IJFANS INTERNATIONAL JOURNAL OF FOOD AND NUTRITIONAL SCIENCES

ISSN PRINT 2319 1775 Online 2320 7876

Research paper UGC CARE Listed (Group -I) Journal Volume 11, Iss 12, Dec 2022

Pareto Type IV SRGM using Order Statistics Approach

B.N.V.Uma Shankar[1] , Research Scholar and Senoir Grade Asst.Prof, Department of Basic Sciences & Humanities, CVR College of Engineering, Hyderabad, India.. Email: balinauma@gmail.com
Prof. K. Rosaiah[2]**,** Department of Statistics, Acharya Nagarjuna University, India

Abstract:
In the modern society, the usage of computers is increasing day by day and is used in diverse areas. A lot of computer applications exist in almost every area in ever day life, reliability becomes a very extent, since in the matter of economy. Finally for producing software with high reliability, to develop a number of models and to make necessary measuring and its control of functionality. There are several software reliability growth models that can be used to forecast software system reliability. The impact of software reliability on the product enhancement process is critical. The non-homogeneous Poisson process is a probabilistic model for determining software reliability (NHPP). For time domain data, order statistics are a new way for measuring software reliability based on NHPP. The Pareto Type IV model as a software reliability growth model is used in this study to generate the expressions for an efficient reliability function. The maximum likelihood (ML) estimation approach is used to determine the parameters. When the live datasets have been analysed, the findings are displayed.

Keywords:
NHPP, Order Statistics, Pareto Type IV distribution, , ML Estimation, Software Reliability.

Introduction:
Software reliability is the forecast, estimation and assessment of software-based systems using statistical approaches applied to data acquired during system development and operation [1]. Software reliability engineering research has been carried out, and a number of NHPP software reliability growth models have been suggested to analyse software reliability. Once the mean value function has been defined, software reliability can be estimated. Over the last three decades, there has been a lot of research on software reliability engineering, and a lot of statistical models for evaluating software reliability. The majority of current software reliability prediction approaches rely entirely on the observation of software product failures, and producing an accurate reliability prediction requires a substantial amount of failure data. For achieving reliable software; the developer mainly focuses on its quality. At the time of the implementation process of the software, testing is usually a prolonged process. The bugs are detected as time elapses and are used for determining the testing time which is required in order to meet various criterions of reliability. To estimate the reliability of the software which is commonly used devised statistical model and describing a prototypical behaviour of the debugging process. In the course of the past three decades, there had been some statistical models developed for estimating software reliability. For predicting software program reliability most of the present models are merely primarily based on the monitoring of software

Research paper© 2012 IJFANS. All Rights Reserved, UGC CARE Listed (Group -I) Journal Volume 11, Iss 12, Dec 2022

failures data to attain accurate reliability.

For estimating reliability of software, several SRGMs are existed and are used at the time of testing period of the software development process. At the software testing period, testing is done to identify the errors in the system and corrected them. The software reliability growth models are necessary for finding out goodness-of-fit, predictability, reliability and so on. Various SRGMS are made for the last three decades and they were provided valuable information by improving reliability [2][3][4]. The primary purpose of these models is to increase software performance. Some of the models that we use to describe software reliability growth are standard Gompertz, Crow-AMSAA, Lloyd-Lipow and modified Gompertz [5].

Reliability and Availability are the two user requirements for the software. Reliability is required when the products non-performance has the greatest impact. On the other hand, the availability is required when the downtime of the system functioning. It is very complicate to tell that software reliability, is probabilistically because we can 't tell that the reliability of the product is said to be 100%, if the working behaviour of the software is correct and the reliability is 0% if the working of the software is incorrect. Many models have been developing with different statistical strategies to adopt distinct testing environments [2]. An often-used approach for computing reliability of the software is by considering an analytical model which the parameters are estimated from accessible software failure data. Reliability and other relevant measures like performance, goodness of fit is computed from the fitted model [6].

Because software is built by humans, it is more likely to have flaws, and there has been ongoing study into constructing software reliability growth models in this regard.

This study uses the pareto type IV model with order statistics to assess the software system's reliability. The major goal of this research is to provide a model that can be used to calculate software performance.

Research Methodology

The most important and measurable aspect of software quality is software reliability, which is strongly focused on the client. It is possible to assess how effectively a program meets its operational requirements using software reliability. Measures of software reliability can help with quantitative design goals and resource scheduling. These actions also aid in project resource management [7].

The user will benefit from the software reliability measure as well, because the user is primarily concerned with the system's failure-free operation. If the operational needs in terms of quality are not accurately specified, the user will either receive a system at an exorbitant price or with an exorbitant operational cost. The probabilistic approach is the most commonly used approach in developing software reliability models [8].

ISSN PRINT 2319 1775 Online 2320 7876

Research paper UGC CARE Listed (Group -I) Journal Volume 11, Iss 12, Dec 2022

There are a variety of software reliability models that can be used, all of which are based on probabilistic assumptions. Error seeding models, failure rate models, and curve fitting models are only a few of the categories. The failure rate models use stochastic processes like the Homogeneous Poisson Process, Non-Homogeneous Poisson Process, and Compound Poisson Process to characterize the failure process. NHPP is used in the bulk of the models.

Because of the errors in the system, a software system is prone to failures at random times. Let {N(x), x>0} be a counting process that represents the total number of failures at time x. Because there are no failures at x=0, we have

$$N(0) = 0$$

Let $m(t)$ is the mean value function to represent the predicted number of software failures by time 't'. It is finite valued, bounded, non-negative and non-decreasing with the boundary conditions.

$$m(t)\begin{cases} = 0, \; t = 0 \\ = a, \; t \to \infty \end{cases}$$

Where 'a' represents the estimated number of software defects to be discovered over time.

Assume $N(t)$ has a poisson probability mass function with parameters $m(t)$ i.e.,

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

then $N(t)$ is called an NHPP. As a result, the process can be used to represent the stochastic nature of software failure occurrences. Various time domain models [9] have been proposed in the literature to characterise the stochastic failure process by an NHPP with different mean value functions [10][11].

The proposed mean-value function $m(t)$ for the Pareto type IV model is given as

$$m(t) = a\left\{1 - \left[1 + \left(\frac{t}{c}\right)^{-b}\right]\right\}$$

a, b, and c are unknown parameters that must be calculated using the Newton Raphson technique in order to test software reliability. For the Pareto type IV model, expressions for estimating 'a', 'b', and 'c' are now given.

$$p\{N(t) = n\} = \frac{[m(t)]^n e^{-m(t)}}{n!}$$

$$\lim_{n\to\infty} p\{N(t)=n\} = \frac{e^{-a}.a^n}{n!}$$

This is also a Poisson model with mean ‘a’.

Let S_k be the time between $(k-1)^{th}$ and k^{th} failure of the software product. It is assumed that X_k be the time up to the k^{th} failure. The probability that X_k exceeds a real number ‘x’ given that the total time up to the $(k-1)^{th}$ failure is equal to s and is given as

i.e., $P\left[S_k > \frac{s}{X_{k-1}} = x\right]$

$$R\ S_k/X_{k-1}(s/x) = e^{-[m(x+s)-m(s)]}$$

The above equation is the function for software reliability.

Parameter Estimation

In software reliability prediction, parameter estimation is critical. The parameters are estimated using the well-known Maximum Likelihood Estimation (MLE) technique once we know the analytical solution for m(t) for the specified model.

The mean value of the Pareto type IV distribution model representing the number of failures encountered at time 't' is given by

$$m(t) = a\left\{1-\left[1+\left(\frac{t}{c}\right)^{-b}\right]\right\} \quad (1)$$

We need to raise m(t) to the power r to group the time domain data into non-overlapping sequential sub groups of size r.

$$m(t) = a\left\{1-\left[1+\left(\frac{t}{c}\right)^{-b}\right]\right\}^r \quad (2)$$

The constants ‘a’, ‘b’, and ‘c’ in the mean value function are referred to as the proposed model's parameters.

Differentiating Equation (2) with respect to ‘t’, we get

$$m^{'}(t) = r\left[a\left(1-\frac{1}{\left(1+\frac{t}{c}\right)^b}\right)\right]^{r-1} \frac{ab}{c\left(1+\frac{t}{c}\right)^{b+1}} \quad (3)$$

The Likelihood function L can be written as

$$L = e^{-m(t)} \prod_{i=1}^{n} m^1(t_i) \tag{4}$$

Substituting Equations (1) and (3) in Equation (4) we get

$$Log\ L = -a^r \left[1 - \frac{1}{\left(1+\frac{t}{c}\right)^b}\right]^r + \sum_{i=1}^{n} \log r + \sum_{i=1}^{n} (r-1)\log\left(a - \frac{a}{\left(1+\frac{t}{c}\right)^b}\right) + \sum_{i=1}^{n}\left(\log a + \log b - \log c - (b+1)\log\left(1+\frac{t}{c}\right)\right)$$

Differentiate LogL with respect to ‘a’, and equating to 0 (i.e., $\frac{\partial LogL}{\partial a} = 0$) we get

$$\frac{\partial \log L}{\partial a} = -r\, a^{r-1}\left[1 - \frac{1}{\left(1+\frac{t}{c}\right)^b}\right]^r + \sum_{i=1}^{n}(r-1)\left[\frac{1}{a - \frac{a}{\left(1+\frac{t}{c}\right)^b}}\right]\frac{\partial}{\partial a} a\left[1 - \frac{1}{\left(1+\frac{t}{c}\right)^b}\right] + \frac{n}{a}$$

$$\frac{\partial LogL}{\partial a} = 0$$

$$\therefore a^r = n\left[\frac{(t+c)^b}{(t+c)^b - c^b}\right]^r \tag{5}$$

The parameters a, b and c would be the solutions of the equations

$$\frac{\partial LogL}{\partial b} = g(b) = 0$$

$$LogL = -a^r \left[1 - \frac{1}{\left(1+\frac{t}{c}\right)^b}\right]^r + \sum_{i=1}^{n} \log r + \sum_{i=1}^{n} (r-1)\log\left(a - \frac{a}{\left(1+\frac{t}{c}\right)^b}\right) + \sum_{i=1}^{n}\left(\log a + \log b - \log c - (b+1)\log\left(1+\frac{t}{c}\right)\right)$$

Differentiate Log L with respect to ‘b’ and then equate to 0

$$g(b) = \left[\left(\frac{nr}{(t+1)^b - 1}\right)\log\left(\frac{1}{1+t}\right)\right] - \sum_{i=1}^{n}\frac{r-1}{(1+t_i)^b - 1}\log\left(\frac{1}{1+t_i}\right) + \frac{n}{b} - \sum_{i=1}^{n}\log(1+t_i) \tag{6}$$

$$g^1(b) = -nr\log\left(\frac{1}{1+t}\right)\frac{(t+1)^b\log(t+1)}{\left[(t+1)^b - 1\right]^2} + \tag{7}$$

$$\sum_{i=1}^{n}(r-1)\log\left(\frac{1}{1+t_i}\right)\frac{(t_i+1)^b\log(1+t_i)}{\left[(1+t_i)^b - 1\right]^2} - \frac{n}{b^2}$$

Differentiating Log L with respect to ‘c’ and equating to 0.

(i.e., $\frac{\partial LogL}{\partial c} = 0$. we get

$$g(c) = \frac{nr}{(t+c)} + \sum_{i=1}^{n}(r-1)\left(\frac{-1}{t_i + c}\right) - \frac{n}{c} + \sum_{i=1}^{n}\frac{2t}{\left(1+\frac{t_i}{c}\right)c^2} \tag{8}$$

ISSN PRINT 2319 1775 Online 2320 7876

 UGC CARE Listed (Group -I) Journal Volume 11, Iss 12, Dec 2022

$$g^1(c) = \frac{-nr}{(t+c)^2} + \sum_{i=1}^{n}(r-1)\frac{1}{(t_i+c)^2} - \frac{n}{c^2} + \sum_{i=1}^{n}\frac{-4t_i^2}{(t_i+c)^2 c^3} \quad (9)$$

Data Analysis

The Reliability is calculated using 4th and 5th-order statistics [12] for a single dataset. The Musa dataset was used to assess the software's reliability, and the results are presented here.

Table- I: Software failure data reported by Musa (1975)

Failure No.	Time between Failures (Hrs)	Failure No.	Time between Failures (Hrs)	Failure No.	Time between Failures (Hrs)	Failure No.	Time between Failures (Hrs)
1	3	35	227	69	529	103	108
2	30	36	65	70	379	104	0
3	113	37	176	71	44	105	3110
4	81	38	58	72	129	106	1247
5	115	39	457	73	810	107	943
6	9	40	300	74	290	108	700
7	2	41	97	75	300	109	875
8	91	42	263	76	529	110	245
9	112	43	452	77	281	111	729
10	15	44	255	78	160	112	1897
11	138	45	197	79	828	113	447
12	50	46	193	80	1011	114	386
13	77	47	6	81	445	115	446
14	24	48	79	82	296	116	122
15	108	49	816	83	1755	117	990
16	88	50	1351	84	1064	118	948
17	670	51	148	85	1783	119	1082
18	120	52	21	86	860	120	22
19	26	53	233	87	983	121	75
20	114	54	134	88	707	122	482
21	325	55	357	89	33	123	5509
22	55	56	193	90	868	124	100
23	242	57	236	91	74	125	10
24	68	58	31	92	2323	126	1071
25	422	59	369	93	2930	127	371
26	180	60	748	94	1461	128	790
27	10	61	0	95	843	129	6150
28	1146	62	232	96	12	130	3321
29	600	63	330	97	261	131	1045
30	15	64	365	98	1800	132	648
31	36	65	1222	99	865	133	5485
32	4	66	543	100	1435	134	1160
33	0	67	10	101	30	135	1864
34	8	68	16	102	143	136	4116

Research paper © 2012 IJFANS. All Rights Reserved, UGC CARE Listed (Group -I) Journal Volume 11, Iss 12, Dec 2022

Table- II: MUSA Dataset (4th Order Statistics)

Failure No	4th Order Time between Failures Sk days	4th Order cumulative Time between Failures $x_n = \sum s_k days$	Failure No	4th Order Time between Failures Sk days	4th Order cumulative Time between Failures $x_n = \sum s_k days$
1	227	227	18	1081	16358
2	217	444	19	1929	18287
3	315	759	20	2280	20567
4	297	1056	21	3560	24127
5	930	1986	22	4333	28460
6	690	2676	23	3948	32408
7	1758	4434	24	5246	37654
8	655	5089	25	4361	42015
9	300	5389	26	281	42296
10	991	6380	27	6000	48296
11	1067	7447	28	3746	52042
12	475	7922	29	1401	53443
13	2336	10258	30	3042	56485
14	917	11175	31	6166	62651
15	1384	12559	32	2242	64893
16	927	13486	33	11164	76057
17	1791	15277	34	12625	88682

In 88682 days, 34 failures for 4th order statistics were recorded in the MUSA dataset. The MLEs of a, b, and c for the dataset can be obtained by solving the equations in Section III using the Newton Raphson technique.

$$a = 3.407049$$
$$b = 0.110178$$
$$c = 1.217387$$

At any period, x beyond 88682 days, the estimator of the reliability function is given by

$$R\ S_k/X_{k-1}(s/x) = e^{-[m(x+s)-m(s)]}$$

$$R\frac{S_{35}}{X_{34}}(88682/7922) = e^{-[m(7922+88682)-m(88682)]}$$

=0.990732597

Table- III: MUSA Dataset (5th Order Statistics)

Failure No	5th Order Time between Failures Sk days	5th Order cumulative Time between Failures $x_n = \sum s_k days$	Failure No	5th Order Time between Failures Sk days	5th Order cumulative Time between Failures $x_n = \sum s_k days$
1	342	342	15	1573	17758
2	228	570	16	2809	20567
3	398	968	17	5343	25910
4	1018	1986	18	3451	29361
5	1112	3098	19	8281	37642
6	1951	5049	20	4373	42015

7	275	5324	21	3391	45406
8	1056	6380	22	4010	49416
9	1264	7644	23	3905	53321
10	2445	10089	24	3164	56485
11	893	10982	25	6176	62661
12	1577	12559	26	11703	74364
13	2149	14708	27	10202	84566
14	1477	16185			

In 84566 days, the MUSA dataset has 27 failures for 5th order statistics. We can get the MLE's of a, b, and c for the dataset by using the Newton Raphson technique to solve the equations in Section III.

$$a = 2.72465$$
$$b = 0.11072$$
$$c = 1.197433$$

At any period x beyond 84566 days, the estimator of the reliability function is given by

$$R\frac{S_{35}}{X_{34}}(84566/17758) = e^{-[m(17758+84566)-m(84566)]}$$

$$=0.983607444$$

Conclusion

This work describes a pareto type IV distribution model with order statistics that can be used to measure software reliability. Today, 70 to 80 percent of people use software, therefore producing trustworthy software is critical. One real data set was used to test the suggested model. For 4th and 5^{th}-order statistics, expressions were constructed and parameters were estimated. The model's reliability was tested for 4th and 5^{th}-order statistics, and the findings showed that it is extremely reliable. The proposed pareto type IV with order statistics model gave excellent results and is very convenient to use for calculating reliability.

References

1. DAC,(2006) http://www.thedacs.com/databases/url/key.php?keycode=2
2. Lyu, M.R.(1996), "The Hand Book of Software Reliability Engineering", McGrawHill & IEEE Computer Society Press.
3. Yamada, S. and Osaki, S. (1985), "Software Reliability Growth Modeling: Models and Applications", IEEE Transactions on Software Engineering, Vol.SE-11, 1431-1437.
4. Quadri, S. M. K., Ahmad, N., Peer, M.A. and Kumar, M(2010), "Software Reliability Growth modelling with NewModified Weibull Testing–effort and Optimal Release Policy", International Journal of Computer Applications.
5. Reliability Hotwire, 2008, E-magazine for the Reliability Professional
6. Goel, A.L.(1982), "Software Reliability Modeling and Estimation Techniques", Rep. RADC-TR- 82-263.
7. Wood, A. (1996) "Predicting Software Reliability", IEEE Computer, 2253-2264.
8. Musa, J.D. (1980), "The Measurement and Management of Software Reliability", Proceeding of the IEEE Vol.68, No.9, 1131-1142.
9. R.R.L. Kantam and R.Subba Rao, 2009, "Pareto Distribution: A Software Reliability Growth model ", International Journal of Performability Engineering. Volume 5, Number 3, April 2009, Paper 9,PP: 275-281.

ISSN PRINT 2319 1775 Online 2320 7876

 UGC CARE Listed (Group -I) Journal Volume 11, Iss 12, Dec 2022

10. Pham. H.. 2003. "Handbook of Reliability Engineering ", Springer.
11. Goel. A.L and Okumoto. K., "A Time-dependent error-detection rate model for software and other performance measures," IEEE Trans. Reliability, vol R-28, Aug, pp. 206 – 211, 1979
12. Sita Kumari K. and Satya Prasad R. (2014), "Pareto Type II Software Reliability Growth Model– An Order Statistics Approach", International Journal of Computer Science Trends and Technology (IJCST), Volume 2, Issue 4, Jul-Aug 2014, 49-54.