Compound and Non Homogeneous Poisson Software Reliability Models

Néstor R. Barraza*

Facultad de Ingeniería. Universidad de Buenos Aires
Paseo Colón 850. Buenos Aires
nbarraza@fibertel.com.ar

Abstract. The efficiency of two Software Reliability growth models are analyzed. The most popular based on a non homogeneous Poisson process and the less known based on a compound Poisson process. Several experimental data are used in order to analyze the goodness of fit of both models. The importance of the estimation method for the parameters involved is also analyzed.

Key words: Software Reliability, Poisson, maximum likelihood, mode estimator, Plackett, Poisson truncated at zero

1 Introduction

Software Reliability has to do with the probability of finding failures in the operation of a given software during a period of time. Failures are a consequence of faults introduced in the code. The interest in Software Reliability models has increased since the software component became an important issue in many Engineering projects. An important class of these models are the so called growth models. They use reports of past failures in order to fit the failure curve, or to predict either the failure rate or the remaining failures in a given time interval. Some Software Reliability Growth Models based on non homogeneous Poisson Processes have been proposed years ago. They are widely used and are still matter of study, [1], [8], [12], [18]. Other models take into account failure correlation or clusters, [7],[20]. A Compound Poisson process has also been introduced in order to model clusters of failures [16]. In the literature of SRGM, only stochastic processes like NHPP, S-shaped, Exponential, etc., which model the whole behavior of the number of failures as a function of time are considered. One of the aim of this work, is to consider the Compound Poisson process model as a SRGM and to analyze cases where it can be preferred over the NHPP. Non homogeneous and Compound Poisson processes try to represent the main characteristic in Software failures, it is the testing process and the failure removal. This characteristic results in increasing the mean time between failures and lowering the failure rate as testing time goes on. Both approaches intend to model this

---

behavior by introducing a non homogeneity in the first case, and a compounding distribution in the last case. In Compound Poisson processes, non homogeneity is replaced by grouping failures in a given time interval, and viewing that groups are less populated as time goes on. The difference in these approaches makes difficult to compare both models, since the Non homogeneous Poisson processes model the behavior of software failures as a function of time, and the Compound Poisson process try to predict the remaining number of failures or the failure rate at a given time instead of fitting the failures curve. However, taking into account that both models give a probability of the number of failures in a given time interval, and they can also be used to predict the remaining number of failures at the end of a given phase, some sort of comparison can be made. In this work, a comparison of these models is made using several failures reports. The estimation of the parameters involved in both models is another important issue. It is well known that the Maximum Likelihood Estimator is not always attained in Non Homogeneous Poisson process models, see [8], [10], [22], and generally, equations are little complicated to solve. Since there is a single parameter to estimate in a unique distribution which does not depend on time, estimation process is easier in Compound Poisson models. Some estimation methods have been proposed in the literature. However, an important point must be remarked here. Since the goodness of fit of a Software Reliability model is evaluated by the number of remaining failures predicted, and it is obtained as the expected number of failures from the distribution function, then, any unbiased estimator for the mean, gives the same prediction as the simple Poisson probability function, as it will be shown. Consequently, biased estimators showing a diminution in the size of groups of failures give better results. Performance of Compound Poisson and NHPP models has been compared in [17], [2], [3]. In this work, the results previously shown in [2] and [3] are extended, a more detailed study of the mode estimator is presented and the median estimator is also introduced for comparison.

2   Software Reliability Growth models

Growth models are an important class of Software Reliability models. They were proposed in order to represent the process of finding and removing faults. These models were introduced in order to fit the failures production curve, on the other hand, past reports of failures production is used to predict the remaining number of failures or to find the failure rate. In a not intended to be exhaustive list, models usually mentioned in the literature of SRGM are: Goel Okumoto [5], S-shaped [23], Log power [24], Musa-Okumoto [13], Log-Logistic [6], Jelinski-Moranda [9], Littlewood-Verrall [11].

SRGM models can be applied during several phases of software development. In an advanced stage of software development, the predicted failure rate will be the starting rate at the testing phase. The prediction of the rate at the end of the testing phase will also be quite important since it is the beginning of the

maturity phase, previous to deployment. The rate predicted at the end of the maturity phase will be that of the product at delivering.

In summary, Software Reliability Growth models (SRGM), can be applied in any stage of software development, all of them are very important. As the product becomes mature, the failure rate gets lower. This behavior is what the SRGM try to predict. What we want to obtain is either a prediction of the number of failures in a given future time interval, or the failure rate at a given time, for example at the moment of delivery, when the software will be in production. This information is useful to know whether the product is ready for release or not, how much testing will be needed to add, etc.

In SRGM, the probability of the number of failures in a given time interval is given by a stochastic process $P(N(t) = n)$ and the predicted number of failures is obtained as the mean value $E[N(t)]$.

Typical curves of cumulative software failures as a function of time are shown in (Fig. 1) and (Fig. 2).
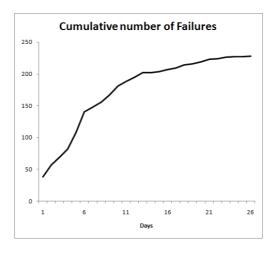

Fig. 1. Software failures as a function of time

Inflexion points in some of these curves usually appear because of the introduction of new code, either due to a necessity in the project or to fix an important bug. All of them show a decreasing failure rate and finally, the failure rate becomes almost constant.

2.1   Non-homogeneous Poisson process models

Non homogeneous Poisson processes are an important class of SRGM, they have been proposed as Software Reliability models years ago, most of them have been compiled in a well known book, [12]. In these models, the Poisson parameter
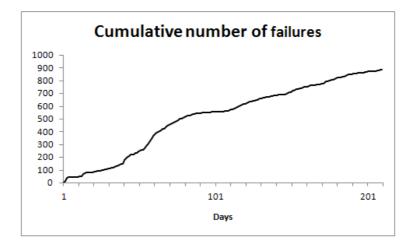
Fig. 2. Software failures as a function of time

$\lambda(t)$ is a given function of time, instead of being constant as in a normal Poisson process.

The probability of the number of failures in a given time interval t is given by:

$$P(N(t) = n) = \frac{\lambda(t)^n}{n!} \exp(-\lambda(t)) \tag{1}$$

There are several models proposing different functions $\lambda(t)$. Two of the main non-homogeneous Poisson process models are the Goel-Okumoto [5], and the Musa-Okumoto model [13], giving the following functions for $\lambda(t)$:

$$\lambda(t) = a(1 - \exp(-bt)) \quad \text{Goel} - \text{Okumoto} \tag{2}$$

$$\lambda(t) = \frac{1}{\theta} \ln(\lambda_0 \theta t + 1) \quad \text{Musa} - \text{Okumoto} \tag{3}$$

Since $\lambda(t)$ is the mean value, the predicted number of failures is also given by equations (2) and (3).

2.2   Compound Poisson process model

Another Software Reliability model based on a Poisson process was proposed years ago [16]. This model considers that failures are produced in clusters, thus, clusters arrive following a Poisson process and a compounding distribution models the clusters size. In the original paper, it was considered that the cluster size follows a geometric distribution, therefore, the number of failures follows a Poisson Compound by a Geometric. Since this model is not intended to adjust the

failures production curve, to the best of our knowledge, it has never been mentioned in the literature of SRGM.

A Compound Poisson process probability function is given by:

$$P(N(t) = n) = \sum_{k=1}^{m} \frac{(\lambda \, t)^k}{k!} \exp(-\lambda t) \, f^{*k}(X_1 + X_2 + \cdots + X_k = n) \qquad (4)$$

where $f^{*k}(X_1 + X_2 + \cdots + X_k = n(t))$ is the probability that the sum of k independent identically distributed random variables X_i following a distribution function $f(X_i)$ gives $n(t)$. As the distribution $f(X)$ modeling the failures cluster size, the geometric distribution was proposed in [16], and the Poisson Truncated at Zero was proposed in [2].

The mean value function for this model is given by:

$$E[N(t)] = \lambda \, t \, E[X] \qquad (5)$$

It is seen from (5) that the Poisson process mean value is adjusted by the expectation of the failures cluster size. Looking at equations (2), (3) and (5), we can see that the predicted failure rate appears as a fundamental difference between Compound and Non homogeneous Poisson process models. It is constant for the first one and dependent on time for the second.

We can see from (5) that any unbiased estimator for the mean value can not take advantage of using a Poisson model more complex than the simple one. In order to see this, expressions for the mean value unbiased estimators are shown:

$$\hat{\lambda} = \frac{m}{t_{past}} \qquad \hat{E}[X] = \frac{n}{m}$$

where m is the number of arrivals by the time t_{past}, and n is the total number of reported failures.

$$E[N(t)] = \hat{\lambda} \, t \, \hat{E}[X]$$
$$= \frac{n}{t_{past}} \, t \qquad (6)$$

As it is seen in Eq. (6), the estimator of $E[N(t)]$ is the same as using the simple Poisson process.

2.3 Compounding distribution

The geometric distribution was the first proposal as the compounding distribution in [16], it is given by:

$$P(X) = r^{(X-1)}(1 - r) \qquad (7)$$

That approach can be partially supported by the fact that the Poisson Compounding by a geometric is the limit of a two states Markov chain. Following the

known characteristic of adjusting failures production, the Poisson Truncated at Zero distribution was proposed as the compounding in [2], see also [3]. Truncation at zero is justified since we are not counting zero faults arrivals. The Poisson Truncated at zero is given by the following formula:

$$P(X) = \frac{a^X}{X!} \frac{1}{(\exp(a) - 1)} \tag{8}$$

2.4 Parameter Estimation for the Compounding Distribution

The Poisson Truncated at Zero distribution has been extensively studied and the estimation of its parameter has been matter of study. The Plackett estimator and the Tate & Goen unbiased minimum variance estimator are shown as an example [19], [14].

$$\hat{a} = \frac{1}{m} \sum_{\substack{i=1 \\ x_i > 1}}^{m} x_i \quad \text{Plackett}$$

$$\hat{a} = \frac{n \left\{ {n-1 \atop m} \right\}}{\left\{ {n \atop m} \right\}} \quad \text{Tate \& Goen}$$

where $\left\{ {n \atop m} \right\}$ is the Stirling number of the second kind

As it was quoted in the introduction, the failures clusters size gets lower as testing time progresses. This behavior must be taken into account by the estimator of the parameters of the Compounding distribution. Then, an estimator which follows this behavior will be better than others which take into account big size clusters of low frequency of occurrence. Consequently, biased estimators should be considered. Following this reasoning, the mode estimator was proposed in [2]. The mode estimator is determined by the cluster size occurring most of the time, not matter the size of the others. Since it is expected that the cluster size of maximum occurrence has a low value, then, the corresponding distribution will predict better the future cluster sizes. This characteristic of the mode estimator is shown in (Fig. 3). The mode estimator is determined by the cluster size occurring most of the time (2 in this example), it will adjust better the future failures clusters size provided they will become smaller as testing time progresses. Conversely, the ML estimator takes into account cluster sizes other than the maximum since it is expecting the same proportion, then its prediction will be worst.

Being k_i the number of occurrences of the cluster size X_i, the mode estimator can be expressed analytically by the following equation:

$$i^* = Arg \max_i (k_i)$$

$$\hat{a} = x_{i^*}$$

Maximum Likelihood estimate for the geometric and the Poisson Truncated at Zero compounding distributions are also unbiased estimators for the mean,
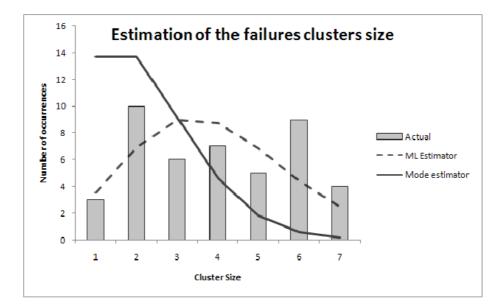
Fig. 3. Mode and ML estimators

having consequently the problem quoted in section 2.2. An alternative estimation method has been addressed in [17] for the Poisson Geometric model, though without considerable improvements over ML.

3   Software Reliability Models Comparison

Analysis and comparisons of SRGM has been widely published, see for example [1], [4], [15]. Besides the number of predicted failures, other methods have been used, for example u-plots and Prequential Likelihood. In this work, we use just the predicted number of failures, since the mean time between failures concept is not applicable to Compound Poisson Models.

Several software failures data from the bibliography and others available through the web, for example from the DACS (DACS - Data & Analysis Center for Software, [21]), can be used in order to test software reliability models.

Software failure predictions are compared next with actual data for two data sets. Comparisons are made for the Goel-Okumoto non homogeneous Poisson process model (GO) and for the Poisson compound by a Poisson Truncated at Zero using the mode, the median and the Plackett estimators.

In order to estimate the Goel-Okumoto model parameters, the maximum likelihood method is applied. Non existence of the estimates in early stages besides other problems in the maximum likelihood method has been reported in the bibliography, (see [10], [8], [22]). Despite the last remark, we prefer the ML estimator because of its better adjustment when it is attainable.

The first group of data shown in (Fig. 4) was obtained from a client-server project written in C under Linux, it is composed of about 250000 lines of C code. Let's call this group of data "Data 1". Failures data were reported during the test and development phase. The graph shows better adjustment for the non homogeneous process model, however, it has an ML estimator convergence problem before day 120th. The median estimator adjusts better than the mode and generally, its performance coincides with that of the Plackett estimator. Cumulative number of failures for these data is shown in (Fig. 2).
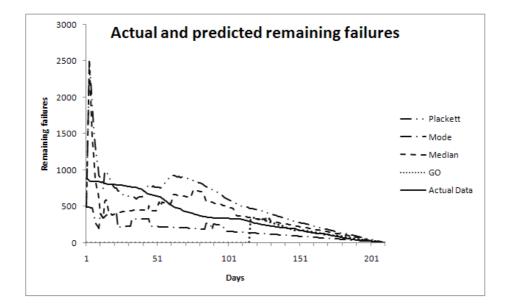

Fig. 4. Software Reliability models comparison. Data 1.

Another comparison is shown in (Fig. 5), data correspond to the system code 5 in [21]. Let's call these data "Data 5". This project has 2.445.000 object lines code and failures data were compiled during the test phase. Predictions for all the models are similar for these data. However, the Compound Poisson model using the mode estimator adjusts better from the beginning of test and also follows the jumps in the number of failures.

In order to understand what kind of data are more suitable for the compound Poisson process model, the evolution in clusters size are shown below.

As it was shown, the proposed model adjusts better for Data "Data 5". We can see from (Fig. 6) that the cluster size of maximum occurrence moves from 5, then 4, and finally 2 for the 10th, 40th and 60th day of test. This "movement" is better followed by the mode estimator. Conversely, as it is seen from (Fig. 7), the cluster size of maximum occurrence is always 1 for almost the whole test
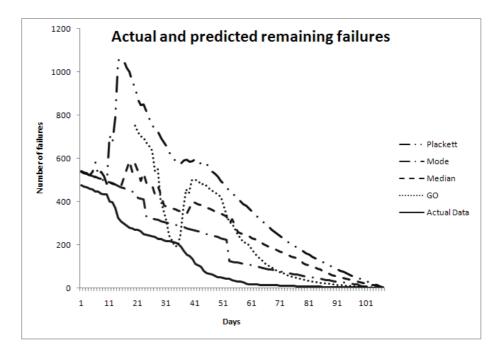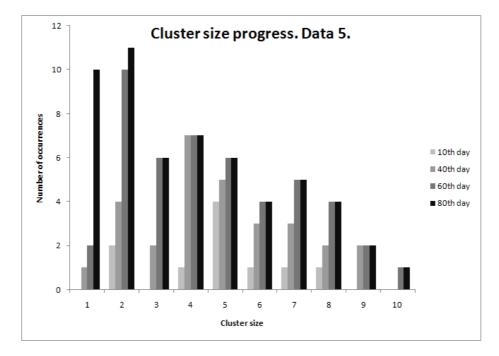
Fig. 5. Software Reliability models comparison. Data 5.
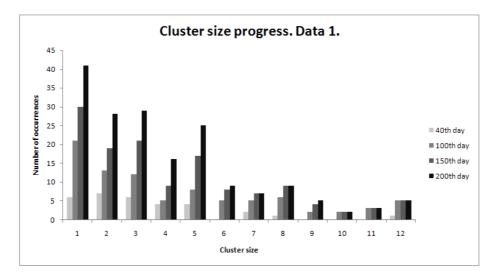

Fig. 6. Cluster size progress.

Fig. 7. Cluster size progress.

time for "Data 1". Then, the Compound Poisson process model does not have the better adjustment for these data.

4   Conclusions

Two Poisson based Software Reliability Growth Models were compared using failure reports from two software projects. Several estimators were analyzed in order to show differences and goodness of fit. The importance of biased estimators was remarked provided failure cluster size gets lower as time goes on. The mode and median estimators were suggested this way. We can conclude that both models adjust well the actual failures data, however, some advantages and disadvantages can be pointed out. Nonhomogenous Poisson processes are useful to represent the whole behavior of the cumulative number of failures as a function of time. On the other hand, as it was quoted, the ML estimate can not always be obtained, and generally, according to our experience and what was reported in the bibliography, ML estimate is not attainable before forty to sixty percent of testing time. Compound Poisson process are easier to implement and its parameters can be rapidly obtained through simple equations from the beginning of test. This model is suitable for applying several biased estimators. This characteristic allows it to adapt faster to changes in the project. The main disadvantage of this model is the constant failure rate prediction, then, it can not be used for a long time prediction, the failure rate needs to be updated time to time. As it was also shown, the Compound Poisson process model seems to adjust better for data whose cluster size of maximum occurrence moves from high to low values along the testing time.

References

1. Almering V., von Genuchten M., Cloudt G., Sonnemans P. J. M.: Using Software Reliability Growth Models in Practice, IEEE Software, 24, 82-88 (2007)
2. Barraza N. R., Applications and Analysis of Cooperative Phenomena using Statistical Mechanics, Contagion and Chains of Rare Events, Ph.D. Thesis, School of Engineering, University of Buenos Aires (1999)
3. Barraza N. R., Pfefferman J. D., Cernuschi-Fras B., Cernuschi F.: An application of the chains-of-rare-events model to software development failure prediction, in Proc. 5th Int. Conf. Reliable Software Technologies. ser. Lecture Notes in Computer Science, H. B. Keller and E. Pldereder, Eds: Springer-Verlag, 1845, 185-195 (2000).
4. Blocklehurst S., Chan P. Y., Littlewood B., Snell J.: Recalibrating Software Reliability Models, IEEE Trans. on Soft. Eng., 16, 458-470 (1990)
5. Goel N. L., Okumoto K., Time-dependent error detection rate model for software reliability and other performances measures, IEEE Trans. on Reliability 28, 206-211 (1979)
6. Gokhale, Swapna S. and Trivedi, Kishor S.: Log-Logistic Software Reliability Growth Model, The 3rd IEEE International Symposium on High-Assurance Systems Engineering, pp. 34-41, IEEE Computer Society, Washington, DC, USA (1998)
7. Goseva-Popstojanova K., Trivedi K.: Failure Correlation in Software Reliability Models, IEEE Trans. on Reliability, 49, 37-48 (2000)
8. Hossain S. A., Dahiya R. C.: Estimating the Parameters of a Non-homogeneus Poisson-Process Model for Software Reliability, IEEE Transactions on Reliability, 42, 604-612 (1993)
9. Jelinski Z, Moranda P.: Software Reliability Research, in Statistical Computer Performance Evaluation, ed. W. Freiberger, New York, Academic Press, 465-484 (1972)
10. Jeske D. R. and Pham H.: On the Maximum Likelihood Estimates for the Goel-Okumoto Software Reliability Model, The American Statistician, 55, 219-222 (2001)
11. Littlewood B., Verral J., A Bayesian Reliability Growth Model for Computer software Reliability, Proc. IEEE Symposium on Computer Software Reliability, New York, 70-76 (1973)
12. Musa, J. D., Iannino A., and Okumoto K.: Software Reliability, Measurement, Prediction, Application, McGraw-Hill, (1989)
13. Musa, J. D., Okumoto K.: Logarithmic Poisson Execution Time Model for Software Reliability Measurement, Proc. 7th. Int. Conf. Software Eng., 230-238 (1984)
14. Plackett R. L.: The Truncated Poisson Distribution, Biometrics, 9, 485-488 (1953)
15. Ray, B. K., Liu Z., Ravishanker N.: Dynamic Reliability Models for Software Using Time-Dependent Covariates, ASA Technometrics, 48, 1-10 (2006)
16. Sahnioglu M.: Compound Poisson Software Reliability model, IEEE Trans. Soft. Eng. 18, 624-630 (1992)
17. Sahnioglu M. and Can U.: , Alternative Parameter Estimation Methods for the Compound Poisson Software Reliability Model with Clustered Failure Data, Software Testing, Verification and Reliability. 7, 35-37 (1997)
18. Stringfellow C. and Andrews A. A.: An Empirical Method for Selecting Software Reliability Growth Models, Empirical Software Engineering, 7, 319-343, (2002).
19. Tate R. F. and Goen R. L.: Minimum Variance Unbiased Estimation for the Truncated Poisson Distribution, Annals of Mathematical Statistics, 29, 755-765 (1958)
20. Tian J.: Better reliability assessment and prediction through data clustering, IEEE Trans. Soft. Eng. 28, 997-1007 (2002)
21. The Data & Analysis Center for Software, `http://www.thedacs.com`

22. Xie M., Hong G. Y., Wohlin C.: A Practical Method for the Estimation of Software Reliability Growth in the Early Stage of Testing, Proc. of 7th Intl. Symposium on Software Reliability Engineering, Albuquerque, USA, 116-123 (1997)
23. Yamada S., Ohba M., Osaki S.: S-Shaped Software Reliability Growth Models and Their Applications, IEEE Trans. Reliability, 33, 289-292, (1984)
24. Zhao M., Xie M.: On the Log-Power NHPP Software Reliability Model, Proc. of 3rd Intl. Symposium on Software Reliability Engineering, Research Triangle Park, North Carolina, 14-22. (1992)