EXCELLENT PUBLISHERS

International Journal of Current Research and Academic Review

ISSN: 2347-3215 Volume 2 Number 9 (September-2014) pp. 25-32

www.ijcrar.com

Software Reliability using SPRT: Log Power Model

R.Satya Prasad[1], B.Indira[2] and G.Krishna Mohan[3]*

[1]Department of Computer Science & Engineering, Acharya Nagrjuna University, Guntur, India
[2]St. Pauls college of Management & Information Technology, Hyderabad, India
[3]Department of Computer Science and Engineering, KL University, Guntur, India
***Corresponding author**

KEYWORDS

Log Power model, Decision lines, Software testing, Software failure data.

A B S T R A C T

In Classical Hypothesis testing volumes of data is to be collected and then the conclusions are drawn, which may need more time. But, Sequential Analysis of Statistical science could be adopted in order to decide upon the reliability / unreliability of the developed software very quickly. The procedure adopted for this is, Sequential Probability Ratio Test (SPRT). It is designed for continuous monitoring. The likelihood based SPRT proposed by Wald is very general and it can be used for many different probability distributions. The parameters are estimated using Maximum Likelihood Estimation method.In the present paper, we have applied the Log Power model to five sets of existing software reliability data and analyzed the results

Introduction

Wald's procedure is particularly relevant if the data is collected sequentially. Sequential Analysis is different from Classical Hypothesis Testing were the number of cases tested or collected is fixed at the beginning of the experiment. In Classical Hypothesis Testing the data collection is executed without analysis and consideration of the data. After all data is collected the analysis is done and conclusions are drawn. However, in Sequential Analysis every case is analyzed directly after being collected, the data collected upto that moment is then compared with certain threshold values, incorporating the new information obtained from the freshly collected case. This approach allows one to draw conclusions during the data collection, and a final conclusion can possibly be reached at a much earlier stage as is the case in Classical Hypothesis Testing. The advantages of Sequential Analysis are easy to see. As data collection can be terminated after fewer cases and decisions taken earlier, the savings in terms of human life and misery, and financial savings, might be considerable.

In the analysis of software failure data we often deal with either Time Between Failures or failure count in a given time interval. If it is further assumed that the average number of recorded failures in a given time interval is directly proportional to the length of the interval and the random number of failure occurrences in the interval is explained by a Poisson process then we know that the probability equation of the stochastic process representing the failure occurrences is given by a Homogeneous Poisson Process with the expression

$$(1.1)\ P\left[N(t)=n\right]=\frac{e^{-\lambda t}(\lambda t)^{n}}{n!}$$

Stieber (1997) observes that if classical testing strategies are used, the application of software reliability growth models may be difficult and reliability predictions can be misleading. However, he observes that statistical methods can be successfully applied to the failure data.

He demonstrated his observation by applying the well-known sequential probability ratio test (SPRT) of Wald (1947) for a software failure data to detect unreliable software components and compare the reliability of different software versions. In this paper we consider popular SRGM Log Power model (Zhao, 1992) and adopt the principle of Stieber (1997) in detecting unreliable software components in order to accept or reject the developed software. The theory proposed by Stieber (1997) is presented in Section 2 for a ready reference. Extension of this theory to the SRGM – Log Power model is presented in Section 3. Application of the decision rule to detect unreliable software with respect to the proposed SRGM is given in Section 4. Analysis of the application of the SPRT on five data sets and conclusions drawn are given in Section 5 and 6 respectively.

Wald's Sequential Test for a Poisson Process

The sequential probability ratio test (SPRT) was developed by A.Wald at Columbia University in 1943. Due to its usefulness in development work on military and naval equipment it was classified as 'Restricted' by the Espionage Act (Wald, 1947). A big advantage of sequential tests is that they require fewer observations (time) on the average than fixed sample size tests. SPRTs are widely used for statistical quality control in manufacturing processes. An SPRT for homogeneous Poisson processes is described below.

Let {N(t),t ≥0} be a homogeneous Poisson process with rate 'λ'. In our case, N(t) = number of failures up to time ' t' and ' λ' is the failure rate (failures per unit time). Suppose that we put a system on test (for example a software system, where testing is done according to a usage profile and no faults are corrected) and that we want to estimate its failure rate ' λ'. We can not expect to estimate ' λ' precisely. But we want to reject the system with a high probability if our data suggest that the failure rate is larger than λ_1 and accept it with a high probability, if it's smaller than λ_0. As always with statistical tests, there is some risk to get the wrong answers. So we have to specify two (small) numbers 'α' and 'β', where 'α' is the probability of falsely rejecting the system. That is rejecting the system even if $\lambda \leq \lambda_0$. This is the "producer's" risk. β is the probability of falsely accepting the system .That is accepting the system even if $\lambda \geq \lambda_1$. This is the "consumer's" risk. With specified choices of λ_0 and λ_1 such that $0 < \lambda_0 < \lambda_1$, the probability of finding N(t) failures in the time span (0,t) with λ_1, λ_0 as the failure rates are respectively given by

$$Q_1 = \frac{e^{-\lambda_1 t}\left[\lambda_1 t\right]^{N(t)}}{N(t)!} \quad (2.1)$$

$$Q_0 = \frac{e^{-\lambda_0 t}\left[\lambda_0 t\right]^{N(t)}}{N(t)!} \quad (2.2)$$

The ratio $\frac{Q_1}{Q_0}$ at any time 't' is considered as a measure of deciding the truth towards λ_1 or λ_0, given a sequence of $t_1 < t_2 < t_3 < ..... < t_K$ time instants say and the corresponding realizations of N(t). $N(t_1), N(t_2), ...N(t_K)$ Simplification of $\frac{Q_1}{Q_0}$ gives

$$\frac{Q_1}{Q_0} = \exp(\lambda_0 - \lambda_1)t + \left(\frac{\lambda_1}{\lambda_0}\right)^{N(t)}$$

The decision rule of SPRT is to decide in favor of λ_1 or to continue by observing the number of failures at a later time than 't' according as $\frac{Q_1}{Q_0}$ is greater than or equal to a constant say A, less than or equal to a constant say B or in between the constants A and B. That is, we decide the given software product as unreliable, reliable or continue the test process with one more observation in failure data, according as

$$\frac{Q_1}{Q_0} \geq A \quad (2.3)$$

$$\frac{Q_1}{Q_0} \leq B \quad (2.4)$$

$$B < \frac{Q_1}{Q_0} < A \quad (2.5)$$

The approximate values of the constants A and B are taken as

$$B \cong \frac{\beta}{1-\alpha}, \quad A \cong \frac{1-\beta}{\alpha}$$

Where 'α' and 'β' are the risk probabilities as defined earlier. A simplified version of the above decision processes is to reject the system as unreliable if N(t) falls for the first time above the $N_U(t) = a.t + b_2$ line (2.6)

to accept the system to be reliable if N(t) falls for the first time below the line

$$N_L(t) = a.t - b_1 \quad (2.7)$$

To continue the test with one more observation on (t, N(t)) as the random graph of [t, N(t)] is between the two linear boundaries given by equations (2.6) and (2.7) where

$$a = \frac{\lambda_1 - \lambda_0}{\log\left(\frac{\lambda_1}{\lambda_0}\right)} \quad (2.8)$$

$$b_1 = \frac{\log\left[\frac{1-\alpha}{\beta}\right]}{\log\left(\frac{\lambda_1}{\lambda_0}\right)} \quad (2.9)$$

$$b_2 = \frac{\log\left[\frac{1-\beta}{\alpha}\right]}{\log\left(\frac{\lambda_1}{\lambda_0}\right)} \quad (2.10)$$

The parameters can be λ_1 and λ_0, α, β chosen in several ways. One way suggested by Stieber (1997) is

$$\lambda_1 = q\frac{\lambda.\log(q)}{q-1}, \qquad \lambda_0 = \frac{\lambda.\log(q)}{q-1}$$

$$where\ q = \frac{\lambda_1}{\lambda_0}$$

If λ_0 and λ_1 are chosen in this way, the slope of $N_U(t)$ and $N_L(t)$ equals λ. The other two ways of choosing λ_0 and λ_1 are from past projects and from part of the data to compare the reliability of different functional areas.

Log Power model

Software reliability growth models (SRGM's) are useful to assess the reliability for quality management and testing-progress control of software development. They have been grouped into two classes of models concave and S-shaped. The most important thing about both models is that they have the same asymptotic behavior, i.e., the defect detection rate decreases as the number of defects detected (and repaired) increases, and the total number of defects detected asymptotically approaches a finite value. The Log Power NHPP model has several interesting properties, such as simple graphical interpretations and simple forms of the maximum likelihood estimates for the parameters. This model is characterized by the following mean value function: $m(t) = a\log^b(1+t)$.

Where, $a, b > 0, t \geq 0$. The failure intensity function of the model, which is defined as the derivative of the mean value function $m(t)$, is given by $\lambda(t) = \frac{ab\log^{b-1}(1+t)}{1+t}$.

Illustration: Parameter Estimation

We used cumulative time between failures data for software reliability monitoring. The use of cumulative quality is a different and new approach, which is of particular advantage in reliability. Using the estimators of 'a' and 'b' we can compute $m(t)$.

The likelihood function of Log-power model is given as,

$$L = e^{-a\log^b(1+t)}\prod_{i=1}^{N}\frac{ab\log^{b-1}(1+t_i)}{1+t_i} \quad (3.1.1)$$

Taking the natural logarithm on both sides, The Log Likelihood function is given as:

$$\log L = \sum_{i=1}^{n}\log\left(\frac{ab\log^{b-1}(1+t_i)}{1+t_i}\right) - a\log^b(1+t_n) \quad (3.1.2)$$

.

Taking the Partial derivative with respect to 'a' and equating to '0'.

$$a = \frac{n}{\log^b(1+t_n)} \quad (3.1.3)$$

Taking the Partial derivative of log L with respect to 'b' and equating to'0'.

$$b = \frac{n}{n\log\left(\log(1+t_n)\right) - \sum_{i=1}^{n}\log\left[\log(1+t_i)\right]} \quad (3.1.4)$$

Sequential Test for Software Reliability Growth Models

In Section 2, for the Poisson process we know that the expected value of N(t) = λt called the average number of failures experienced in time 't' .This is also called the mean value function of the Poisson process. On the other hand if we consider a Poisson process with a general function m(t) as its mean value function the probability equation of a such a process is

$$P[N(t) = Y] = \frac{[m(t)]^y}{y!}.e^{-m(t)}, y = 0,1,2,----$$

Depending on the forms of m(t) we get various Poisson processes called NHPP. For the Log Power model the mean value function is given as $m(t) = a\log^b(1+t)$ where $a > 0, b > 0$

We may write

$$Q_1 = \frac{e^{-m_1(t)}.[m_1(t)]^{N(t)}}{N(t)!}$$

$$Q_0 = \frac{e^{-m_0(t)}.[m_0(t)]^{N(t)}}{N(t)!}$$

are values of the mean $m_0(t), m_1(t)$ Where, value function at specified sets of its parameters indicating reliable software and P_1, P_0 unreliable software respectively. Let be values of the NHPP at two specifications

respectively. It $(b_0 < b_1)$ where b_0, b_1 of b say is b_1 at $m(t)$ can be shown that for our models . Symbolically b_0 greater than that at . Then the SPRT procedure is as $m_0(t) < m_1(t)$ follows:

Accept the system to be reliable

$$\frac{Q_1}{Q_0} \le B$$

$$\frac{e^{-m_1(t)}.[m_1(t)]^{N(t)}}{e^{-m_0(t)}.[m_0(t)]^{N(t)}} \le B \text{ i.e.,}$$

$$N(t) \le \frac{\log\left(\frac{\beta}{1-\alpha}\right) + m_1(t) - m_0(t)}{\log m_1(t) - \log m_0(t)} \text{ i.e.,}$$

(4.1)

Decide the system to be unreliable and reject if

$$\frac{Q_1}{Q_0} \ge A$$

$$N(t) \ge \frac{\log\left(\frac{1-\beta}{\alpha}\right) + m_1(t) - m_0(t)}{\log m_1(t) - \log m_0(t)} \text{ i.e.,}$$

(4.2)

Continue the test procedure as long as

$$\frac{\log\left(\frac{\beta}{1-\alpha}\right)+m_1(t)-m_0(t)}{\log m_1(t)-\log m_0(t)} < N(t) < \frac{\log\left(\frac{1-\beta}{\alpha}\right)+m_1(t)-m_0(t)}{\log m_1(t)-\log m_0(t)}$$

(4.3)

Substituting the appropriate expressions of the respective mean value function – m(t) of Log Power model we get the respective decision rules and are given in followings lines

Acceptance region:

$$N(t) \le \frac{\log\left(\frac{\beta}{1-\alpha}\right) + a\left[\log\left(1+t_i\right)^{b_1} - \log\left(1+t_i\right)^{b_0}\right]}{\log\left(\frac{\log\left(1+t_i\right)^{b_1}}{\log\left(1+t_i\right)^{b_0}}\right)}$$

(4.4)

Rejection region:

$$N(t) \ge \frac{\log\left(\frac{1-\beta}{\alpha}\right) + a\left[\log\left(1+t_i\right)^{b_1} - \log\left(1+t_i\right)^{b_0}\right]}{\log\left(\frac{\log\left(1+t_i\right)^{b_1}}{\log\left(1+t_i\right)^{b_0}}\right)}$$

(4.5)

It may be noted that in the above model the decision rules are exclusively based on the strength of the sequential procedure (α,β) and the values of the respective mean value . If the mean $m_1(t), m_0(t)$ functions namely, value function is linear in 't' passing through origin, that is, m(t) = λt the decision rules become decision lines as described by Stieber (1997). In that sense equations (4.1), (4.2) , (4.3) can be regarded as generalizations to the decision procedure of Stieber (1997). The applications of these results for live software failure data are presented with analysis in Section4.

SPRT Analysis of Live Data Sets

The developed SPRT methodology is for a software failure data which is of the form [t, N(t)]. Where,N(t) is the failure number of software system or its sub system in 't' units of time. In this section we evaluate the decision rules based on the considered mean value function for Fivedifferent data sets of the above form, borrowed from (Xie, 2002), (Pham, 2006) and (Ashoka, 2010). Based on the estimates of the parameter 'b' in each mean value function, we have chosen the specifications of $b_0 = b - \delta$, $b_1 = b + \delta$ equidistant on either side of estimate of b obtained through a Data Set to apply SPRT such that $b_0 < b < b_1$. Assuming the value of $\delta = 0.25$, the choices are given in the following table.

Table.5.1 Estimates of a, b & Specifications of b_0, b_1 for Time domain

Data Set	Estimate of 'a'	Estimate of 'b'	b_0	b_1
DS1 XIE	0.030479	3.650341	3.400341	3.900341
DS2 NTDS	0.033023	3.901233	3.651233	4.151233
DS3 IBM	0.022149	3.747340	3.497340	3.997340
DS4 ATT	0.073480	3.040257	2.790257	3.290257
DS5 SONATA	0.000084	6.341276	6.091276	6.591276

Using the selected b_0, b_1 and subsequently the $m_0(t), m_1(t)$ for the model, we calculated the decision rules given by Equations 5.3.4 and 5.3.5, sequentially at each 't' of the data sets taking the strength (α, β) as (0.05, 0.2). These are presented for the model in Table 5.2.The following consolidated table reveals the iterations required to come to a decision about the software of each Data Set.

Table.5.2 SPRT analysis for 5 data sets of Time domain data

Data Set	T	N(t)	Acceptance region (≤)	Rejection Region (≥)	Decision
DS1 Xie	30.02	1	0.273521	7.293050	
	31.46	2	0.438460	7.384110	
	53.93	3	2.684028	8.925140	
	55.29	4	2.805463	9.019324	
	58.72	5	3.106184	9.255769	
	71.92	6	4.198595	10.146840	
	77.07	7	4.600707	10.485165	*Reject*
	80.9	8	4.892114	10.733194	
	101.9	9	6.390735	12.039389	
	114.87	10	7.245995	12.802976	
	115.34	11	7.276120	12.830066	
	121.57	12	7.670113	13.185484	
	124.96	13	7.880500	13.376073	
	134.07	14	8.432798	13.878834	
DS2 NTDS	9	1	-2.875326	7.509721	*Reject*
	21	2	-0.028822	7.646324	
	32	3	1.943780	8.863185	
	36	4	2.603806	9.349697	
	43	5	3.706315	10.214626	
	45	6	4.010642	10.462338	

	50	**7**	4.753181	11.079558	
	58	**8**	5.892786	12.055368	
	63	**9**	6.578265	12.655436	
	70	**10**	7.507287	13.481132	
	71	**11**	7.637289	13.597669	
	77	**12**	8.404047	14.289327	
	78	**13**	8.529716	14.403343	
	87	**14**	9.635384	15.413438	
	91	**15**	10.113023	15.853246	
	92	**16**	10.231185	**15.962339**	
DS3 IBM	**10**	**1**	-2.971355	6.932089	***Continue***
	19	**2**	-1.471243	6.422993	
	32	**3**	-0.037051	6.882354	
	43	**4**	0.963592	7.471903	
	58	**5**	2.164280	8.326861	
	70	**6**	3.032216	9.006062	
	88	**7**	4.222592	9.990946	
	103	**8**	5.135112	10.775335	
	125	**9**	6.371809	11.867153	
	150	**10**	7.660839	13.030999	
	169	**11**	8.573699	13.867232	
	199	**12**	9.919319	15.113947	
	231	**13**	11.248142	16.358101	
	256	**14**	12.222737	17.277189	
	296	**15**	13.685312	18.664981	
DS4 ATT	**5.5**	**1**	-4.474704	9.341603	***Accept***
	7.33	**2**	-3.421839	8.106007	
	10.08	**3**	-2.483274	7.386119	
	80.97	**4**	**4.725894**	10.566211	
DS5 SONATA	**52.5**	**1**	-1.710861	4.560103	***Accept***
	105	**2**	-0.526262	5.098968	
	131.25	**3**	0.047634	5.508469	
	183.75	**4**	1.182868	6.424914	
	201.25	**5**	1.557790	6.745855	
	306.25	**6**	3.767336	8.730053	
	411.25	**7**	5.905189	10.729556	
	432.25	**8**	6.324252	11.126639	
	467.25	**9**	7.016583	11.785352	
	502.25	**10**	7.701449	12.439813	
	554.75	**11**	8.715198	13.412892	
	607.25	**12**	9.713315	14.375154	
	712.25	**13**	11.665425	16.266498	

	747.25	**14**	12.303772	16.887157
	799.75	**15**	13.250343	17.809134
	852.25	**16**	14.184288	18.720507
	887.25	**17**	14.800174	19.322328
	939.75	**18**	15.714270	20.216625
	1044.75	**19**	17.509171	21.975917
	1149.75	**20**	19.262583	23.698065
	1254.75	**21**	20.977607	25.385297
	1359.75	**22**	**22.656973**	27.039704

From the above table, a decision of either to accept, reject the system or continue is reached much in advance of the last time instant of the data.

Conclusion

The above consolidated table ofLog Power model as exemplified for 5 Data Sets indicates that the model is performing well in arriving at a decision. The model has given a decision of acceptance for 2 Data Setsi.e DS4 & DS5 at 4^{th} and 22^{nd} instances respectively, a decision of rejection for 2 Data Setsi.e DS1 & DS2 at 14^{th} and 16^{th} instances respectively and a decision of continue for 1 Data Set i.e DS3. Therefore, we may conclude that, applying SPRT on data sets we can come to an early conclusion of reliability / unreliability of software.

References

1. Michael. R. Lyu, (1996). “The hand book of software reliability engineering”, McGrawHill& IEEE Computer Society press.
2. Pham. H., (2006). “System software reliability”, Springer.
3. Satya Prasad (2007).”Half logistic Software reliability growth model “Ph.D Thesis of ANU, India.
4. Stieber, H.A. (1997). “Statistical Quality Control: How To Detect Unreliable Software Components”, Proceedings the 8th International Symposium on Software Reliability Engineering, 8-12.
5. Ashoka, M. (2010). “Sonata software limited data set”, Bangalore.
6. Wald. A., (1947). “Sequential Analysis”, John Wiley and Son, Inc, New York.
7. Wood, A. (1996). “Predicting Software Reliability”, IEEE Computer, 2253-2264.
8. Xie, M., Goh. T.N., Ranjan.P. (2002). “Some effective control chart procedures for reliability monitoring” -Reliability engineering and System Safety 77 143 - 150.
9. Zhao, M and Xie, M. (1992). “On the Log-Power NHPP Software Reliability Model”, IEEE, pp[14-22].