International Journal of Software Engineering.
ISSN 0974-3162 Volume 5, Number 1 (2014), pp. 7-16

http://www.irphouse.com

SPC on Ungrouped Data: Power Law Process Model

DR. R. Satya Prasad, Mrs. A. Srisaila and Dr. G.Krishna Mohan

Assoc. Professor, dept. Of cs&engg.
Acharya nagrjuna university
profrsp@gmail.com
Asst. Professor, V.R.Siddhartha Engg.College
sr.saila@gmail.com
Assoc.Professor, Dept. of CSE, KL University
km_mm_2000@yahoo.com

Abstract

The Duane model, also known as Power-Law Process model, is a two-parameter NonHomogeneous Poisson Process model which is widely used in software reliability growth modeling. In this paper, we propose to apply Statistical Process Control (SPC) to monitor software reliability process. A control mechanism is proposed based on the cumulative observations of failures which is ungrouped using mean value function of the Power Law Process model. The Maximum Likelihood Estimation (MLE) approach is used to estimate the unknown point estimate parameters of the model. The process is illustrated by applying to real software failure data.

Keywords: Power-law process, NonHomogeneous Poisson Process, Maximum Likelihood Estimation, SPC, Point estimation.

1. INTRODUCTION

Many software reliability models have been proposed in last 40 years to compute the reliability growth of products during software development phase. These models can be of two types i.e. static and dynamic. A static model uses software metrics to estimate the number of defects in the software. A dynamic model uses the past failure discovery rate during software execution over time to estimate the number of failures. Various software reliability growth models (SRGMs) exist to estimate the expected number of total defects (or failures) or the expected number of remaining defects (or failures).

The goal of software engineering is to produce high quality software at low cost. As, human beings are involved in the development of software, there is a possibility of errors in the software. To identify and eliminate human errors in software development process and also to improve software reliability, the Statistical Process Control concepts and methods are the best choice. SPC concepts and methods are used to monitor the performance of a software process over time in order to verify that the process remains in the state of statistical control. It helps in finding assignable causes, long term improvements in the software process. Software quality and reliability can be achieved by eliminating the causes or improving the software process or its operating procedures (Kimura et al., 1995).

The most popular technique for maintaining process control is control charting. The control chart is one of the seven tools for quality control. Software process control is used to secure, that the quality of the final product will conform to predefined standards. In any process, regardless of how carefully it is maintained, a certain amount of natural variability will always exist. A process is said to be statistically “in-control” when it operates with only chance causes of variation. On the other hand, when assignable causes are present, then we say that the process is statistically “out-of-control”. Control charts should be capable to create an alarm when a shift in the level of one or more parameters of the underlying distribution occurs or a non-random behavior comes into. Normally, such a situation will be reflected in the control chart by points plotted outside the control limits or by the presence of specific patterns. The most common non-random patterns are cycles, trends, mixtures and stratification (Koutras et al., 2007). For a process to be in control the control chart should not have any trend or nonrandom pattern. The selection of proper SPC charts is essential to effective statistical process control implementation and use. The SPC chart selection is based on data, situation and need (MacGregor and Kourti, 1995).

Chan et al.,(2000) proposed a procedure based on the monitoring of cumulative quantity. This approach has shown to have a number of advantages: it does not involve the choice of a sample size; it raises fewer false alarms; it can be used in any environment; and it can detect further process improvement. Xie et al.,(2002) proposed t-chart for reliability monitoring where the control limits are defined in such a manner that the process is considered to be out of control when one failure is less than LCL or greater than UCL. Assuming an acceptable false alarm α=0.0027 the control limits were defined. In section 5 of present paper, a method is presented to estimate the parameters and defining the limits. The process control is decided by taking the successive differences of mean values.

2. BACKGROUND THEORY

This section presents the theory that underlies NHPP models, the SRGMs under consideration and maximum likelihood estimation for ungrouped data. If ‘t’ is a continuous random variable with pdf: $f(t;\theta_1,\theta_2,\ldots,\theta_k)$. Where, $\theta_1,\theta_2,\ldots,\theta_k$ are k unknown constant parameters which need to be estimated, and cdf: $F(t)$. Where, The mathematical relationship between the pdf and cdf is given by: $f(t)=F^{'}(t)$. Let ‘a’ denote the expected number of faults that would be detected given infinite testing time. Then, the mean value function of the NHPP models can be written as: $m(t) = aF(t)$, where F(t) is a cumulative distribution function. The failure intensity function $\lambda(t)$ in case of NHPP models is given by: $\lambda(t) = aF'(t)$ (Pham, 2006).

2.1. NHPP MODEL

The Non-Homogenous Poisson Process (NHPP) based software reliability growth models (SRGMs) are proved to be quite successful in practical software reliability engineering. The main issue in the NHPP model is to determine an appropriate mean value function to denote the expected number of failures experienced up to a certain time point. Model parameters can be estimated by using Maximum Likelihood Estimate (MLE). Various NHPP SRGMs have been built upon various assumptions. Many of the SRGMs assume that each time a failure occurs, the fault that caused it can be immediately removed and no new faults are introduced. Which is usually called perfect debugging. Imperfect debugging models have proposed a relaxation of the above assumption.

2.2. POWER LAW PROCESS 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. This model is proposed by Duane(1964). The model has been applied in analyzing failure data of repairable systems. In software reliability context, this model has been discussed by many authors, see khoshgoftaar and woodstock(1991), Lyu and Nikora (1991). This model is characterized by the following mean value function: $m(t) = at^b$. 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 $\lambda(t) = abt^{b-1}$.

2.3. MAXIMUM LIKELIHOOD ESTIMATION

In much of the literature the preferred method of obtaining parameter estimates is to use the maximum likelihood equations. Likelihood equations are derived from the model equations and the assumptions which underlie these equations. The parameters are then taken to be those values which maximize these likelihood functions. These values are found by taking the partial derivate of the likelihood function with respect to the model parameters, the maximum likelihood equations, and setting them to zero. Iterative routines are then used to solve these equations. Unfortunately, the SRGM literature is sadly lacking in advice on which iterative routines to use, and with what starting values. This is unfortunate because the accuracy of parameter estimates and thus the accuracy of the models themselves greatly depend on the ability of the iterative search methods used to overcome local minima and find good values for the parameters.

If we conduct an experiment and obtain N independent observations, $t_1, t_2, \ldots, t_N$. The likelihood function may be given by the following product:

$$L(t_1, t_2, \ldots, t_N \mid \theta_1, \theta_2, \ldots, \theta_k) = \prod_{i=1}^{N} f(t_i; \theta_1, \theta_2, \ldots, \theta_k)$$

Likelihood function by using λ(t) is: $L = e^{-m(t)} \prod_{i=1}^{n} \lambda(t_i)$

Log Likelihood function for ungrouped data [10] is given as, $\log L = \sum_{i=1}^{n} \log[\lambda(t_i)] - m(t_n)$

The maximum likelihood estimators (MLE) of $\theta_1, \theta_2, \ldots, \theta_k$ are obtained by maximizing L or Λ, where Λ is ln L . By maximizing Λ, which is much easier to work with than L, the maximum likelihood estimators (MLE) of $\theta_1, \theta_2, \ldots, \theta_k$ are the simultaneous solutions of k equations such as: $\frac{\partial(\Lambda)}{\partial \theta_j} = 0$, j=1,2,…,k.

3. 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 Power Law Process model is given as,

$$L = e^{-a t_n^b} \prod_{i=1}^{N} a b t_i^{b-1} \tag{3.1}$$

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

$$\begin{aligned} \log L &= \sum_{i=1}^{n} \log\left(abt_i^{b-1}\right) - at_n^b \\ &= \log(a) + n\log(b) + (b-1)\sum_{i=1}^{n} \log(t_i) - a(t_n)^b \end{aligned} \tag{3.2}$$

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

$$\hat{a} = \frac{n}{t_n^{\hat{b}}} \tag{3.3}$$

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

$$\hat{b} = \frac{n}{\sum_{i=1}^{n} \ln\left(\frac{t_n}{t_i}\right)} \tag{3.4}$$

4. TIME DOMAIN FAILURE DATA SETS

The techniques examined here deal with data about the time at which failures occurred; or alternatively, data about the time between failure occurrences. These two forms can be considered equivalent. Although most software reliability growth models use data of this form, and such models have been in use for several decades, finding suitable data to verify models and improvement techniques is difficult. Early work generally focused on data based on calendar or wall clock time. Musa asserts that CPU execution time is a better measure than wall clock time, during which the actually time spent running a program can vary greatly based on CPU load, man hours, and other factors.

The performance of the model under consideration is exemplified by applying on the data sets given in tables 4.1 and 4.2.

Data Set #1: SONATA Software Limited
The data is collected for a project X in SONATA software Limited during the Testing phase (Ashoka, 2010).
Data Set #2: (Lyu, 1996)

Table 4.1: Data Set #1

Failure Number	Inter Failure Time	Failure Number	Inter Failure Time	Failure Number	Inter Failure Time
1	52.5	**11**	52.5	**21**	105
2	52.5	**12**	52.5	**22**	105
3	26.25	**13**	105	**23**	52.5
4	52.5	**14**	35	**24**	52.5
5	17.5	**15**	52.5	**25**	52.5
6	105	**16**	52.5	**26**	52.5
7	105	**17**	35	**27**	105
8	21	**18**	52.5	**28**	52.5
9	35	**19**	105	**29**	52.5
10	35	**20**	105	**30**	52.5

Table 4.2: Data Set #2

Failure Number	Inter Failure Time	Failure Number	Inter Failure Time	Failure Number	Inter Failure Time
1	0.5	**9**	1.4	**17**	3.2
2	1.2	**10**	3.5	**18**	2.5
3	2.8	**11**	3.4	**19**	2
4	2.7	**12**	1.2	**20**	4.5
5	2.8	**13**	0.9	**21**	3.5
6	3	**14**	1.7	**22**	5.2
7	1.8	**15**	1.4	**23**	7.2
8	0.9	**16**	2.7	**24**	10.7

5. ESTIMATED PARAMETERS AND THEIR CONTROL LIMITS

The estimated parameters and the calculated control limits of the Failure control Chart for Data Set#1 and Data Set #2 with the false alarm risk, α = 0.0027 are given in Table 5.1. Using the estimated parameters and the estimated limits, we calculated the control limits UCL=$m(t_U)$, CL=$m(t_C)$ and LCL=$m(t_L)$. They are used to find whether the software process is in control or not. The estimated values of 'a' and 'b' and their control limits are as follows.

$$t = (0.99865)^{1/b} = T_U$$

$$t = (0.5)^{1/b} = T_C$$

$$t = (0.00135)^{1/b} = T_L$$

Table 5.1: Parameter estimates and Control limits

Data Set	Estimated Parameters		Control Limits		
	a	**b**	**UCL=m(T_u)**	**CL=m(T_c)**	**LCL=m(T_l)**
LYU	0.988826	0.748933	0.987491	0.494413	0.001335
SONATA	0.019820	0.974573	0.019793	0.009910	0.000027

Goodness-of-fit

Model comparison and selection are the most common problems of statistical practice, with numerous procedures for choosing among a set of models proposed in the literature. Goodness-of-fit tests for this process have been proposed by (Park and Kim, 1992), (Rigdon, 1989). The AIC is a measure of the relative quality of a statistical model, for a given set of data. As such, AIC provides a means for model selection. AIC deals with the tradeoff between the goodness of fit of the model and the complexity of the model.

$$AIC = -2 * L + 2 * k$$

Where 'k' is the number of parameters in the statistical model, and 'L' is the maximized value of the likelihood function for the estimated model.

Given a set of candidate models for the data, the preferred model is the one with the minimum AIC value. Hence AIC not only rewards goodness of fit, but also includes a penalty that is an increasing function of the number of estimated parameters. This penalty discourages over-fitting. The Log likelihood and the AIC for 5 real life failure data sets for the power law process model is given in table 5.2 as follows.

Table 5.2: Goodness-of-fit for Power law process model

Data Set	Log L	AIC
LYU	-48.563908	101.127817
SONATA	-39.642182	83.284365
Xie	-95.102089	194.204178
NTDS	-50.386846	104.773692
ATT	-74.844868	153.689737

6. DISTRIBUTION OF TIME BETWEEN FAILURES

The mean value successive differences of time between failures cumulative data of the considered data sets are tabulated in Table 6.1 and 6.2. Considering the mean value successive differences on y axis, failure numbers on x axis and the control limits on Failure control chart, we obtained figure 6.1 and 6.2. A point below the control limit $m(t_L)$ indicates an alarming signal. A point above the control limit $m(t_U)$ indicates better quality. If the points are falling within the control limits it indicates the software process is in stable.

Table 6.1: Successive differences of mean values, LYU

F. No	C_TBF	m(t)	SD	F. No	C_TBF	m(t)	SD
1	0.5	0.588395	0.882938	**13**	26.1	11.378600	0.550641
2	1.7	1.471332	1.578896	**14**	27.8	11.929241	0.447138
3	4.5	3.050229	1.286923	**15**	29.2	12.376379	0.847489
4	7.2	4.337152	1.209782	**16**	31.9	13.223867	0.981469
5	10	5.546934	1.204390	**17**	35.1	14.205336	0.751171
6	13	6.751324	0.688585	**18**	37.6	14.956507	0.591928
7	14.8	7.439909	0.336314	**19**	39.6	15.548434	1.305230
8	15.7	7.776223	0.513719	**20**	44.1	16.853664	0.992103
9	17.1	8.289942	1.240621	**21**	47.6	17.845767	1.440910
10	20.6	9.530562	1.155206	**22**	52.8	19.286677	1.937759
11	24	10.685768	0.397686	**23**	60	21.224436	2.775564
12	25.2	11.083454	0.295146	**24**	70.7	24.000000	

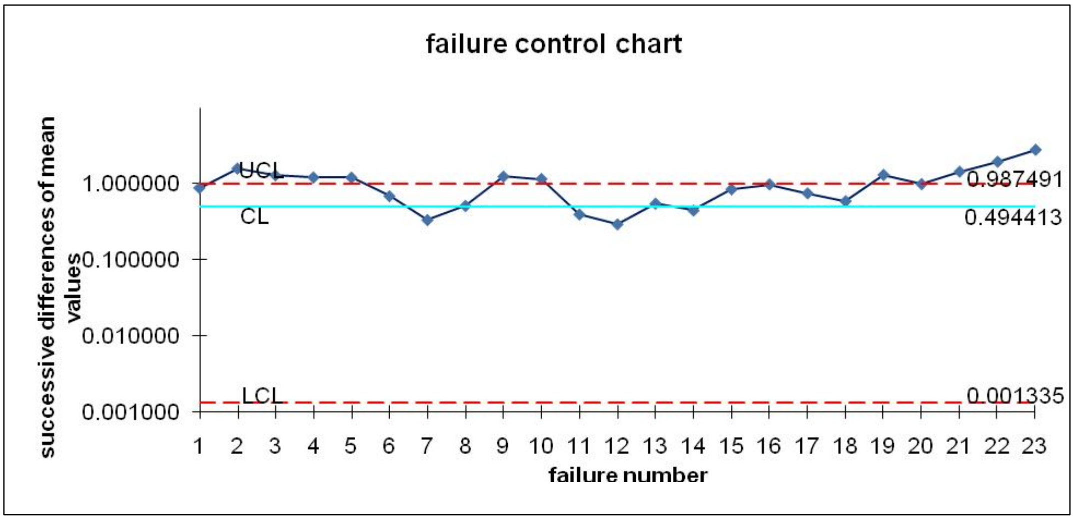

Figure 6.1: Failure control Chart of LYU

Table 6.1: Successive differences of mean values, SONATA

F. No	C_TBF	m(t)	SD	F. No	C_TBF	m(t)	SD
1	52.5	0.940860	0.907986	**16**	852.25	14.228392	0.569177
2	105	1.848845	0.449136	**17**	887.25	14.797569	0.852703
3	131.25	2.297981	0.891786	**18**	939.75	15.650272	1.701839
4	183.75	3.189767	0.295716	**19**	1044.75	17.352111	1.697488
5	201.25	3.485483	1.762191	**20**	1149.75	19.049599	1.693545
6	306.25	5.247674	1.746578	**21**	1254.75	20.743144	1.689941
7	411.25	6.994252	0.347850	**22**	1359.75	22.433085	0.843710
8	432.25	7.342102	0.578805	**23**	1412.25	23.276795	0.842913
9	467.25	7.920908	0.577703	**24**	1464.75	24.119708	0.842145
10	502.25	8.498611	0.864657	**25**	1517.25	24.961853	0.841404
11	554.75	9.363268	0.862576	**26**	1569.75	25.803257	1.680686
12	607.25	10.225843	1.719617	**27**	1674.75	27.483943	0.839328
13	712.25	11.945460	0.571723	**28**	1727.25	28.323271	0.838679
14	747.25	12.517184	0.856319	**29**	1779.75	29.161950	0.838050
15	799.75	13.373502	0.854890	**30**	1832.25	30.000000	

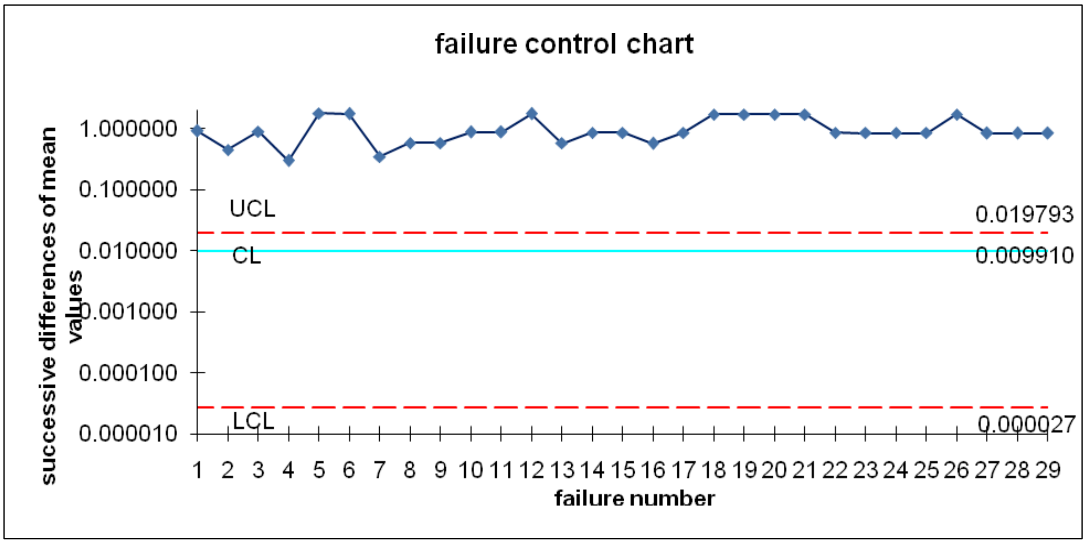

Figure 6.2: Failure control Chart of SONATA

7. CONCLUSION

The given Time between failures data are plotted through the estimated mean value function against the failure serial order. The graphs have shown out of control signals i.e below the LCL. Hence we conclude that our method of estimation and the control chart are giving a positive recommendation for their use in finding out preferable control process or desirable out of control signal. By observing the Control chart it is identified that, for DS#1 the failure process out of UCL. For DS#2 the failure situation is detected at 15th point below LCL. Hence our proposed Control Chart detects out of control situation. Many of the successive differences have gone out of upper control limits for the present model.

References

1. Ashoka. M.,(2010), “Sonata Software Limited” Data Set, Bangalore.
2. Chan, L.Y, Xie, M., and Goh. T.N., (2000), “Cumulative quality control charts for monitoring production processes. Int J Prod Res; 38(2):397-408.
3. Duane, J.T., (1964). “Learning curve approach to reliability monitoring”, IEEE Trans. Aerospace, AS-2, pp.[563-566].
4. Gaudoin,O., Yang, B. and Xie, M. (March, 2003). “A simple goodness-of-fit test for the power-law process, based on the duane plot”, IEEE trans. Rel., Vol. 52, No, 1.
5. Khoshgoftaar, T.M. and Woodstock, T.G. (1991). “Software reliability model selection: a case study”, Proc. Int. Symp. On software reliability engineering, may 18-19, Austin, Texas, pp.[183-191].
6. Kimura, M., Yamada, S., Osaki, S., 1995. ”Statistical Software reliability prediction and its applicability based on mean time between failures”. Mathematical and Computer Modeling Volume 22, Issues 10-12, Pages 149-155.
7. Koutras, M.V., Bersimis, S., Maravelakis,P.E., 2007. “Statistical process control using shewart control charts with supplementary Runs rules” Springer Science + Business media 9:207-224.
8. Lyu, M.R. and Nikora, A. (1991). “A Heuristic approach for software reliability prediction: the equally weighted linear combination model. Proc. Int. Symp. On software reliability engineering, may 18-19, Austin, Texas, pp.[172-181].
9. Lyu, M.R., (1996). “Handbook of Software Reliability Engineering”, McGraw-Hill, New York..
10. MacGregor, J.F., Kourti, T., 1995. “Statistical process control of multivariate processes”. Control Engineering Practice Volume 3, Issue 3, March 1995, Pages 403-414 .
11. Park, W. J and Kim, Y.G., (1992). “Goodness-of-fit tests for the power law process”, IEEE trans. Rel., Vol. 41, pp. 107-111.
12. Park, W. J and Scoh, M., (1994). “More Goodness-of-fit tests for the power law process”, IEEE trans. Rel., Vol. 43, pp. 275-278.
13. Pham. H., 2006. “System software reliability”, Springer.
14. Rigdon, S.E., (1989). “Testing goodness-of-fit for the power law process”, communications in statistics: theory and methods, Vol.18, pp.4665-4676.
15. Xie. M, T.N Goh and P.Ranjan. (2002). “Some effective control chart procedures for reliability monitoring”, Reliability Engineering and System Safety. 77, 143-150.

Authors Profile:

Dr. R. Satya Prasad received Ph.D. degree in Computer Science in the faculty of Engineering in 2007 from Acharya Nagarjuna University, Andhra Pradesh. He received gold medal from Acharya Nagarjuna University for his outstanding performance in Masters Degree. He is currently working as Associate Professor in the Department of Computer Science & Engineering, Acharya Nagarjuna University. His current research is focused on Software Engineering. He has published 75 papers in National & International Journals.

Mrs. A.Srisaila received her M.Tech (CSE) from Bapatla Engineering College, Acharya Nagarjuna University. Presently she is working as Asst. Professor in Department of Information Technology in V.R.Siddhartha Engineering College. Her research area includes Software Reliability Engineering, Graphical Passwords, Human Computer Interaction.

Dr. G. Krishna Mohan is working as an Assoc. Professor in the Department of CSE, KL University, Vaddeswaram, Guntur. He obtained his M.C.A degree from Acharya Nagarjuna University in 2000, M.Tech from JNTU, Kakinada, M.Phil from Madurai Kamaraj University and Ph.D from Acharya Nagarjuna University. His research interests lies in Data Mining and Software Engineering. He has 14 years of teaching experience. He has qualified APSLET in 2012. He published 28 papers in National & International journals.