How to Cite:
Krishna, V. R., Rao, T. S., & Roy, L. D. (2022). Arranged sample in GHLD for software reliability growth model. *International Journal of Health Sciences*, *6*(S2), 1853–1861. https://doi.org/10.53730/ijhs.v6nS2.5400

Arranged sample in GHLD for software reliability growth model

V. Rama Krishna
Professor, Vignan's Foundation for Science, Technology and Research, Vadlamudi, Guntur
Email: vramakrishna2006@gmail.com

T Sbhamastan Rao
Associate Professor, CMR Technical Campus, Hyderabad
Email: mastan1061@gmail.com

Lakshmi Deepika Roy
Assistant Professor, CMR Technical Campus, Hyderabad
Email: deepikaroy.cse@cmrtc.ac.in

***Abstract*---**As the size and complexity increases, it is difficult to generate reliable software for the users. Reliability depends on number of failures which create a great loss in the software system. Though many software reliability growth models exist, but for time domain data can be handled through Arranged sample approach and one can construct NHPP leads to reliability function and formulate SRGM. In this paper, an attempt is made to present the GHLD type - I model with arranged sample as a software reliability growth model and derive the expressions for Reliability function that facilitates to compute the reliability of a product. Maximum likelihood estimation procedure is used to estimate the parameters of the model. Through the analysis of live data sets the results are exhibited.

***Keywords*---**GHLD type I, arranged sample, maximum likelihood estimation, SRGM.

Introduction

Programming dependability is perhaps the main attributes of programming quality. Its estimation and the executives advancements utilized during the product life cycle are fundamental for creating and looking after quality/solid programming frameworks[14]. Programming Reliability is the likelihood of disappointment free activity of programming in a predetermined climate during determined time[1][13]. Over the most recent a very long while, numerous

Software Reliability Growth Models (SRGMs) like Crowand Basu (1988), Goel Okumoto(1979,1984), Musa(1980), Pham(2005), Lutfiah Ismail A turk1 & Wejdan Saleem Al ahmadi (2018)[15] and a few models have been created to enormously work with specialists and administrators in following and estimating the development of unwavering quality as programming is being improved[2]. The principle objective in fostering these models is to improve the product execution. During the testing period of a product item, the disappointment information will be gathered and these models anticipate the future framework operability dependent on the disappointment information. As the product item is created by humanities bound to have flaws and in such manner there was a persistent examination going on in fostering the product unwavering quality development models. The majority of the models accept that the time between disappointments follows a remarkable appropriation so the boundaries differ with the blunders staying in the product framework.

This paper presents the Generalized Half Logistic Type I model with orchestrated example to investigate the dependability of the product framework. Section II portrays the Generalized Half Logistic type - I model with organized example and mean worth capacity for the fundamental NHPP, Section III clarifies the boundary assessment for Generalized Half Logistic Type I The principle objective of this paper is to foster a model that gives a quantifiable programming execution. The design of this paper is as per the following: Section III with masterminded test approach considering the time area information, Section IV depicts the strategy used to examine the disappointment informational indexes for live applications and segment V alludes to end.

Order generalized half logistic type I model

Programming dependability is the most significant and most quantifiable part of programming quality and it is very client situated. With programming Reliability it is feasible to quantify how well the program capacities in gathering its operational prerequisites. Programming unwavering quality measures can advance quantitative particular of plan objectives and timetables the assets as required. These actions additionally help in the better administration of undertaking assets [4]. The client will likewise be profited by programming unwavering quality measure, in light of the fact that the client is principally worried about the disappointment free activity of the framework. In the event that the operational requirements concerning quality are precisely indicated, the client will either get a framework at an unreasonably exorbitant cost or with an unnecessarily high operational expense .The most well-known methodology in fostering the product dependability models is the probabilistic methodology [3]. The probabilistic model addresses the disappointment events and the issue expulsions as probabilistic occasions. There are numerous software reliability models available for use according to probabilistic assumptions. They are classified into various groups, including error seeding models, failure rate models, curve fitting models, reliability growth models, Markov structure Models and non-homogenous passion process (NHPP) models[2]. Most of the models are NHPP based models.

A software system is subject to failures at random times due to the errors present in the system. Let {N(t), t>0} be a counting process representing the cumulative

number of failures by time t. Since there are no failures at t=0 we have N(0) = 0. It is assumed that the number of software failures during non-overlapping time intervals do not affect each other. It can be mentioned that for finite times t1<t2<t3<….<tn , the n random variables N(t1), {N(t2)-N(t1)}, ….. {N(tn)-N(tn-1)} are independent. It implies that the counting process {N(t), t>0} has independent increments. Let m(t) denote the expected number of software failures by time 't'. Since the expected number of errors remaining in the system at any time is finite, m(t) is bounded, non-decreasing function of 't' with the boundary conditions.

M(t) = 0, t=0
= a, as t -> ∞

Where a is the expected number of software errors need to be detected. Assume that N(t) is known to have a Poisson Probability mass function with parameters m(t) i.e.,

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

Where N(t) is called NHPP. The behavior of software failure phenomena can be illustrated through N(t) process. Several time domain models exist in the literature [7] which specify that the mean value function m(t) will be varied for each NHPP process The mean value function of Generalized Half Logistic Type I [16] software reliability growth is given by

$$m(t) = \left[\frac{1 - e^{-bt}}{1 + e^{-bt}}\right]^{\theta}$$

Here, we consider the performance given by the Generalized Half Logistic Type I software reliability growth model based on order statistics and whose mean value function is given by

$$m(t) = a^r \left[\frac{1 - e^{-bt}}{1 + e^{-bt}}\right]^{\theta r}$$

Where [m(t)/a] is the cumulative distribution function of Ordered Generalized Half Logistic distribution type – I model

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

N(t)= N(∞)-N(t)

E [N(t)]= E[N(∞)]- E[N(t)]

$$= a - a\left[\frac{1-e^{-bt}}{1+e^{-bt}}\right]^{\theta}$$

$$= a\left(1 - \left[\frac{1-e^{-bt}}{1+e^{-bt}}\right]^{\theta}\right)$$

Let Sk be the time between (k-1)th and Kth failure of the software product. It is assumed that Yk be the time up to the Kth failure. We need to find out the

probability of the time between (k-1)th and Kth failures. The Software Reliability function is given by

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

Parameter Estimation For Order Generalized Half Logisitc Type I Model

In this section, the expressions are generated for estimating the parameters of the Ordered Generalized Half Logistic Type I model based on the time between the failures. The expressions for a, b, and c has to be derived. Let S1, S2,…. be a sequence of times between consecutive software failures associated with an NHPP N(t). Let yk be equal to

$$\sum_{i=1}^{k} s_i\,, k = 1,2,3\ldots$$

This represents the time at which failure k occurs. Suppose we are given with 'n' software failure times say Y1, Y2, … , Yn, there are 'n' time instants at which the first, second, third …. N th failure of software is observed. The mean value function of Ordered Generalized Half Logistic Type I is given

$$m(t) = a^r\left[\frac{1-e^{-bt}}{1+e^{-bt}}\right]^{\theta r}$$

The constants a and b in the mean value function are called parameters of the proposed model. To assess the software reliability, it is necessary to compute the expressions for finding the values of a, b. For doing this, Maximum Likelihood estimation is used whose Likelihood function is given by

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

The maximum likelihood estimators (MLEs) are the one that maximize the Likelihood function 'L' and the method is called maximum likelihood method of estimation. [5]Differentiating m(t) with respect to 't'

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

$$\text{Log L} = \log\left[e^{-m(t_i)}\prod_{i=1}^{n} m'(t_i)\right]$$

$$= -m(t_i) + \sum_{i=1}^{n} log\left[\frac{2a^r\,\theta r e^{-bt_i}(1-e^{-bt_i})^{\theta r-1}}{(1+e^{-bt_i})^{\theta r+1}}\right]$$

$$= -a^r\left[\frac{1-e^{-bt_i}}{1+e^{-bt_i}}\right]^{\theta r} + \sum_{k=1}^{n}[\log 2 + \log b + r\log a + \log\theta + \log r - bt_i]$$

$$+(\Theta r\text{-}1)\log(1-e^{-bt_i}) - (\Theta r+1)\log(1+e^{-bt_i})]$$

$$\frac{1}{L}\frac{\partial L}{\partial a} = -ra^{r-1}\left[\frac{1-e^{-bti}}{1+e^{-bt_i}}\right]^{\Theta r} + \sum_{i=1}^{n}\left[\frac{r}{a}+0\right]$$

$$= \frac{r}{a}\left[-a^{r}\left[\frac{1-e^{-bt_i}}{1+e^{-bi}}\right]^{\Theta r} + n\right]$$

$$\frac{1}{L}\frac{\partial L}{\partial a} = 0 \quad \Rightarrow \quad a^{r} = n\left[\frac{1-e^{-bt}}{1+e^{-bt}}\right]^{\Theta r} \qquad 3.1$$

$$\frac{1}{L}\frac{\partial L}{\partial b} = -\frac{2a^{r}\,\Theta r e^{-bt_i}(1-e^{-bt_i})^{\Theta r-1}}{(1+e^{-bti})^{\Theta r+1}} + \sum_{i=1}^{n}\left[\frac{n}{b} - t_i + be^{-bt_i}\left(\frac{\Theta r-1}{1-e^{-bt_i}} + \frac{\Theta r+1}{1+e^{-bt_i}}\right)\right]$$

On simplification,

$$g(b) = -\frac{2a^{r}\,\Theta r e^{-bt_i}\left(1-e^{-bti}\right)^{\Theta r-1}}{\left(1+e^{-bti}\right)^{\Theta r+1}} + \frac{n}{b} - nt_i + \sum_{i=1}^{n}\left[\frac{2be^{-bt_i} - e^{-bt_i}}{1-e^{-2bt_i}}\right] \qquad 3.2$$

$$g'(b) = \frac{2ne^{-bti}\left[(1-bs_n)(1-e^{-2bs_n})-2b\,2\,e^{-2bt_i}\right]}{\left(1-e^{-2bt_i}\right)^{2}} + \frac{n}{b^{2}}$$

$$+ 2\theta r\sum_{i=1}^{n}\frac{e^{bt_i}-e^{-bt_i}-bt_i\left(e^{bt_i}-e^{-bt_i}\right)}{\left(e^{bt_i}-e^{-bt_i}\right)^{2}} \text{-}2\text{n}\sum_{i=1}^{n}\frac{e^{2bt_i}-1-2bt_ie^{2bt_i}}{\left(e^{2bt_i}-1\right)^{2}} \qquad 3.3$$

Arranged sample

Masterminded test can be utilized for a few applications. They can be utilized in a few applications like information pressure, endurance investigation, Study of Reliability and numerous others[9]. Allow Y to indicate a consistent irregular variable with likelihood thickness work f(y) and combined conveyance work F(y), and let (Y1 , Y2 , … , Yn) mean an arbitrary example of size n drawn on Y. The first example perceptions might be unordered as for size. A change is needed to create a comparing requested example. Let (Y(1) , Y(2) , … , Y(n)) signify the arranged arbitrary example to such an extent that Y(1) < Y(2) < … < Y(n); at that point (Y(1), Y(2), … , Y(n)) are all things considered known as the masterminded test got from the parent Y. The different distributional attributes can be known from Balakrishnan and Cohen [10]. The between disappointment time information address the time pass between each two successive disappointments. Then again if a sensible hanging tight an ideal opportunity for disappointments is certainly not a major issue, we can bunch the between disappointment time information into non covering progressive sub gatherings of size 4 or 5 and add the disappointment times with in each sub gathering. For example if an information of 100 interfailure times are accessible we can bunch them into 20 disjoint subgroups of size 5. The whole in every subgroup would dedicate the time slip by between each fifth organized example in an example of size 5. Overall for between disappointment information of size 'n', if r (any characteristic no) not as much as

'n' and ideally a factor n, we can helpfully partition the information into 'k' disjoint subgroups (k=n/r) and the aggregate absolute in every subgroup demonstrate the time between each rth disappointment. The likelihood circulation of such a period slip by would be that of the arranged measurements in a subgroup of size r, which would be equivalent to force of the dispersion capacity of the first factor (m(t)). The entire interaction includes the numerical model of the mean worth capacity and information about its boundaries. In the event that the boundaries are referred to they can be taken as they are for the further examination, if the boundaries are not realized they must be assessed utilizing an example information by any allowable, proficient strategy for assessment. This is fundamental on the grounds that as far as possible rely upon mean worth capacity, which thus relies upon the boundaries. On the off chance that product disappointments are very regular monitoring between disappointment is dreary. On the off chance that disappointments are more continuous orchestrated example are best [11].

Data Analysis

To analyze softwre reliability, CSR3 Data Set [12] has been considered.

Table.1 CSR3 Data Set (Michael R.Lyu., 1996a)

No	Time Between failures	Failure No.	Time Between failures	FailureNo	Time Between Failures(hrs)	Failure No	Time Between failures
1	33	27	10	53	1	79	20
2	9	28	2	54	400	80	79
3	4	29	22	55	294	81	24
4	66	30	53	56	227	82	540
5	0.5	31	19	57	118	83	52
6	18	32	58	58	13	84	1596
7	149	33	20	59	47	85	314
8	14	34	3	60	89	86	1
9	15	35	92	61	242	87	763
10	50	36	5	62	99	88	10
11	81	37	66	63	607	89	20
12	34	38	289	64	83	90	144
13	85	39	3	65	2	91	28
14	54	40	9	66	26	92	56
15	3	41	12	67	586	93	476
16	15	42	18	68	708	94	65
17	6	43	9	69	6	95	98
18	8	44	75	70	4	96	884
19	130	45	15	71	55	97	212
20	19	46	291	72	409	98	287
21	19	47	212	73	36	99	53
22	112	48	4	74	15	100	3
23	15	49	5	75	573	101	831
24	16	50	308	76	583	102	43

25	154	51	269	77	60	103	55
26	50	52	276	78	19	104	109

Table. 2 CSR3 Data Set (4th and 5th order) (Michael R.Lyu., 1996a)

Failure No	4th order time between	4th order cumulative time between failures	5th order time between failures Sk	5th order cumulative time between failures
1	112	112	112.5	112.5
2	181.5	293.5	246	358.5
3	180	473.5	257	615.5
4	157	630.5	178	793.5
5	163	793.5	316	1109.5
6	162	955.5	137	1246.5
7	216	1171.5	192	1438.5
8	152	1323.5	372	1810.5
9	120	1443.5	129	1939.5
10	367	1810.5	820	2759.5
11	114	1924.5	1240	3999.5
12	522	2446.5	494	4493.5
13	858	3304.5	1033	5526.5
14	922	4226.5	1330	6856.5
15	267	4493.5	1088	7944.5
16	1031	5524.5	761	8705.5
17	1322	6846.5	2526	11231.5
18	474	7320.5	938	12169.5
19	1207	8527.5	723	12892.5
20	178	8705.5	1439	14331.5
21	2212	10917.5		
22	1088	12005.5		
23	248	12253.5		
24	1523	13776.5		
25	555	14331.5		
26	1038	15369.5		

The CSR3 data set consists of 26 failures for 4th order statistics in 15369 days. By solving the equations in section III by Newton Raphson method , we can obtain the MLE's of a and b when θ = 2 for CSR3 data set.

a^ = 27.026768
b^ = 1.000576

The estimator of the reliability function at any time x beyond 15369.5 days is given by

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

$$R_{\frac{S_{27}}{X_{26}}}(15369.5/955.5) = e^{-[m(955.5+15369.5)-m(15369.5)]}$$

= 0.999669

The CSR3 data set consists of 20 failures for 5th order statistics in 14331 days. By solving the equations in section III by Newton Raphson method , we can obtain the MLE's of a and b when θ = 2 for CSR3 data set.

a^ = 21.00698
b^ = 0.999933

The estimator of the reliability

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

$$R\frac{S_{27}}{X_{26}}(15369.5/955.5) = e^{-[m(955.5+15369.5)-m(15369.5)]}$$

$$= 0.999765$$

Similarly we can obtain a, b values and reliability for θ = 3 presented in the table below:

Table 5.1 Computed Reliability figures for different θ, a and b values of the model

θ	Order	a	b	Reliability
2	4	27.02676	1.000576	0.999669
	5	21.00698	0.999933	0.999765
3	4	28.26547	1.000783	0.999842
	5	22.32759	1.000325	0.999784

Conclusions

In this paper, the Generalized half logistic distribution type – I model with request measurements has been proposed. Today 70 to 80 % of individuals use programming and it is a lot of fundamental for produce dependable programming. The proposed model has been tried with live informational collection for fourth and fifth order as shown in the table 5.1 and demonstrated that it has high dependability. It is likewise seen that the unwavering quality is high for fifth order measurement than fourth order insights. At last it tends to be presumed that the model has created generally excellent outcomes and is especially agreeable to register the unwavering quality.

References

1. Musa J.D, Software Reliability Engineering MCGraw-Hill, 1998.
2. Pham. H (2005) "A Generalized Logistic Software Reliability Growth Model", Opsearch, Vol.42, No.4, 332-331.
3. Musa,J.D. (1980) "The Measurement and Management of Software Reliability", Proceeding of the IEEE vol.68, No.9, 1131-1142
4. WOOD, A. predicting software Reliability, IEEE Computer, 1996; 2253-2264
5. Sitakumari.k, Satya Prasad.R , Assessing Software Quality with Time Domain Pareto Type II using SPC SPC, IJCA, 2014
6. Dr.R.Satya Prasad, NGeetha Rani, Prof R.R.L.Kantham, Pareto Type II

Based Software Reliability Growth Model, International Journal of Software Engineering, Vol (2), 2011

7. R.R.L.Kantam and R.Subbarao, 2009. “Pareto Distribution: A Software Reliability Growth Model”. International Journal of Performability Engineering, Volume 5, Number 3, April 2009, Paper 9, PP: 275- 281.
8. J.D.Musa and K.Okumoto,”A Logorithmic Poisson Execution time modelfor software reliability measure-ment”, proceeding seventh international conference on software engineering, orlando, pp.230-238,1984
9. Arak M. Mathai ;Order Statistics from a Logistic Dstribution and Applications to Survival and Reliability Analysis;IEEE Transactions on Reliability, vol.52, No.2; 2003
10. Balakrishnan.N., Clifford Cohen; Order Statistics and Inference; Academic Press inc.;1991
11. K.Ramchand H Rao, R.Satya Prasad, R.R.L.Kantham; Assessing Software Reliability Using SPC – An Order Statistics Approach; IJCSEA Vol.1, No.4, August 2011
12. Michael R.Lyu 1996a, Handbook of Software Reliability Engineering.
13. C. Jin and S.-W. Jin, “Parameter optimization of software reliability growth model with S-shaped testing-effort function using improved swarm intelligent optimization,” *Applied Soft Computing*, vol. 40, pp. 283–291, 2016.
14. DalilaAmara, Latifa BenArfa Rabai Towards a New Framework of Software Reliability Measurement Based on Software Metrics**”,** Volume 109, 2017, Pages 725-730
15. Lutfiah Ismail A turk1 & Wejdan Saleem Al ahmadi “Comparative Study of the Non-Homogeneous Poisson Process Type-I Generalized Half-Logistic Distribution”, International Journal of Statistics and Probability; Vol. 7, No. 6; November 2018.
16. V. Rama Krishna, R R L Kantam and T Subhamastan Rao “A Software Quality measurement using Generalized Half Logistic Distribution”, International Journal of Advanced Science and Technology (2020), Vol. 29, No.3, pp. 9665-9669.