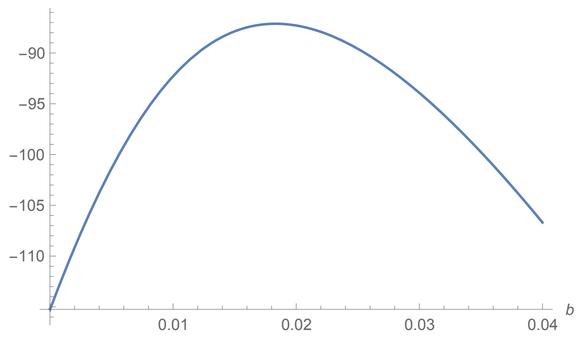
Figure 11.2: Goel-Okumoto model equations plotted to data

11.3 Yamada S-shaped Model (perfect repair)

The existence and uniqueness of the maximum likelihood estimators for the Yamada S-shaped model has clear conditions for ungrouped data (Knafl and Morgan, 1996). In the previous section we have shown that the software failure data set satisfies these conditions. This means that maximizing the likelihood function and finding the root of the derivative to b of the likelihood should give the same, unique estimate for b. This b subsequently uniquely defines an estimate for a. In Figure 11.3 we see that this is indeed the case. If we determine the root of the derivative of the log-likelihood to b we find $\hat{b} = 0.01831$. When we determine the maximum of the log-likelihood we get the same value for $\hat{b}$.



(a) Derivative of Log-Likelihood to b



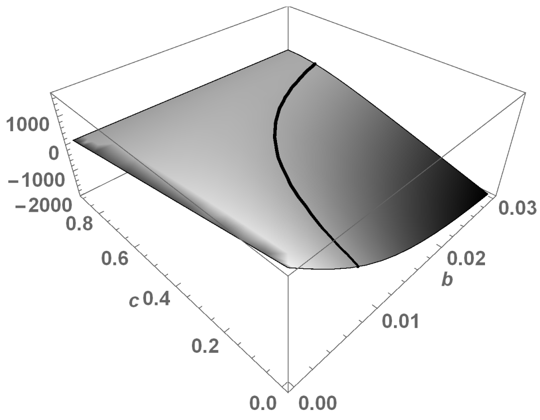
(b) Log-Likelihood

Figure 11.3: Yamada S-shaped model equations plotted to data

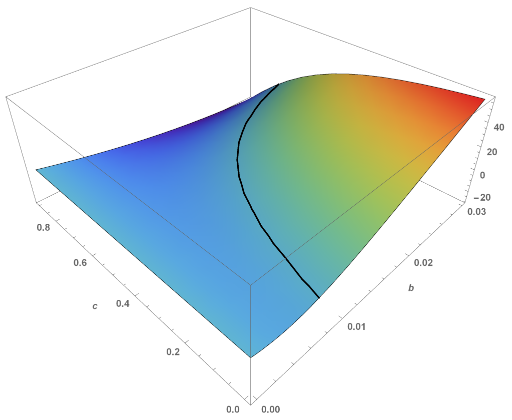
For the models with three parameters we can use this type of visualization in 3D. With this we can compare the root(s) of the derivatives of the likelihood equations to the parameters with the maxima of the likelihood equations.

11.4 Pure Error Generation Model

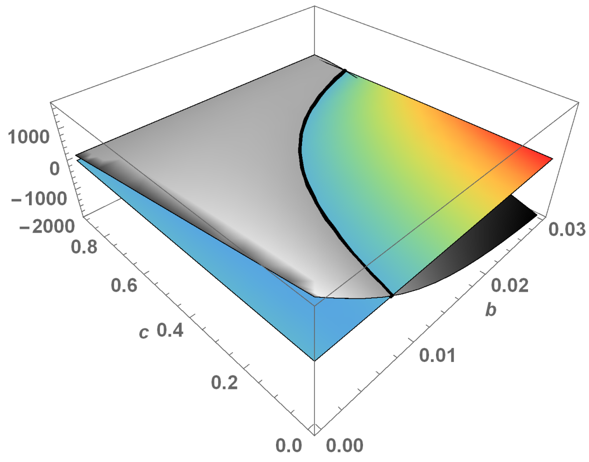
For the Pure Error Generation model of Chapter 5 we use (5.6), (5.7) and (5.8) analyse the behaviour of the maximum likelihood equations. From Section 5.1, we know that this model has a non-identifiable parameter. Figure 11.4 represents these equations for this data set. In Figure 11.4a and 11.4b the intersection of the plane and $z = 0$ is shown by the black curve. It is clear from Figure 11.4c that the two derivatives of the likelihood equations do not give a unique solution, since both planes are equal to 0 on a curve rather than at a point. This is supported by Figure 11.4d, where we can see that the maximum of the likelihood equation is indeed this curve. We do see that when we choose or know c, there is a unique solution for b and thus there exists an estimator for b. This supports the remarks in Section 5.1.



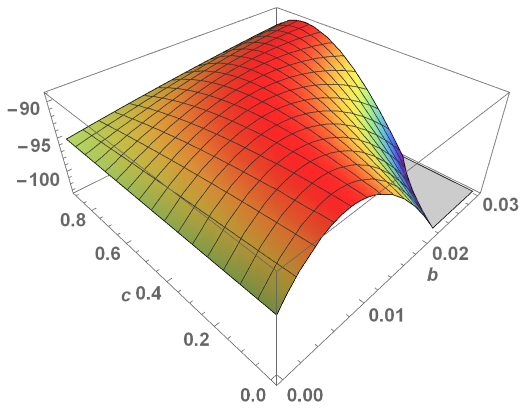
(a) Derivative of Log-Likelihood to b



(b) Derivative of Log-Likelihood to c



(c) Combination of derivative of Log-Likelihood to b and c

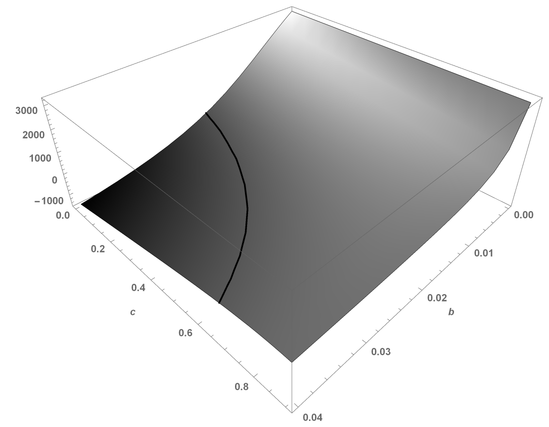


(d) Log-Likelihood

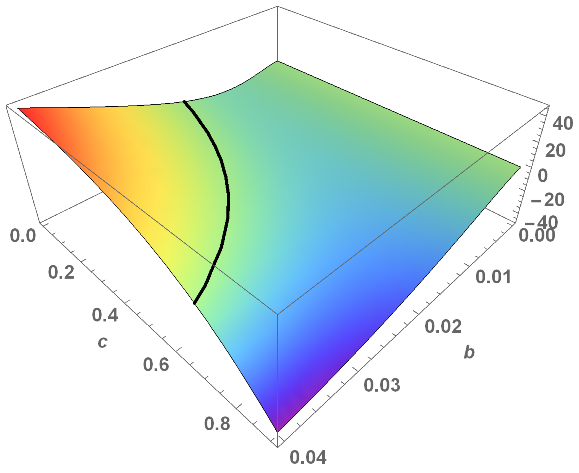
Figure 11.4: Pure Error Generation model Likelihood equations and derivatives based on data

11.5 Delayed S-shaped Pure Error Generation Model

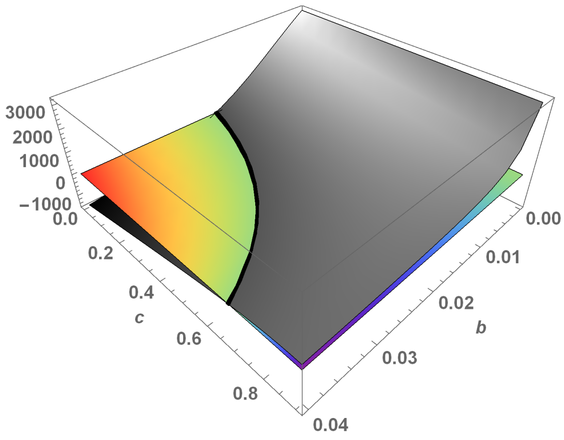
For the Delayed S-shaped Pure Error Generation Model we use (6.7), (6.8) and (6.9) to check whether the maximum likelihood equations have a unique solution and whether it has a non-identifiable parameter. Figure 11.5 represents these equations for this data set. In Figure 11.5a and 11.5b the intersection of the plane and $z = 0$ is shown by the black curve. It is clear from Figure 11.5c that the two derivatives of the likelihood equations do not give a unique solution, since both planes are equal to 0 on a curve rather than a point. We do see that when we choose or know c, there is a unique solution for b and thus there exists an estimator for b. This is supported by Figure 11.5d, where we can see by that the maximum of the likelihood equation is indeed this same curve. From this data analysis it seems that the Delayed S-shaped Pure Error Generation Model has a non-identifiable parameter. This is not clear from the mean value function (Section 6.1) and needs further investigation.



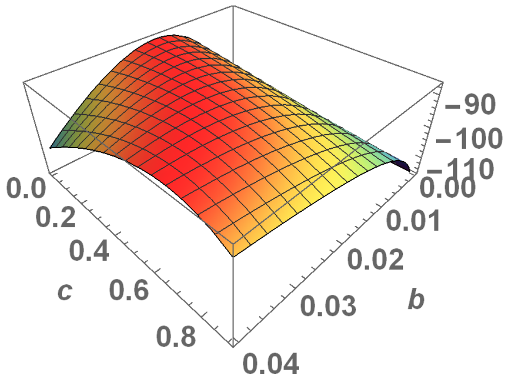
(a) Derivative of Log-Likelihood to b



(b) Derivative of Log-Likelihood to c



(c) Combination of derivative of Log-Likelihood to b and c

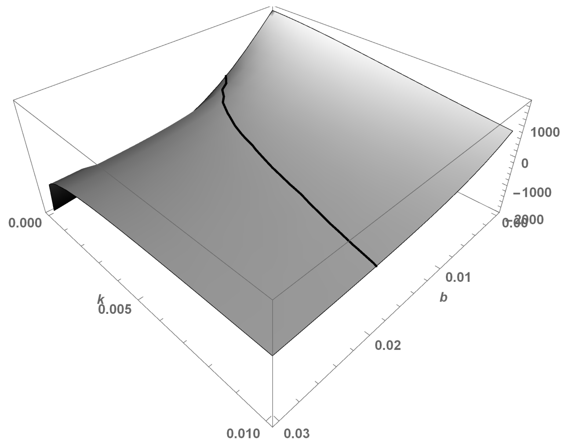


(d) Log-Likelihood

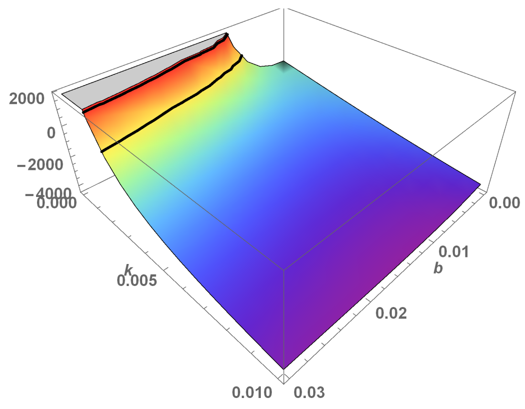
Figure 11.5: Delayed S-shaped Pure Error Generation model, Likelihood equations and derivatives based on data

11.6 Yamada Imperfect Debugging Model 1

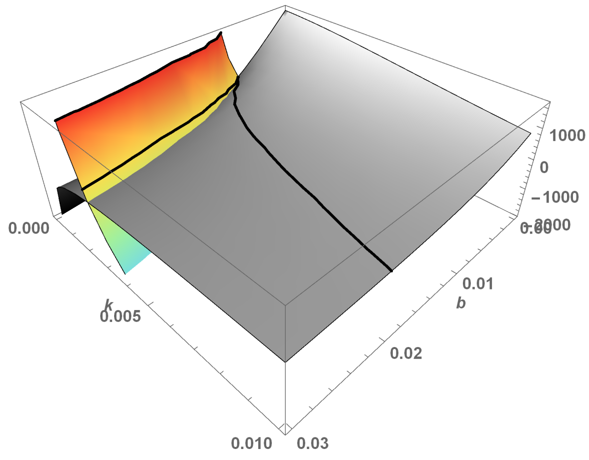
For the Yamada Imperfect Debugging Model 1 we use (7.6), (7.7) and (7.8) to check whether the maximum likelihood equations have a unique solution. Figure 11.6 represents these equations for this data set. In Figure 11.6a and 11.6b the intersection of the plane and $z = 0$ is shown by the black curve. For these equations we see that there possibly is a unique solution to setting the derivatives of the likelihood equations equal to zero. This is where the two black lines possibly intersect, near $k = 0$. The plot of the likelihood supports this. In Figure 11.7 we zoom in on the point of intersect. It appears that the solution of the maximum likelihood Equations is in $k = 0$. This reduces the model to the Goel-Okumoto model, as described in Section 4.2. In order to investigate this model further, different datasets have to be used, where k might have an influence. This analysis does not raise any problems concerning identifiability of the parameters.



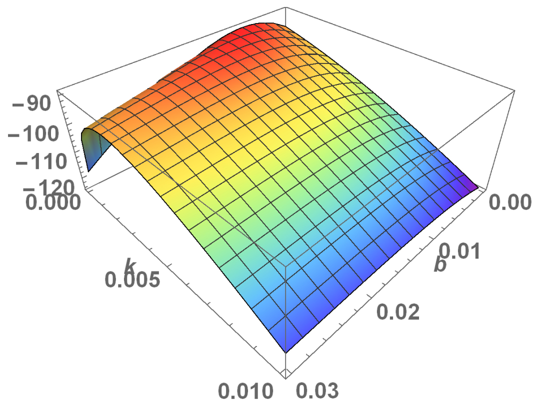
(a) Derivative of Log-Likelihood to b



(b) Derivative of Log-Likelihood to k



(c) Combination of derivative of Log-Likelihood to b and k



(d) Log-Likelihood

Figure 11.6: Yamada Imperfect Debugging model 1, Likelihood equations and derivatives based on data

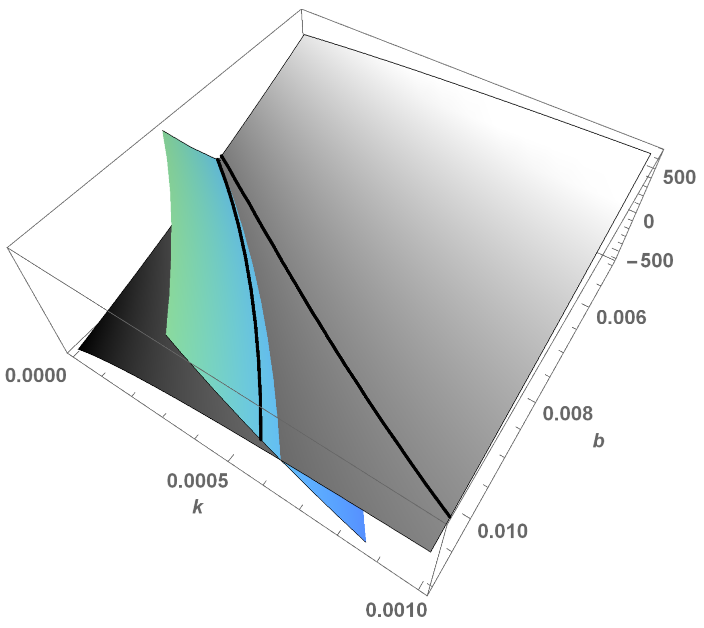
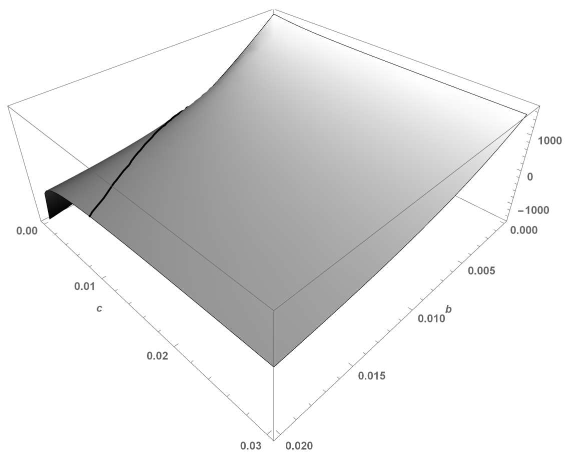


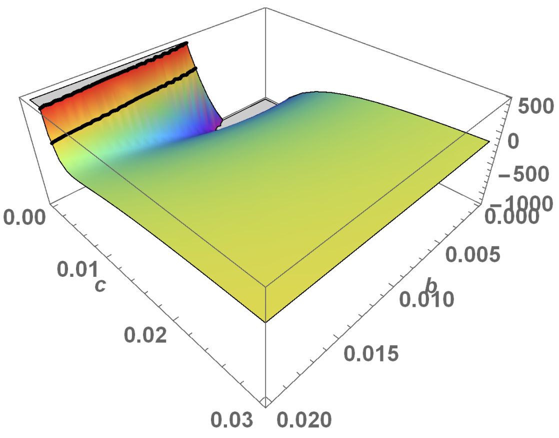
Figure 11.7: Yamada Imperfect Debugging model 1, zoom derivatives maximum likelihood Equations

11.7 Yamada Imperfect Debugging Model 2

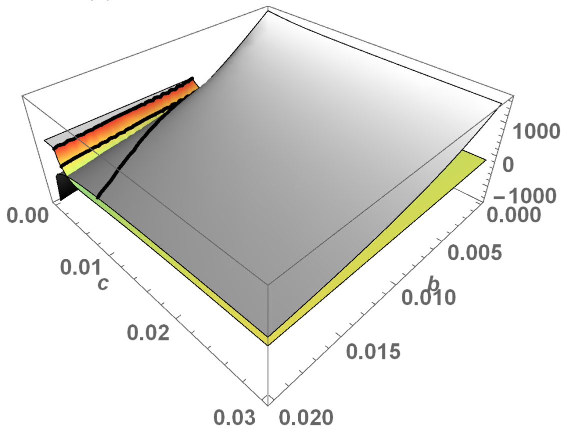
For the Yamada Imperfect Debugging Model 2 we use (8.6), (8.7) and (8.8)to check whether the maximum likelihood equations have a unique solution. Figure 11.6 represents these equations for data set of Section 11.1. In Figure 11.8a and 11.8b the intersection of the plane and $z = 0$ is shown by the black curve. For these equations we see that there possibly is a unique solution to setting the derivatives of the likelihood equations equal to zero. This is where the two black lines possibly intersect, near $c = 0$. The plot of the likelihood supports this. In Figure 11.9 we zoom in on the point of intersect. It appears that the solution of the maximum likelihood Equations is in $c = 0$. This reduces the model to the Goel-Okumoto model, as described in Section 4.2. In order to investigate this model further, different datasets have to be used, where c might have an influence. This analysis does not raise any problems concerning identifiability of the parameters.



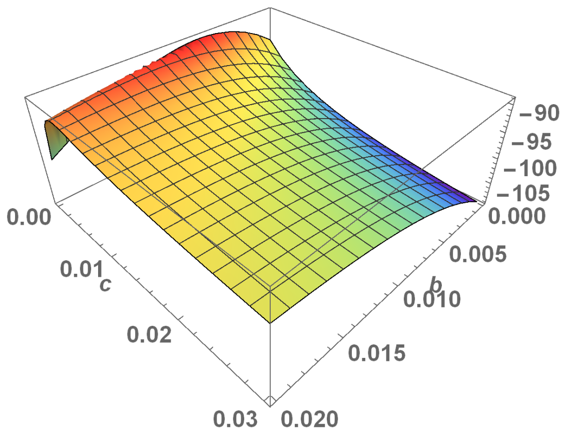
(a) Derivative of Log-Likelihood to b



(b) Derivative of Log-Likelihood to c



(c) Combination of derivative of Log-Likelihood to b and c



(d) Log-Likelihood

Figure 11.8: Yamada Imperfect Debugging model 2, Likelihood equations and derivatives based on data

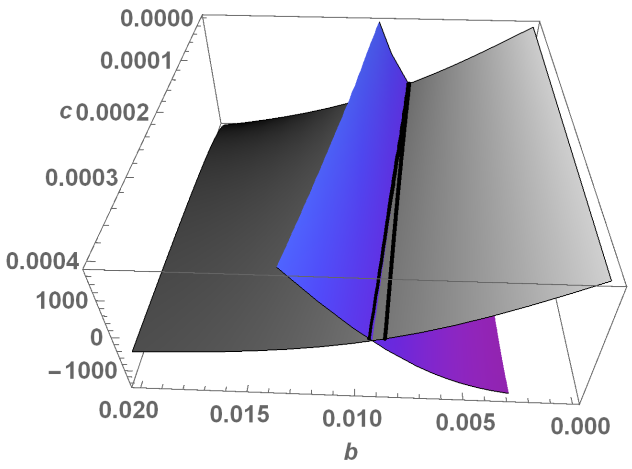


Figure 11.9: Yamada Imperfect Debugging model 2, zoom derivatives maximum likelihood Equations

Chapter 12

Conclusion

In this report we studied imperfect debugging NHPP models in software reliability engineering. A clear distinction must be made between imperfect fault debugging models and error generation models. In this report we treated both error generation and imperfect fault debugging models. We investigated existence and uniqueness of the maximum likelihood estimators and the identifiability of the parameters for the following models. We will first sum up the major conclusions of this report and then describe into detail.

- The Pure Error Generation Model has a non-identifiable parameter, which is not mentioned in literature.
- Conditions were derived for the existence of a unique positive solution of the MLE for the Pure Error Generation model with c known.
- The mean value function of the fault detection process of The Pure Imperfect Fault Debugging Model is similar to the mean value function of the Pure Imperfect Fault Debugging Model with $p = 1 - c$.
- We make a conjecture that the Delayed S-shaped Pure Error Generation Model also has a non-identifiable parameter, which is supported by the numerical analysis in Chapter 11.
- No problems with identifiability were raised for the Yamada Imperfect Debugging Model 1 and Yamada Imperfect Debugging Model 1.

In Chapter 4 we presented an overview of perfect, imperfect fault debugging and error generation debugging models. The connection between the models was studied there and concludes that all models in this report, reduce to a perfect debugging model, when choosing one or two specific parameters equal to zero.

In Chapter 5 we found that the Pure Error Generation Model has a non-identifiable parameter, which means that the model can only be used when one parameter (b or c) is known. This aspect is not mentioned in the literature, but is really important when using this model, since it requires insight in the value of c. We derived conditions for existence of the maximum likelihood equations with c known.

In Chapter 6 we prove the limit to infinity of the derivitave to b of the likelihood equations is negative. It remains to derive conditions on the existence of the maximum likelihood equations with c known for the Delayed S-shaped Pure Error Generation Model. This needs more research.

We derived the likelihood equations for the Yamada Imperfect Debugging Model 1 (Chapter 7), Yamada Imperfect Debugging Model 2 (Chapter 8), PNZ Model (Chapter 9) and the Pham Exponential Imperfect Model (Chapter 10). These likelihood equations are quite complex, especially for the PNZ Model and the Pham Exponential Imperfect Model. It is therefore not certain that conditions can be derived explicitly as is the case for perfect models.

No conjectures of non-identifiability of parameters were raised with the Yamada Imperfect Debugging Model 1 , the Yamada Imperfect Debugging Model 2, the PNZ Model and the Pham Exponential Imperfect Model. For the Yamada Imperfect Debugging Model 1 and Yamada Imperfect Debugging Model 2 this is supported by the numerical analysis.

In Chapter 11 we analysed the maximum likelihood equations of the 3-parameter models with a data set. This confirmed our findings for the Pure Error Generation Model and supported the conjecture on the non-identifiability of the Delayed S-shaped Pure Error Generation Model. This needs more research since there was no theoretical proof for this. For the Yamada Imperfect Debugging Model 1 and the Yamada Imperfect Debugging Model 2 more numerical analysis should be performed, since in this analysis there seemed to be no contribution of the error generation rates, parameter k or c, respectively.

Chapter 13

Future Research

In this chapter we describe aspects of this report which require further research.

For all models mentioned in this report, more investigation is needed to obtain necessary and sufficient conditions on the existence of a unique and positive solution of the maximum likelihood estimators. For the Delayed S-shaped Pure Error Generation Model one part is done. It remains to show that the derivative to b is decreasing and positive near zero.

Another aspect that needs to be shown is the non-identifiability of the parameters of the MLE for the Delayed S-shaped Error Generation Model. From the numerical analysis this seems clear but the theoretical proof remains open.

The Yamada Imperfect Debugging Model 1 and the Yamada Imperfect Debugging Model 2 require more numerical analysis since the contribution of the error generation rate for this specific dataset was found to be zero.

The PNZ Model and the Pham Exponential Debugging model can be researched even more. Since they are 4-parameter models, no simple visualization is possible in numerical analysis.

Knafl and Morgan (1996) and Meyfroyt (2012) use a simple separation technique using monotonicity. We expect that this will not work for the more complicated models treated here. Heavier tools are needed. Scholz (1996) give an accessible proof of a special application of Morse Theory which might be a very useful theory to address the uniqueness of maximum likelihood Estimators for models with two or more parameters. There also local concavity of the log-likelihood function is described as a way to pose uniqueness arguments.

Bibliography

R. E. Barlow and B. Davis. Analysis of time between failures for repairable components. Technical report, California University Berkeley Operations Research Center, 1977.

K.-Y. Cai, D.-B. Hu, C.-G. Bai, H. Hu, and T. Jing. Does software reliability growth behavior follow a non-homogeneous Poisson process. *Information and Software Technology*, 50(12):1232–1247, 2008.

K. Edris and O. Shatnawi. The Pham Nordmann Zhang (PNZ) software reliability model revisited. In *Proc. of the Tenth IASTED International Conference on Software Engineering, Innsbruck, Austria*, pages 15–17, 2011.

A.L. Goel. Software reliability models: Assumptions, limitations, and applicability. *IEEE Trans. Soft. Eng.*, 11 (12):1411–1423, 1985.

A.L. Goel and K. Okumoto. Time-dependent error detection rate model for software reliability and other performance measures. *IEEE Trans. Rel.*, R-28:206–211, 1979.

D. Hanagal and N. Bhalerao. Modeling and statistical inference on generalized inverse Weibull software reliability growth model. *Journal of the Indian Society for Probability and Statistics*, 17(2):145–160, 2016.

S.A. Hossain and R.C. Dahiya. Estimating the parameters of a non-homogeneous Poisson-process model for software reliability. *IEEE Trans. Rel.*, 42(4):604–612, 1993.

P. K. Kapur and R. B. Garg. Optimal sofware release policies for software reliability growth models under imperfect debugging. *RAIRO-Operations Research*, 24(3):295–305, 1990.

P.K. Kapur, O. Singh, and A. Gupta. Some modeling peculiarities in software reliability. *Quality, reliability and infocom technology, trends and future directions. Narosa Publications Pvt. Ltd., New Delhi*, pages 20–34, 2005.

P.K. Kapur, H. Pham, A. Gupta, and P.C. Jha. *Software Reliability Assessment with OR Applications*. Springer, London, 2011.

V.S. Kharchenko, O.M. Tarasyuk, V.V. Sklyar, and V.Yu. Dubnitsky. The method of software reliability growth models choice using assumptions matrix. In *COMPSAC '02: Proceedings of the 26th International Computer Software and Applications Conference on Prolonging Software Life: Development and Redevelopment*, pages 541–546, Washington, DC, USA, 2002. IEEE Computer Society.

T. M. Khoshgoftaar and T. G. Woodcock. Software reliability model selection: a case study. In *Software Reliability Engineering*, pages 183–191. IEEE, 1991.

G.J. Knafl. Solving Maximum Likelihood equations for two-parameter software reliability models for using grouped data. In *Proceedings of the International Symposium on Software Reliability Engineering*, pages 205–213. IEEE, 1992.

G.J. Knafl and J. Morgan. Solving ML equations for 2-parameter Poisson-process models for ungrouped software-failure data. *IEEE Trans. Rel.*, 45(1):42–53, 1996.

S.L.P. Koh. Weighted least-square esimate for software error intensity. *Journal of the Chinese Institute of Industrial Engineers*, 25(2):162–173, 2008.

M.E. Kuhl, H. Damerdji, and J.R. Wilson. Least Squares estimation of nonhomogeneous Poisson processes. In D.J. Medeiros, E.F. E.F. Watson, J.S. Carson, and M.S. Manivannan, editors, *Proceedings of the 1998 Winter Simulation Conference*, pages 637–645, 1998.

A. Kumar. Software reliability growth models, tools and data sets-a review. In *Proceedings of the 9th India Software Engineering Conference*, ISEC '16, pages 80–88, New York, NY, USA, 2016. ACM.

J. T. Kvaløy and B.H. Linqvist. TTT-based tests for trend in repairable systems data. *Rel. Eng. System Safety*, 60:13–28, 1998.

C.-T. Lin, K.-W. Tang, J.-R. Chang, and C.-Y. Huang. An investigation into whether the NHPP framework is suitable for software reliability prediction and estimation. In *2010 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 626–630. IEEE, 2010.

P.H.A. Meyfroyt. Parameter estimation for software reliability models. Master's thesis, Eindhoven University of Technology, Department of Mathematics and Computer Science, 2012.

M. Ohba and X-M Chou. Does imperfect debugging affect software reliability growth? In *Proceedings of the 11th international conference on Software engineering*, pages 237–244. ACM, 1989.

H. Pham. *Software Reliability.* Springer, 2000.

H. Pham. *System Software Reliability.* Springer Series in Reliability Engineering. Springer, London, 2006.

H. Pham, L. Nordmann, and X. Zhang. A general imperfect-software-debugging model with S-shaped fault-detection rate. *IEEE Trans. Rel.*, 48(2):169–175, 1999.

S.E. Rigdon and A.P. Basu. *Statistical Methods for the Reliability of Repairable Systems.* Wiley, 2000.

F. Scholz. Maximum likelihood estimation for type I censored Weibull data including covariates. In *ISSTECH-96-022, Boeing Information & Support Services, PO Box 24346, MS-7L-22*, 1996.

W.A. Thompson, Jr. On the foundations of reliability. *Technometrics*, 23(1):1–13, 1981.

S. Yamada, M. Ohba, and S. Osaki. S-shaped software reliability growth models and their applications. *IEEE Transactions on Reliability*, R-33(4):289–292, 1984.

S. Yamada, K. Tokuno, and S. Osaki. Imperfect debugging models with fault introduction rate for software reliability assessment. *International Journal of Systems Science*, 23(12):2241–2252, 1992.

M. Zhao and M. Xie. On Maximum Likelihood estimation for a general non-homogeneous Poisson process. *Scand. J. Statist.*, 23(4):597–607, 1996.