M. Xie and C. Wohlin, "An Additive Reliability Model for the Analysis of Modular Software Failure Data", Proceedings IEEE 6th International Symposium on Software Reliability Engineering, pp. 188-194, Toulouse, France, 1995.

An Additive Reliability Model for the Analysis of Modular Software Failure Data

M. Xie
Dept of Industrial and Systems Engineering
National University of Singapore
Kent Ridge, Singapore 0511
Fax: +(65) 777 1434
E-mail: ISEXIEM@NUS.SG

C. Wohlin
Dept of Communication Systems
Lund Institute of Technology
Lund University, S-221 00, Lund, Sweden
Fax +(46) 46 145823
E-mail: CLAESW@TTS.LTH.SE

Abstract

Most software reliability models are applicable to a single piece of software. For more complex systems, Markov models have been studied assuming that complete reliability information at the module level is available. In this paper, we study an additive model which assumes that each subsystem or module undergoes independent testing and the reliability of the complete system has to be assessed. Subsystem reliabilities are estimated and the system reliability is assessed using the additive model. The approach is simple, but generic in a sense that existing models can be combined. An example is also presented based on the log-power model, a model with simple graphical interpretation, and it shows that the modular information should be used for system reliability assessment whenever such information is available.

1 Introduction

In software reliability modelling and analysis, it is commonly assumed that, during testing, the software fails when an instruction containing a software fault is executed. We experience an intensity of software failure which can be denoted by $\lambda(t)$. By correcting software fauls, this failure intensity can be reduced and the reliability increases. The expected number of failures in $[0,t)$, called the mean value function, is denoted as $\mu(t)$, and the relationship between $\lambda(t)$ and $\mu(t)$ is given as

$$\lambda(t) = \frac{d\mu(t)}{dt}; \quad t \geq 0. \tag{1}$$

Once $\mu(t)$ or $\lambda(t)$ is known, future failure behavior can be predicted and reliability related decisions can be made. Most studies in software reliability modeling and estimation are concerned with different forms for $\lambda(t)$ or $\mu(t)$. By making use of the testing data, unknown parameters are estimated, and hence, future failure behavior can be predicted, see e.g. the book by Musa et al. [10] for a general treatment of this topic.

Large software systems usually consist of a number of subsystems or modules which may be developed and tested in parallel. When two or more subsystems are put together, the reliability prediction is usually difficult. Because of the deadline to release the system, it may not be possible to collect enough data for making a system-level reliability assessment. If this is possible, then it is often necessary to determine the additional amount of testing so that the reliability requirement is met. The estimate may not be accurate if known information about the change of the system is not used, as most software reliability models assume that the testing is totally random and no new subsystems are added over the time.

Few studies have been carried out in solving this problem although there are many software reliability models proposed for assessing the reliability of a single system, see e.g. Xie [13] in which most of the existing models are summarized. In this paper we study an additive model for assessing the reliability of this type of integrated system. We assume that each

1071-9458/95 $4.00 ©1995 IEEE

individual subsystem is thoroughly tested and the aim is to assess the system reliability. Here a subsystem can be a module, a unit or a part of the system. However, it should be of a reasonable size and there are failure data at this level available.

Unlike some previous papers, such as Littlewood [8], Kubat [6], Laprie and Kanoun [7] and Dugan and Lyu [2], which assume that the subsystem reliabilities are known, subsystem reliabilities are estimated using subsystem failure data. Some possible problems associated with the proposed methods are also discussed. A simple numerical example is presented to illustrate the idea.

2 An additive model

We assume in this paper that the software system is composed of a number of subsystems, which may have been developed in parallel. They might be functioning or tested independently. This is a common approach in practice, especially for large software systems.

It should be pointed out here that in case of integration testing, failure times are still independent in random testing situation and failures can still be classified according to from which subsystem the original fault comes from. Hence, this assumption can easily be met as this type of information can be made available in practice.

As we consider any subsystem failure as a system failure, we can use a simple additive model. From a reliability point of view, the system can be considered as a series system with a number of subsystems without considering their functional structure. Note that a fault-tolerant system can also be considered as a series system from a reliability point view if any subsystem failure is treated as a system problem, although from a functionality point of view, the system is a parallel one.

We assume that the system failure intensity, $\lambda_s(t)$, is the sum of subsystem failure intensities, $\lambda_i(t)$, i=1,2,...n. The relationship between $\lambda_s(t)$ and $\lambda_i(t)$ is given by

$$\lambda_s(t) = \lambda_1(t) + \lambda_2(t) + \ldots + \lambda_n(t). \qquad (2)$$

The expected cumulative number of system failures at time t, $\mu_s(t)$, is given by

$$\mu_s(t) = \sum_{i=1}^{n} \mu_i(t) = \sum_{i=1}^{n} \int_0^t \lambda_i(s)\, ds$$

$$= \int_0^t \left(\sum_{i=1}^{n} \lambda_i(s) \right) ds. \qquad (3)$$

The mean value function is commonly used in software reliability studies as it directly measures the expected number of failures as a function of time although it is the intensity function that is of greater interest in reliability engineering. As given in (1), their relationship is however very simple. The failure intensity function is also the derivative of the mean value function. From (3), we see that the system level mean value function is also an additive function of subsystem level mean value functions.

Note that for the sake of convenience, we have assumed that t is set to zero at the same time for all subsystems. When subsystems are introduced at different time points, this assumption is not valid. Relaxation of this assumption will be discussed later on. The different starting points for different subsystem is also the situation for the numerical example presented in Section 5.

A comparison with some other approaches can be made here. First, as it is probably a common practice, we should start collecting new data if we know that the system is subjected to major change. For example, a new subsystem might have been added. In this case, we are not making use of complete knowledge, and it may take a long time to gather enough data for new estimation.

Another possibility is that we could just ignore the information that there are a number of subsystems and use the system level data. A single model can then be used in fitting the data and reliability can be predicted using this model. However, there is no existing model that can model the sudden change of the failure intensity, which is possible due to the new subsystem.

In case the subsystem level reliability growth rates differ considerably, it is pointed out in Xie and Goh [14] that inaccurate prediction will be made if a single model commonly used is applied. The two alternatives discussed previously all suffer from this

problem in case of different reliability growth rates for subsystems.

On the other hand, it is also a waste of information if the available knowledge is not made use of. In this case, the time of the introduction of new subsystem, and the knowledge of where the original fault is from are important pieces of information which are easily obtained. Hence, the additive model is more preferable and it is the focus in this paper.

3 The use of existing models

The simple additive model proposed here can be seen as a superposition of nonhomogeneous Poisson process (NHPP) models. NHPP models are commonly used in software reliability studies, see e.g., Yamada and Osaki [16], Goel [3] and Xie [13]. It can be noted that the mean value function and the failure intensity function are uniquely determined when any one of them is known. It is well-known that if we have a number of subsystem failure processes and all can be modelled by a NHPP, then the system failure process defined as the sum of all failure processes can also be described as a NHPP with the mean value function as the sum of the underlying mean value functions.

For the interpretation of results, we assume that the underlying subsystem failure processes can be described by NHPP with mean value function $\mu_i(t)$, i=1,2,...n. Hence the additive model is also a NHPP model, with mean value function given in (3) and the corresponding failure intensity function is given in (2). Note that this is different than the weighted model presented in Lyu and Nikora [9]. All properties of NHPP are then valid for our additive model and the system failure process that is described by the additive model.

The additive model is a generic model. Any existing software reliability model can be used as the underlying software reliability growth model for each subsystem. In Xie and Goh [14], some examples on hardware reliability assessment are presented using the Duane reliability growth model as the underlying subsystem reliability growth model. The Duane model, although widely referred to, is not suitable for software failure data.

A number of studies have been carried out to select a suitable model, see e.g. Keiller and Miller [5] and Lyu and Nikora [9]. Although this is not the topic of this paper, the models have to be selected based on simplicity, accuracy and the underlying assumptions associated with each model. This is a difficult issue and can only be tested when data are available. However, when past data can be obtained, the accuracy of the model used for similar products gives a good guideline for model selection.

Although the application of the additive model is straightforward and any existing software reliability growth model can be used as the underlying reliability growth model, there are some potential problems. Some of them are easy to solve, while others are more difficult to deal with. We have to note that the overall model cannot be expected to perform better than the worst of the underlying models. This implies that different models can be used for different subsystems in the system. Hence, the selection of underlying models is still very important, although we will not elaborate it further.

The most immediate problem when the additive model is used in software reliability analysis is that the starting time may not be the same for all subsystems. It is common that some modules or subsystems may be developed in order to complement earlier parts of the system with separate and additional functions, as it may be demanded by new customers. If this is the case, existing models have to be modified to indicate the difference between the time when testing started or the time has to be adjusted.

As the problem involves a number of models, the complexity increases. Hence, it is advisable to use simple models or to rely on software tools for reliability assessment. To simplify interpretation of the results, it is also better to use similar models for subsystem failures. Since subsystems within the same company, are usually developed and tested in a similar manner, the use of similar models is justified. In this paper, the graphical approach is emphasized, although there are other software reliability assessment packages that can be used.

Models similar to the additive model have been used indirectly in reliability growth studies although no formal description is available, see, e.g., Hansen and Thyregod [4] and Ray et al. [12]. Yamada et al. [17] have earlier studied a related model called two-type of error model related to software reliability growth analysis. However, in this paper we present a study for solving some practical problems when the additive model is interpreted and applied directly.

4 A graphical approach

To solve the problem of increased complexity due to the use of the additive model, we should use simple

graphical models, such as the Duane model, the log-power model or other models for which simple graphical estimation procedures exist, see Xie and Zhao [15]. Models with graphical interpretations have been called First-Model-Validation-Then-Parameter-Estimation models since they can be validated before the parameters are estimated. In practice, the parameter estimation and model validation, which is usually done by checking the fit after the parameters are estimated, are difficult tasks for other models.

The Duane model originally proposed in Duane [1] has shown to be very useful in system reliability growth studies, although its application to software testing is not encouraging. However, most of the existing software reliability models do not possess the graphical property similar to that of the Duane model, and estimation has to be carried out using numerical procedures, and not to mentioned that the model validation is another more difficult issue.

Some modifications retaining the graphical property of the Duane model are presented in Xie and Zhao [15]. One of the models which is promising is the log-power model, and this model will be employed here.

The log-power model with parameters a and b has the mean value function given as

$$\mu(t) = a\ln^b(1+t), \quad t\geq 0. \tag{4}$$

It can be noted that a linear relationship can be obtained by taking the logarithm on both sides and we have

$$\ln\mu(t) = \ln a + b\ln\ln(1+t), \quad t\geq 0. \tag{5}$$

Hence, by plotting the empirical cumulative number of failures versus time using log-double-log scaled axis, the plot should approximately be on a straight line with a slope equal to b. This is provided that the log-power model is valid, and otherwise, the model should be rejected without a need for further investigation. Little time is wasted in the case of not accepting the model.

5 A numerical example

In this section we present a study using a set of data from a large communication software project to illustrate the use of the additive model. It simply consists of two subsystems, which are tested independently. The failure data for each subsystem and that for the system is given in Table 1 with the plot of empirical system failures displayed in Figure 1.

Month	Sub-system 1	Sub-system 2	System
1	2	NA	2
2	11	NA	11
3	18	NA	18
4	10	NA	10
5	12	NA	12
6	4	NA	4
7	28	NA	28
8	6	NA	6
9	7	NA	7
10	6	NA	6
11	17	NA	17
12	31	NA	31
13	8	NA	8
14	7	NA	7
15	10	NA	10
16	2	NA	2
17	2	NA	2
18	0	NA	0
19	3	NA	3
20	2	NA	2
21	1	NA	1
22	1	NA	1
23	1	3	4
24	0	3	3
25	1	38	39
26	1	19	20
27	0	12	12
28	0	13	13
29	0	26	26
30	1	32	33
31	0	8	8
32	0	8	8
33	0	11	11
34	0	14	14
35	1	7	8
36	0	7	7
37	1	7	8
38	0	0	0
39	0	2	2
40	0	3	3
41	0	2	2
42	0	5	5
43	1	2	3
44	1	3	4
45	0	4	4
46	0	1	1
47	1	2	3
48	0	1	1
49	0	0	0
50	1	1	2

Table 1. A set of system failure data with information at subsystem level.

It is clear from the plot of system failures that no single model in the existing literature can fit the data very well because of the obvious change in the failure intensity around t=26. There is no model that can incorporate this sudden change of system failure behavior. Note that t=26 is the time when a large number of failures occurs in Subsystem 2, but it is not the same as the time for the introduction of the second subsystem.

Such a change is common and it is usually caused by the introduction of a new subsystem into the system. If the whole system is regarded as new, then a large amount of additional testing has to be done to make use of the model again. The early part of the information is usually not made use of and it is a waste of information which can be costly.

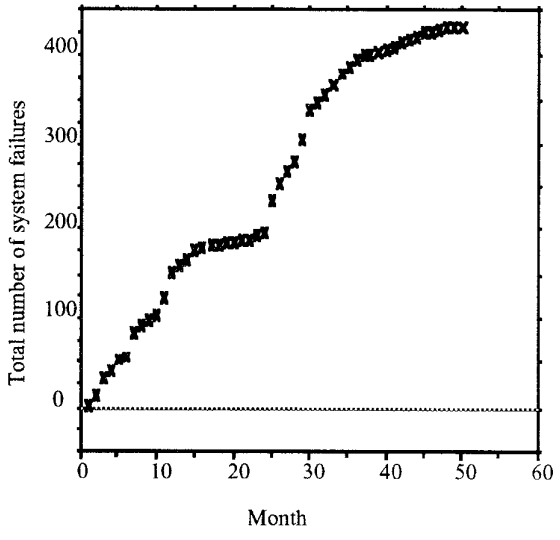

Figure 1. The plot of empirical failure data for the whole system.

From the subsystem failure data, we can actually make use of all information by analyzing each subsystem separately. Note that even in the case when subsystem failure information is not presented, it may be worthwhile to classify the system failures into subsystem failures. The plot for the two subsystems based on log-power models are displayed in Figure 2 and 3, respectively.

The regression is done in a standard way using any spreadsheet software. . From the regression equation, the estimates can read according to (5) and an estimated mean value function can be obtained.

The estimated mean value function for Subsystem 1 failures is

$$\mu_1(t) = e^{2.344} \ln^{2.56}(1+t), \quad t \geq 0. \tag{6}$$

and the corresponding mean value function for Subsystem 2 is

$$\mu_2(t) = e^{3.314} \ln^{1.895}(1+t), \quad t \geq 23. \tag{7}$$

Based on this information and by noting that the second subsystem is incorporated at t=23, we estimate the overall system failure intensity as

$$\mu(t) = \begin{cases} e^{2.3} \ln^{2.6}(1+t), & 23>t>0; \\ e^{2.3} \ln^{2.6}(1+t)+e^{3.3} \ln^{1.9}(1+t), & t \geq 23. \end{cases} \tag{8}$$

and it is this equation that should be used for the prediction of future behavior.

Some remarks are in order here. As we have highlighted an approach to estimate system failures based on subsystem failure data and the emphasis is on the simplicity of the approach, we are not stating that the log-power model is the best model for this set of data. In fact, from the raw data, we can see that there is a sudden drop of the failure intensity for the first subsystem at around t=16. This can be caused either by relocating the testing effort or a change of testing strategy. Execution time, which is probably better in this case, could be tried if it was available.

For both data sets, it seems that the failure intensity is increasing at the beginning as the number of failures in the first two intervals are significantly smaller than the third. This phenomenon is usually caused by the learning effect as the testers may not be familiar with the system.

For this type of data, we would suggest the use of the s-shaped model as this type of model is suitable in dealing with the increasing failure intensity at the beginning, see e.g., Ohba [11] and Yamada and Osaki [16]. Despite these minor problems, the approach of using subsystem information in system reliability estimation is still the same with the use of additive models and we would suggest them to be incorporated in this type of analysis.

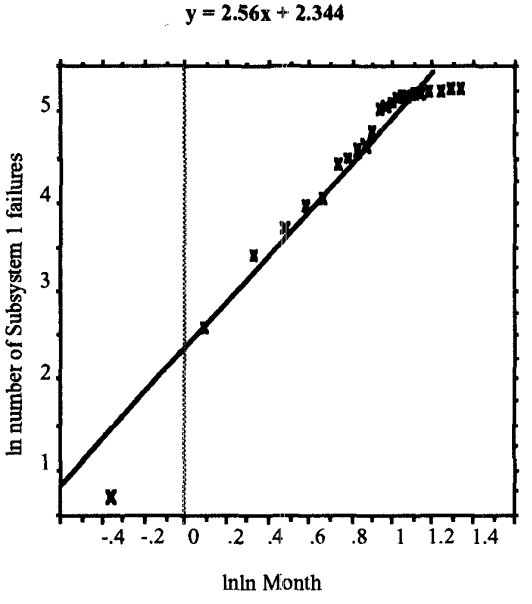

Figure 2. The log-power model applied to Subsystem 1 failure data.

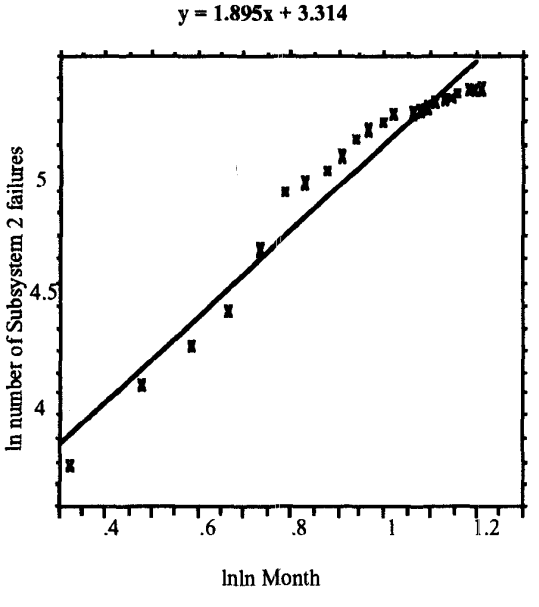

Figure 3. The log-power model applied to Subsystem 2 failure data.

6 Discussion

In this paper we have presented a study using a simple additive model in studying software reliability when the system is modular and developed and tested independently. The approach, although very simple, is practical for more accurate decision-making. It incorporates some easily available system development information in reliability estimation and it provides a better statistical description of system failures.

There are some potential problems with this approach, and some of them are also areas for further study. First, we assume that the subsystems are developed independently without knowledge of each other. This may not be true in practice, although it is likely that some knowledge would have been gained when the first subsystem is developed, so that the subsequent subsystems can be better initially. However, without additional information, our model is a step forward in making reliability prediction by making use of some easily available historical information of system development.

In our example, the second subsystem seems to be worse, as it can be seen from Table 1 that there are more failures with the second subsystem. One possibility is that it is likely that the second subsystem is developed in a hurry to meet the deadline. This will certainly affect the overall reliability. Hence it is inconclusive and further studies have to be presented.

Second, we assume that there is no problem with the interface of two subsystems. In integration testing, this is usually not the case. However, the results and conclusions are still valid if the interface errors are excluded, as for that particular type of faults, the testing data can and should be gathered and analyzed separately if they are many in number.

Acknowledgment: The authors would like to thank a number of referees who have provided constructive criticism and suggestions that have helped us to remove a number of errors and make the paper more reader-friendly.

References:

[1] J.T. Duane, "Learning curve approach to reliability monitoring," *IEEE Trans. on Aerospace*, AS-2, 563-566, 1964.