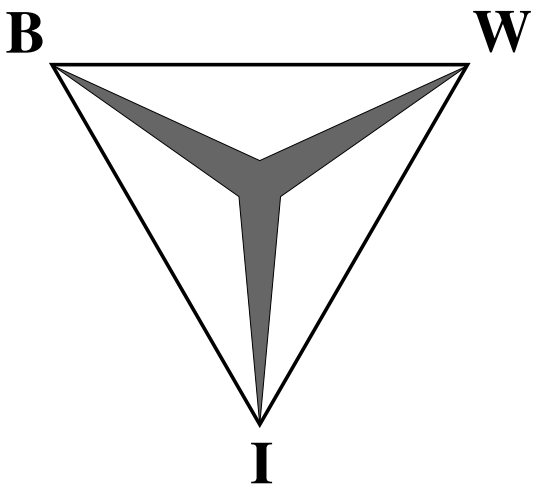
Modeling the intensity of corrective software maintenance after date of release



Tonny Verbaken

Vrije Universiteit, Division of Mathematics and Computer Science
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
amverbak@cs.vu.nl

May 16, 2002

Preface

This paper is my 'BWI-werkstuk' and is part of my study 'Business Mathematics and Informatics' (BWI) at the Free University (VU) in Amsterdam. The paper is one of the subjects in the fourth and last year of the study and is the result of a small research about modeling the intensity of corrective software maintenance. This means I try to model the arrivals of errors after the release of the product. With this model one then can try to predict the arrival of errors of a future release.

The subject is a result of the internship I also did for my study. That internship included a research for Baan and there I was introduced to the problem of customers who find errors in a release of a product and ask for a solution at the 'Service'-department of the company. A prediction of the arrival of those errors could help the 'Service'-department to optimize the service at a acceptable price. During the internship the main goal was to find a model and get practical results. After the internship I was still interested in some further mathematical aspects of the subject. One aspect was thus the modeling of the intensity of corrective software maintenance.

To write this paper I had some great help from Geurt Jongbloed. He is the man who helped me during my internship as well as with writing this paper. I would like to thank him here for all the time he put in my subject and the advice that he gave me.

Tonny Verbaken
May 2002

Contents

Chapter 1

Introduction

This paper describes the research of modeling the intensity of corrective software maintenance. In this chapter there first will be given an introduction to the subject. There will be described what the problem looks like and which part has been investigated here.

1.1 The problem

Software companies produce software products. If they have developed a new product or a new release of an existing product, they first test that product and at a certain time they decide to release the product. If a product has been released it is not unusual that customers find errors in the product, post-release errors. Those customers will then ask the company to fix those bugs. The 'Service'-department of the company will in most of the cases be the one to fix those bugs. Such a department has not a constant flow of work to do for specific products, because the amount of work depends on the arrival of errors. The planning of the capacity is therefore difficult and it would help if one could predict the intensity of the arrival of errors in time.

1.2 The intensity curve

In this paper not the whole problem will be discussed. The research namely has been focussed on estimating the intensity curve of the arrival of errors in time. This means that one only looks at the shape of the intensity function and tries to estimate it from the shapes of the intensity function of past releases. The idea is that if one finds a general intensity curve one then only has to predict the total number of errors of a new release and the time horizon in which those errors arrive, to get a prediction of the intensity function of that new release. This intensity function shows the predicted arrival of errors in time for the 'Service'-department.

1.3 The research

The research has been done in three steps which are the same as the next three chapters of this paper.

- First one has to choose a stochastic model that fits the problem.

- Then one has to estimate the parameters of the model. Here that means that the shape of the intensity has to be estimated.
- The fourth chapter includes a case study to look at how the model fits in practice.

In the last chapter one can find the conclusions of this research.

Chapter 2

The stochastic model

2.1 The data

To make a prediction of the process of arrivals of post-release errors in time, one can use a stochastic model. If one can find a model which fits the data of past releases well, one can make a prediction for a future release by estimating the parameters of this specified model. The data of the past releases can be represented like this:

$$\begin{pmatrix} t_1^{(1)} & , & n_1^{(1)} \\ t_2^{(1)} & , & n_2^{(1)} \\ & \vdots & \\ t_{m_1}^{(1)} & , & n_{m_1}^{(1)} \end{pmatrix} \quad \begin{pmatrix} t_1^{(2)} & , & n_1^{(2)} \\ t_2^{(2)} & , & n_2^{(2)} \\ & \vdots & \\ t_{m_2}^{(2)} & , & n_{m_2}^{(2)} \end{pmatrix} \quad \cdots \quad \begin{pmatrix} t_1^{(a)} & , & n_1^{(a)} \\ t_2^{(a)} & , & n_2^{(a)} \\ & \vdots & \\ t_{m_a}^{(a)} & , & n_{m_a}^{(a)} \end{pmatrix},$$

with:

- a the number of past releases,
- m_i the number of time units in the time horizon of release $i, a \geq i \geq 1$,
- $t_{m_i}^{(i)}$ $(=T^{(i)})$ the time horizon of release $i, a \geq i \geq 1$,
- $t_j^{(i)}$ observation time $j, m_i \geq j \geq 1$ of release $i, a \geq i \geq 1$,
- $n_j^{(i)}$ the number of errors in $\left(t_{j-1}^{(i)}, t_j^{(i)}\right]$ for $m_i \geq j \geq 1$,
- $t_0^{(i)} = 0$ and $n_0^{(i)} = 0$ for every release $i, a \geq i \geq 1$,
- $N^{(i)} = \sum_{j=1}^{m_i} n_j^{(i)}$ the total number of errors of release i.

2.2 Poisson process

The arrival of the post-release errors is typical for an (inhomogeneous) Poisson process. One can assume that the errors arrive independently from each other with a certain intensity which can fluctuate in time. If one counts the arrivals, for instance per week, one gets a vector of weekly counts $(n_1^{(i)}, n_2^{(i)}, \ldots, n_{m_i}^{(i)})$. That can be modelled as a discretized inhomogeneous

Poisson process with independent Poisson-distributed components $n_j^{(i)} \sim Poisson\left(\lambda_j^{(i)}\right)$ with $\lambda^{(i)}$ the intensity function (or intensity vector, because the intensity is constant per time unit) of release i and $\lambda_j^{(i)}$ the constant intensity in $\left(t_{j-1}^{(i)}, t_j^{(i)}\right]$. Here the time unit is set on a week because of the size of the time horizon (This is the time after the date of release in which all expected errors arrive). With a Poisson process however one can easily make a prediction with the time unit set on a day or a month.

To predict the intensity function $\lambda^{(r)}$ of a future release r one can start by analysing which components build the intensity function. Eventually one can then choose new model parameters which are easier to predict. This can be done in various ways. Here the intensity function is divided into three components which can be estimated independently. These three components are the standardized intensity curve λ_0, the total number of errors $N^{(r)}$ and the time horizon $T^{(r)}$ for the errors. With these three components one can calculate the expected $\lambda^{(r)}$ like this:

$$\lambda_j^{(r)} = \int_{t_{j-1}^{(r)}}^{t_j^{(r)}} \lambda^{(r)}(s)ds = N^{(r)} \int_{t_{j-1}^{(r)}/T^{(r)}}^{t_j^{(r)}/T^{(r)}} \lambda_0(s)ds. \tag{2.1}$$

This follows from

$$\lambda^{(r)}(t) = \frac{N^{(r)}\lambda_0(t/T^{(r)})}{T^{(r)}}, \tag{2.2}$$

and from

$$\int_{t_{j-1}^{(r)}}^{t_j^{(r)}} \lambda^{(r)}(t)dt = N^{(r)} \int_{t_{j-1}^{(r)}}^{t_j^{(r)}} \frac{\lambda_0(t/T^{(r)})}{T^{(r)}}dt = N^{(r)} \int_{t_{j-1}^{(r)}/T^{(r)}}^{t_j^{(r)}/T^{(r)}} \lambda_0(s)ds, \tag{2.3}$$

with substitution $s = t/T^{(r)}$ ($ds = \frac{1}{T^{(r)}}dt$) and time t, $0 \leq t \leq T^{(r)}$.

The second and third component ($N^{(r)}$ and $T^{(r)}$) are random variables that have to be predicted based on pre-release covariates. This will not be discussed in this paper. The first component (λ_0) has to be chosen from a certain set which has been composed due to homogeneity among certain releases. In this case we assume here there is only one standard curve and so it is not necessary to use pre-release covariates to choose a certain curve.

2.3 Multinominal distribution

If $N^{(r)}$ and $T^{(r)}$ are known, one can divide the time axis of length $T^{(r)}$ in m_r time intervals which here are the same as the time unit. Therefore m_r and $T^{(r)}$ are also the same. The $N^{(r)}$ errors then have to be distributed to those intervals following a certain pattern. This pattern is given by the probabilities $p_j^{(r)}$ which is the probability that an error arrives in the time interval $\left(t_{j-1}^{(r)}, t_j^{(r)}\right]$. This gives a new distribution for the vector $(n_1^{(r)}, n_2^{(r)}, \ldots, n_{m_r}^{(r)})$, namely $(n_1^{(r)}, n_2^{(r)}, \ldots, n_{m_r}^{(r)}) \sim Multinominal\left(N^{(r)}, p_1^{(r)}, p_2^{(r)}, \ldots, p_{T^{(r)}}^{(r)}\right)$. The probabilities $p_j^{(r)}$ play the same part as the $\lambda_j^{(r)}$'s, they give the shape of the intensity curve, with this difference:

$$\lambda_j^{(r)} = N^{(r)} * p_j^{(r)}, \qquad m_r \geq j \geq 1. \tag{2.4}$$

(2.1) can therefore be rewritten to:

$$p_j^{(r)} = \int_{t_{j-1}^{(r)}/T^{(r)}}^{t_j^{(r)}/T^{(r)}} \lambda_0(s)ds. \tag{2.5}$$

2.4 Summary

Summarizing, the stochastic model for predicting the arrival of post-release errors for a new release r is an inhomogeneous Poisson process with three components (λ_0, $N^{(r)}$, $T^{(r)}$) instead of the one ($\lambda(t)$) of a 'normal' (inhomogeneous) Poisson model. The parameter λ_0 'is a constant' that still has to be estimated. For a new release therefore only the components $N^{(r)}$ and $T^{(r)}$ have to be predicted. The same prediction has to be made if one uses a Multinominal distribution like here. The only difference is that one now has to estimate the probabilities p_j instead of λ_0. The relation, as already seen in (2.5), is

$$p_j = \int_{\tilde{t}_{j-1}}^{\tilde{t}_j} \lambda_0(s)ds. \tag{2.6}$$

Here j numbers a time interval $(\tilde{t}_{j-1}, \tilde{t}_j]$, with $1 \leq j \leq J$ and J the maximum number of intervals that can be created by all the past release (see also chapter 3). The λ_0 can also be computed from the probabilities p_j if λ_0 is constant on $(\tilde{t}_{j-1}, \tilde{t}_j]$, with $1 \leq j \leq J$.

Chapter 3

Estimation of the model parameters

The prediction of the variables $N^{(r)}$ and $T^{(r)}$ for a new release r, as already said before, will not be investigated here. They can be predicted based on pre-release covariates.

3.1 Standardization of past releases

Here the parameter λ_0, the standard curve, will be investigated, or better, the probabilities p_j. They represent the shape of the intensity function. This shape stays the same until new releases give information that could influence that shape. The standard curve is built up from all the standardized curves of the past releases.

To get the standardized curve from the intensity curve of a release one first has to make a choice about the way the process of arrivals is going to be standardized. The most logical way, based on the manner the intensity function has been divided, is to create a density function with of course a surface of one and also a time axis from zero to one. If one has such a standard curve it is easy to produce the actual intensity function after estimating $N^{(r)}$ and $T^{(r)}$. This has also been mentioned in chapter 2. To create this density function one has to make a decision about the way the total time horizon of a past release has to be converted to get the standardized curve. An elementary choice would be to make a general assumption about how long the number of errors arriving is significant that one wants to predict it. With that assumption one can define the time horizon $T^{(i)}$ of a past release i and therefore one can convert that part into the interval $(0, 1)$. If one knows the time horizon $T^{(i)}$ one also automatically knows the total number of errors $N^{(i)}$ of that release by counting the errors in $T^{(i)}$. Knowing both $T^{(i)}$ as well as $N^{(i)}$ means that one can convert the intensity curve of the release i to the standardized curve.

3.2 Estimation of the intensity curve

When all the past releases have been converted, one can put all the observation points $\tilde{t}_j^{(i)}$ (conversion from the real data $t_i^{(j)}$) together. One now has points in time $\tilde{t}_j^{(i)}$ in $[0, 1]$ of all the past releases. All the $\tilde{t}_j^{(i)}$'s together create time intervals (of different length) along the time axis. One now wants to calculate for every interval I_j between t_{j-1} and t_j, with $j = 1, \ldots, J$ and J the number of intervals, the number of errors in that interval without

looking from which release they are. This gives an estimation of the chance that an error of a release arrives in a certain time interval. This is the situation of the multinomial distribution mentioned in chapter 2, which means that if n errors arrive in total, the probability that x_1 errors arrive in interval I_1, x_2 errors in I_2 and x_J errors in I_J is $\frac{n!}{x_1!x_2!\cdots x_J!}p_1^{x_1}p_2^{x_2}\cdots p_J^{x_J}$, with $\sum_{j=1}^{J} x_j = n$ and $\sum_{j=1}^{J} p_j = 1$, with p_j the probability that an error will arrive in the interval I_j. If one now would know exactly how many errors of every release belong to the intervals I_j, one would have a good indication of the values of the p_j's, from now on referred to as p. These indications would be good initial values for the estimation of the real p by using the conditional maximum likelihood estimator (MLE). The conditional log-likelihood function looks as follows (leaving out the parts that are not important for maximizing):

$$l(\lambda) = \sum_{i=1}^{a} \log P_\lambda \left(\vec{N}^{(i)} = \vec{n}^{(i)} | T^{(i)}, N^{(i)} \right). \tag{3.1}$$

Because

$$P_\lambda(\vec{N}^{(i)} = \vec{n}^{(i)} | T^{(i)}, N^{(i)}) \cong \left(\sum_{j \in I_1^{(i)}} p_j \right)^{n_1^{(i)}} \left(\sum_{j \in I_2^{(i)}} p_j \right)^{n_2^{(i)}} \cdots \left(\sum_{j \in I_{m_i}^{(i)}} p_j \right)^{n_{m_i}^{(i)}}, \tag{3.2}$$

it follows that

$$l(\lambda) = \sum_{i=1}^{a} \sum_{k=1}^{M_i} n_k^{(i)} \log \left(\sum_{j \in I_k^{(i)}} p_j \right), \tag{3.3}$$

where $l(\lambda)$ can again also be written as $l(p')$, where λ and p' are maximized to λ_0 and p.

3.2.1 Complete data

The problem here is that one has not the complete data to make this estimation. The information one has of a release is the number of errors between two observation times $\tilde{t}_{j-1}^{(i)}$ and $\tilde{t}_j^{(i)}$. If an observation time $\tilde{t}_k^{(m)}$ of a release m ($m \neq i$) is between $\tilde{t}_{j-1}^{(i)}$ and $\tilde{t}_j^{(i)}$, this means that one only knows the number of errors for the intervals $\left(\tilde{t}_{j-1}^{(i)}, \tilde{t}_k^{(m)}\right)$ and $\left(\tilde{t}_k^{(m)}, \tilde{t}_j^{(i)}\right)$ together, but not for them individually. It follows from this lack of information (or incomplete data) that the parameter p can not be estimated directly, but has to be estimated iteratively. One can do this by first making an estimation of what the complete data looks like by choosing a starting value for p, and then estimate the complete log-likelihood. The result p' of the maximization will then be the starting value p for the new estimation of the maximum likelihood. This algorithm is known as the EM-algorithm, with the 'E' from 'Expectation' and the 'M' from 'Maximization'. The 'complete data' which is worked with here will be defined like this:

$$\begin{pmatrix} I_1 & , & f_1^{(1)} & , & f_1^{(2)} & , & \cdots & , & f_1^{(a)} \\ I_2 & , & f_2^{(1)} & , & f_2^{(2)} & , & \cdots & , & f_2^{(a)} \\ \vdots & & \vdots & & \vdots & & & & \vdots \\ I_J & , & f_J^{(1)} & , & f_J^{(2)} & , & \cdots & , & f_J^{(a)} \end{pmatrix},$$

with $f_j^{(i)}$ the number of errors of release i in the j-th interval I_j and

$$f_j = \sum_{i=1}^{a} f_j^{(i)} \tag{3.4}$$

the number of errors of all the past releases together in the j-th interval I_j.

For this complete data the log-likelihood function is:

$$\begin{aligned} l(p) &= \sum_{i=1}^{a}\sum_{k=1}^{J} \log P_p\left(N_k^{(i)} = f_k^{(i)}\right) = \sum_{i=1}^{a}\sum_{k=1}^{J} f_k^{(i)} \log p_k \\ &= \sum_{k=1}^{J}\left(\sum_{i=1}^{a} f_k^{(i)}\right)\log p_k = \sum_{k=1}^{J} f_k \log p_k. \end{aligned} \tag{3.5}$$

Maximizing this log-likelihood over all probability vectors gives:

$$p_k = \frac{f_k}{\sum_{j=1}^{J} f_j} \tag{3.6}$$

In the EM-algorithm this result is also important.

3.2.2 Estimation with the EM-algorithm

The expectation step of the EM-algorithm is to compute, based on a current iterate p, the conditional expectation of the complete log-likelihood as a function of p', given the (incomplete) data.

$$\begin{aligned} Q(p'|p) = E_p\left(l(p')|n_1^{(1)},\cdots,n_{m_a}^{(a)}\right) &= \sum_{k=1}^{J} E_p\left(f_k \log p_k' | n_1^{(1)},\cdots,n_{m_a}^{(a)}\right) \\ &= \sum_{k=1}^{J} E_p\left(f_k | n_1^{(1)},\cdots,n_{m_a}^{(a)}\right)\log p_k'. \end{aligned} \tag{3.7}$$

With

$$E_p\left(f_k | n_1^{(1)},\cdots,n_{m_a}^{(a)}\right) = \tilde{n}_k = \sum_{i=1}^{a} \tilde{n}_k^{(i)} \tag{3.8}$$

and

$$\tilde{n}_k^{(i)} = \frac{p_k}{\sum_{l\in I_j^{(i)}} p_l} n_j^{(i)}, \qquad k \in I_j^{(i)}, \tag{3.9}$$

for all releases $i, 1 \le i \le a$, (3.7) can be rewritten to

$$Q(p'|p) = \sum_{k=1}^{J} \tilde{n}_k \log p_k'. \tag{3.10}$$

The maximization step of the EM-algorithm is now to maximize (3.10) over all probability vectors p', with p fixed. This equation has the same structure as (3.3), the log-likelihood

function based on the complete data. If we make no assumption about the shape of the intensity function λ_0, the algorithm will converge to the optimal $\hat{p}'$:

$$\hat{p}'_k = \frac{\tilde{n}_k}{N}, \tag{3.11}$$

with N the total number of errors of all releases together. In steps the EM-algorithm is:

- **STEP 1:** Choose an initial value for p_k, say $p_k = \frac{1}{J}$, with $1 \leq k \leq J$. Name this $p_k^{(1)}$ as the starting value of the first iteration.
- **STEP 2:** Start iteration i ($i \geq 1$). Use (3.9) to calculate the $\tilde{n}_k^{(i)}$'s and with them the $\tilde{n}_k$'s using $p_k^{(i)}$ as p_k.
- **STEP 3:** Calculate $p_k^{(i+1)} = \frac{\tilde{n}_k}{N}, \quad \forall k.$
- **STEP 4:** Stop if $p_k^{(i+1)} \approx p_k^{(i)}$. The $p_k^{(i+1)}$'s give p', with $p' = \hat{p}'$ the optimum. Else, go to step 2 for iteration $i+1$ using $p_k^{(i+1)}$ as p_k.

With this algorithm one can thus calculate the optimal $\hat{p}'$ and, using (2.6), one has the standard shape of the intensity function, λ_0.

3.3 Summary

One now has a standard curve λ_0 of the intensity function of a release. If one can now predict $N^{(r)}$ and $T^{(r)}$ for a specific release, one has a prediction of the intensity of that release. This intensity predicts the flow of arriving errors of the new release in time. One can even simulate the inhomogeneous Poisson process with this intensity to get a better view of the possible fluctuation around the intensity curve. One can also give a prediction for a different time unit due to the properties of a Poisson process. This however will not be done here.

Chapter 4

Case study

In this chapter there will be a case study of the estimation of the intensity curve. There has been data used of twenty fictional releases where of course the assumption is made that the intensity curves all have about the same shape.. The data looks the same as in section 2.1, with $a = 20$. The total number of errors N and the time horizon T for every release i are:

Release	N	T
1	1248	77
2	4907	154
3	1117	189
4	2847	122
5	3886	128
6	3655	76
7	4534	178
8	2063	180
9	2457	117
10	6100	131
11	1906	189
12	2977	85
13	5646	144
14	6999	117
15	2023	97
16	5814	133
17	7521	80
18	1949	134
19	1692	94
20	1203	100

Table 4.1: *Total number of errors N and time horizon T of the twenty releases that were used in this case.*

The errors of a release are spread over the time horizon in a certain way, the intensity curve. This will never happen exactly according to the expected number of errors per week and therefore there is also a fluctuation around the intensity included in the data. This to better simulate a real situation. An example of this is given in figure 4.1.

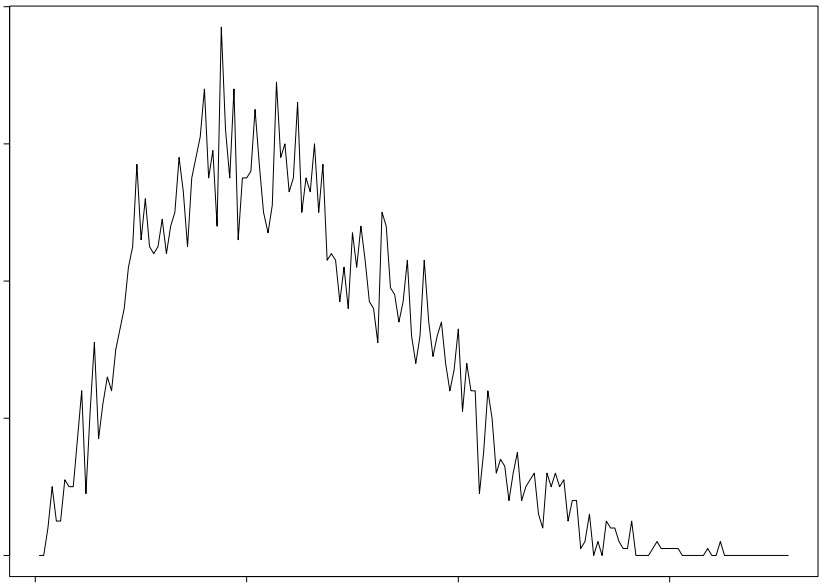


Figure 4.1: *Simulation of the number of errors per week of release 7.*

The releases are first standardized as explained in section 3.1. Then the intervals I_j, with $j = 1, \ldots, J$, are generated. J is here 1968 which is not the same as the sum of all the time horizons (2525), what one might expect, because there are releases with the same time horizon and also there are time horizons with some interval borders that are equal. The data and the intervals are now the input for the EM-algorithm, the steps of which are explained in section 3.2.2. There is no real 'stop condition' used, but there are a few runs made with always a (different) fixed number of iterations. There are two indicators used to see if enough iterations are made. First one can look at the maximum absolute deviation of all the probabilities p_j between two different number of iterations. Every p_j has then a deviation which is not bigger then that maximum during a certain number of iterations. This means that if that maximum is very small, the p_j's do not change that drastic anymore and one could say that one has found the optimum $p' = \hat{p}'$. The second indicator is the conditional log-likelihood (3.3). This is a number which increases with every extra iteration. It will however at one point not increase that quick anymore, because again the p_j's will not change that drastic anymore. Both these indicators can give someone a good idea about the required number of iterations to reach the optimum $p' = \hat{p}'$. Some results of the first indicator, the maximum absolute deviation, are given in table 4.2. When comparing the maximum absolute deviations, consider that the difference in iterations is not always the same. In figure 4.2 one can see the second indicator, the conditional log-likelihood. One can see clearly that the value of the log-likelihood increases with every iteration, but also that it increases up to a certain maximum boundary. In table 4.2 as well as in figure 4.2 one can see that after a certain amount of iterations the p_j's do not change that drastic anymore. Here it will not be discussed after ex-

iterations	1	2	3	4	10	100	1000	10000
1	x	0.00044	0.00073	0.00094	0.00147	0.00435	0.00502	0.00513
2	0.00044	x	0.00029	0.00050	0.00103	0.00427	0.00495	0.00505
3	0.00073	0.00029	x	0.00021	0.00074	0.00420	0.00487	0.00499
4	0.00094	0.00050	0.00021	x	0.00059	0.00412	0.00481	0.00492
10	0.00147	0.00103	0.00074	0.00059	x	0.00367	0.00448	0.00460
100	0.00435	0.00427	0.00420	0.00412	0.00367	x	0.00237	0.00291
1000	0.00502	0.00495	0.00487	0.00481	0.00448	0.00237	x	0.00077
10000	0.00513	0.00505	0.00499	0.00492	0.00460	0.00291	0.00077	x

Table 4.2: *Maximum absolute deviation of the p_j's between the two given number of iterations.*

actly how many iterations the optimum $p' = \hat{p}'$ has been reached, but one can see that 10000 iterations is more than enough. The results will therefore be presented with that number of iterations. First however a few pictures (figure 4.3 and 4.4) will be shown of the p_j's after a certain number of iterations, starting with 1 and ending with 10000 iterations. This is just to show what the changes are after more iterations. Again one can see that there are not many differences between 1000 and 10000 iterations anymore which means that enough iterations have been made.

At the end of the case the goal is to get an estimation of the intensity curve. This intensity curve was in the previous chapters represented by λ_0. This λ_0 has therefore now to be calculated from the p_j's with help from (2.6) and the assumption that λ_0 is constant on every interval $(\tilde{t}_{j-1}, \tilde{t}_j]$, with $1 \leq j \leq J$. In figure 4.5 this intensity curve λ_0 is shown. One can probably not see right away what the use is of this picture and what is says about the shape of the intensity curve. If one however takes one release, say for example release 7 again, and one now calculates the probabilities for the intervals of release 7 only, one gets a much clearer picture (see figure 4.6). In figure 4.7 one can see a picture which includes the number of errors per week of release 7 (see also figure 4.1) as well as the intensity curve of that release. This intensity curve is built up from λ_0 and the N and T from release 7. One can see that de the intensity fits very nicely.

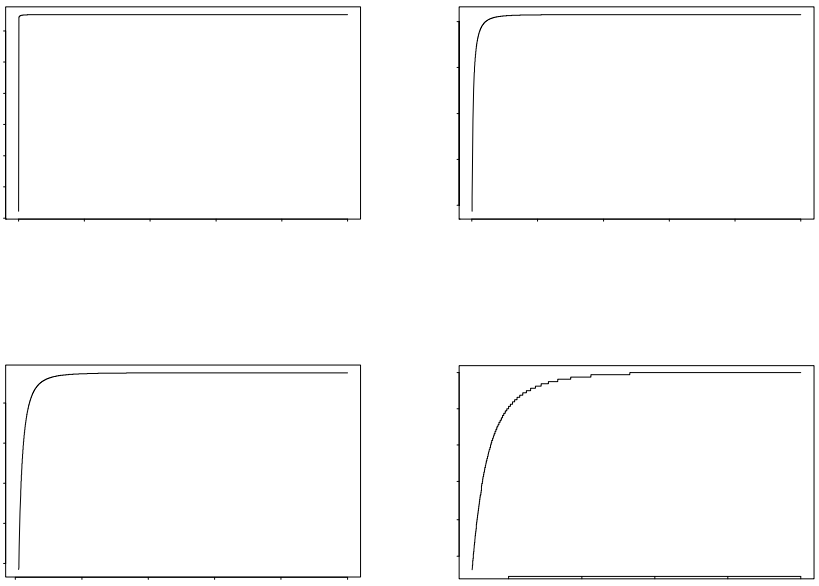


Figure 4.2: *The values of the conditional log-likelihood.*

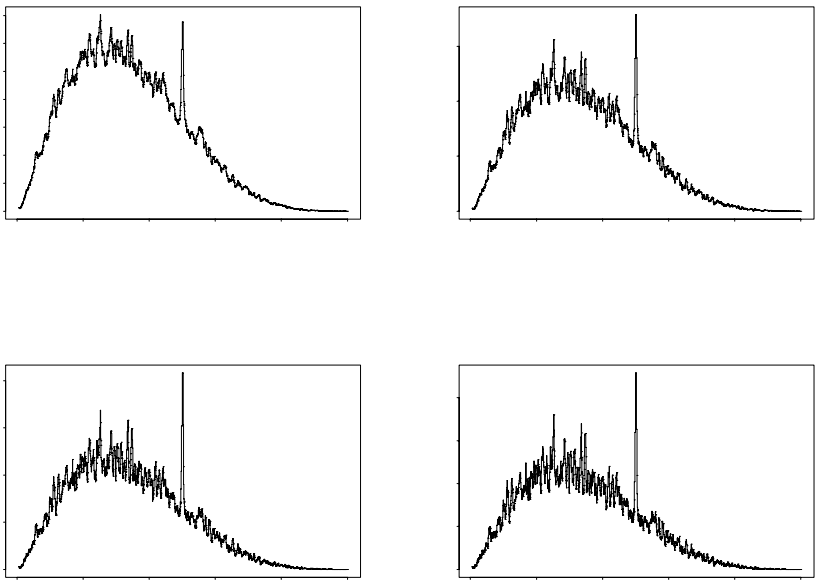


Figure 4.3: *The p_j's after 1,2,3 and 4 iterations.*

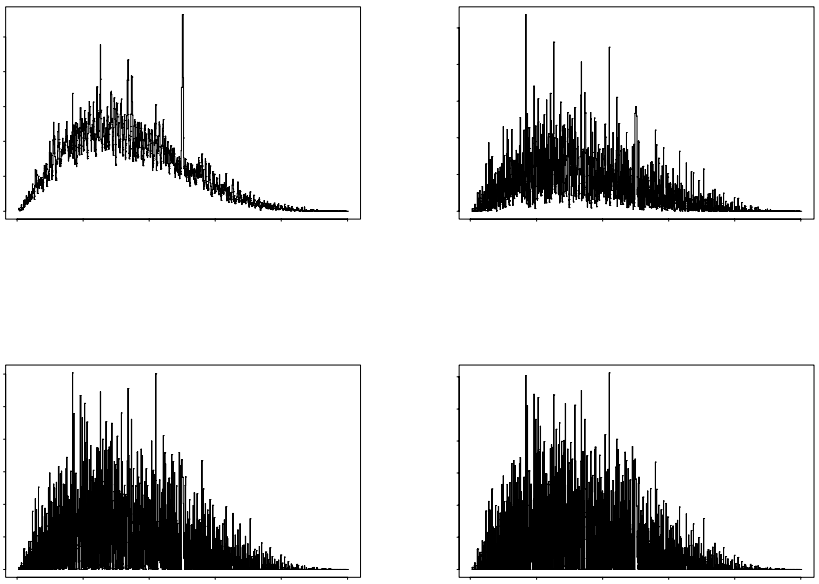


Figure 4.4: *The p_j's after 10,100,1000 and 10000 iterations.*

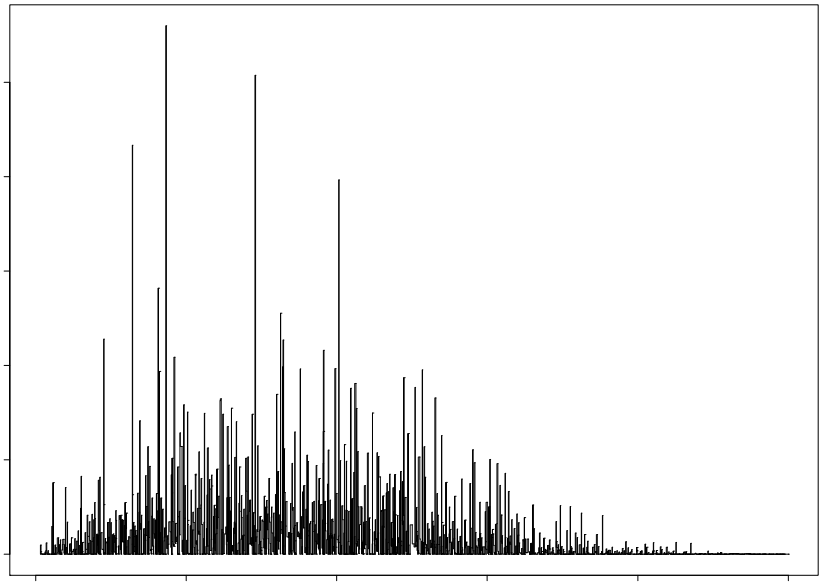


Figure 4.5: *The intensity curve.*

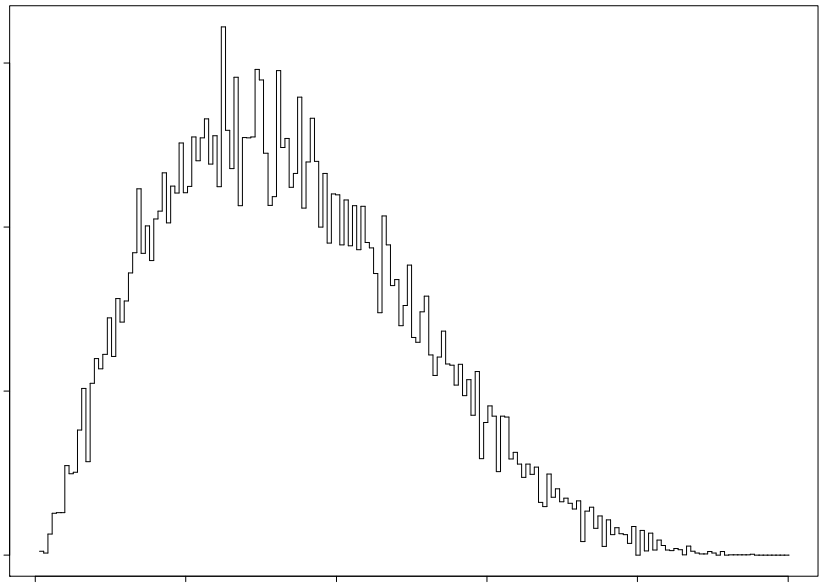


Figure 4.6: *The probabilities per interval of release 7.*

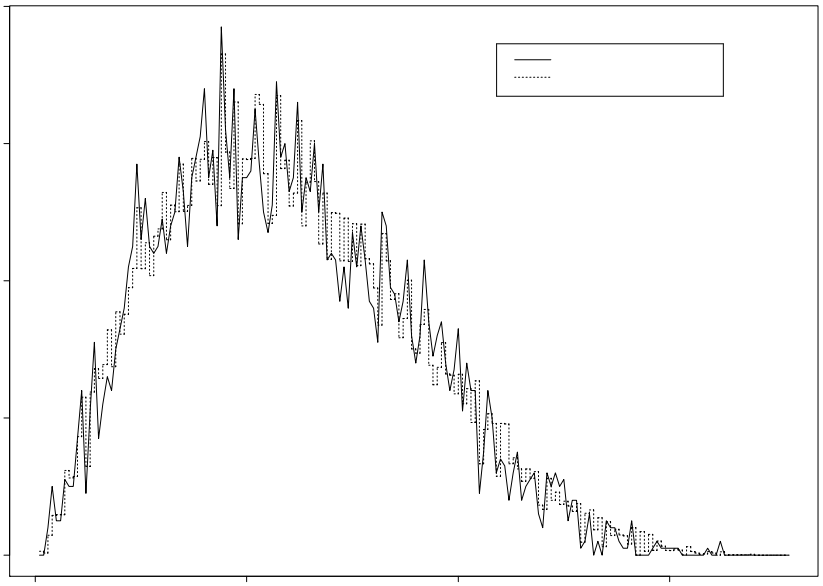


Figure 4.7: *The errors per week and the intensity of release 7.*

Chapter 5

Conclusion

At the end of this paper there are a few things one can conclude. There is also some future work one can do on this subject.

5.1 The conclusions

- If one wants to predict the intensity of some Poisson process, here the number of arriving errors per week of a release of a software product, one can do this by dividing the intensity into three components: The shape of the intensity curve λ_0, the total number of errors N and the time horizon T in which those errors arrive. If, like in this case, the intensity curve has always about the same shape, one can go even one step further and make an estimation of that shape. One now has a fixed intensity curve and for the prediction of the whole intensity one then only has to predict N and T, for instance based on pre-release covariates.
- The estimation of λ_0 can be done by estimating the p_j's of a Multinomial distribution (see (2.6) for the link between λ_0 and p_j), with p_j the probability of an error arriving in interval $(\tilde{t}_{j-1}, \tilde{t}_j]$, with $1 \leq j \leq J$.
- The p_j's can be estimated with the EM-algorithm (see section 3.2.2). One computes, based on a current iterate p, the conditional expectation of the complete log-likelihood as a function of p', given the (incomplete) data.

5.2 Future work

In the future the estimation of λ_0 can be extended with some restrictions on the shape. For instance, one can assume that the intensity curve has an unimodal shape, which is not unlikely in the case described in this paper. This means that the intensity curve first increases until a certain maximum has been reached and then decreases again $\lambda_1 \leq \lambda_2 \leq \cdots \geq \lambda_{J-1} \geq \lambda_J$, with $\lambda_j = \frac{p_j}{\tilde{t}_j - \tilde{t}_{j-1}}$. This restriction would mean that one has to do the M-step of the EM-algorithm (see section 3.2.2) under this assumption of unimodality.

Bibliography

[1] Dempster, A. P.; Laird, N. M.; Rubin, D. B. (1977) *Maximum Likelihood from incomplete data via the EM algorithm* J. R. Stat. Soc. Ser. B Stat. Methodol. 39, no. 1, 1-22

[2] Ramsay, J. O. (1998) *Estimating smooth monotone functions* J. R. Stat. Soc. Ser. B Stat. Methodol. 60, no. 2, 365-375

[3] Ramsay, J. O.; Silverman, Bernard W. (1997) *Functional data analysis*. Springer cop., New York

[4] Robertson, Tim; Wright, F. T.; Dykstra, Richard L. (1988) *Order restricted statistical inference*. Wiley cop., Chichester

[5] Ross, Sheldon M. (1997) *Introduction to probability models*. Academic Press, San Diego