QATAR UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

INFERENCE IN THE LOG-LOGISTIC DISTRIBUTION BASED ON AN ADAPTIVE PROGRESSIVE TYPE-II CENSORING SCHEME

BY

MAHA F. SEWAILEM

A Thesis Submitted to

the Faculty of the College of Arts and Sciences

in Partial Fulfillment of the Requirements for the Degree of

Masters of Science in Applied Statistics

June 2019

COMMITTEE PAGE

The members of the Committee approve the Thesis of Maha Faiq Sewailem defended on 26/03/2019.

Ayman Bakleezi
Thesis/Dissertation Supervisor

Mohammad Salehi
Committee Member

Faiz Elfaki
Committee Member

Approved:

Rashid Al-Kuwari, Dean, College of College of Arts and Sciences

ABSTRACT

SEWAILEM, MAHA, FAIQ, Masters: June: 2019, Applied Statistics

Title: Inference in the Log-Logistic Distribution Based on an Adaptive Progressive Type-II Censoring Scheme

Supervisor of Thesis: Advisor's Ayman, Suleiman, Bakleezi.

The primary aim of this study is to explore the maximum likelihood estimation (MLE) and the Bayesian approach to estimate the parameters of log-logistic model and calculate the approximate confidence interval for the parameters and the survival function in both methods based on an adaptive progressive type-II censoring scheme. The parameters of the probability distribution are estimated via the Newton-Raphson Method and the Bayes estimators, based on squared error loss function (SELF). The approximate confidence interval for the reliability function has been calculated using the delta method; the approximate credible intervals for the unknown parameters and the survival function using the Bayesian approach have been constructed using Markov Chain Monte Carlo (MCMC) method. Moreover, a Monte Carlo study has performed to examine the proposed methods under different situations, based on mean squared error, bias, coverage probability, and expected length estimated criteria. Application to real life data is included, in order to view how the proposed methods, work in practice. It is observed that the Bayesian approach is better than MLE for estimating the log-logistic model parameters.

Key words: Maximum likelihood estimation (MLE), Bayesian estimation, adaptive progressive type-II censoring scheme, squared error loss function (SELF)

DEDICATION

This work is dedicated to the three greatest persons in my life:

my mother, my father and my husband.

I could not have done it without them.

Thank you for believing in me.

ACKNOWLEDGMENTS

First, I would like to thank Professor Ayman Bakleezi; all stages of this work were performed under his supervision and guidance. A special thanks to our committee members Professor Mohammad Salehi and Dr. Faiz Elfaki for their recommendations and comments, which raised my awareness and helped me improve my research. Moreover, a big thanks to my husband, mother, and father, who encouraged and supported me through all the successive stages to complete this thesis. Finally, many thanks to my children, sisters, friends, and to the school administrator Mrs. Zainab Al-Reyashi for their help and support.

TABLE OF CONTENTS

DEDICATION iv

ACKNOWLEDGMENTS v

LIST OF TABLES ix

LIST OF FIGURES x

CHAPTER 1: INTRODUCTION 1

1.1 General Background and key Terms 1

1.1.1 Lifetime Analysis 5

1.1.2 Censored Data 6

1.1.3 Type I-Censored Data 6

1.1.4 Type-II Censored Data 6

1.1.5 Progressive Censoring Scheme 6

1.1.6 Progressive Type-I Censored Data 6

1.1.7 Progressive Type-II Censored Data 7

1.1.8 Type-II Progressive Hybrid Censored Data 7

1.1.9 Adaptive Progressive Type-II Censored Data 7

1.2 Literature Review 8

1.2.1 Inference Based on the Log-Logistic Distribution 8

1.2.2 Inference Based on an Adaptive Progressive Type-II Censoring Scheme 11

1.3 Problem Statement 15

1.4 Significance of the Study 16

1.5 Objective of the Study 17

1.6 Specific Objectives 17

1.7 The Scope of the Study 18

CHAPTER 2: MAXIMUM LIKELIHOOD AND BAYES ESTIMATION 19

2.1 Brief Overview of the Maximum Likelihood Inference 19

2.1.1 Log-Logistic Distribution 20

2.1.2 Likelihood Based on an Adaptive PT-II Censored Data 22

2.1.3 Maximum Likelihood Estimation 23

2.2 Asymptotic Confidence Intervals for μ and σ 25

2.3 Approximate Confidence Interval for Survival Function 27

2.4 Brief Overview of the Bayesian Approach 28

2.4.1 Bayesian Estimation 29

CHAPTER 3: SIMULATION STUDY, RESULTS AND COMPARISONS 34

3.1 Simulation Study 34

3.1.1 Convergence Diagnostic 37

3.2 Results and Comparisons 43

CHAPTER 4: REAL DATA ANALYSIS 58

CHAPTER 5: SUMMARY, CONCLUSION AND SUGGESTIONS FOR FURTHER STUDY 68

5.1 Summary 68

5.2 Conclusion 69

5.3 Suggestions for Further Study 70

REFERENCES 72

APPENDIX 75

APPENDIX A: QUANTILE FUNCTION OF LOG-LOGISTIC DISTRIBUTION 75

APPENDIX B: R CODE FOR A MONTE CARLO SIMULATION STUDY 76

APPENDIX C: R CODE FOR REAL LIFE EXAMPLE 114

LIST OF TABLES

Table 1: The Different Progressive Censoring Scheme R with Different Choices of n and m 35

Table 2: Mean, Bias and MSE of Location & Scale Parameters Based on T=1........ 44

Table 3: Mean, Bias and MSE of S(t) Based on T=1 46

Table 4: Mean, Bias and MSE of Location & Scale Parameters Based on T=1.8........ 48

Table 5: Mean, Bias and MSE of S(t) Based on T=1.8 49

Table 6: Expected Length of 95% CI. & Coverage Probability for Location & Scale Parameters Based on T=1........ 52

Table 7: Expected Length of 95% CI. & coverage Probability for S(t) Based on T=1........ 53

Table 8: Expected Length of 95% CI. & Coverage Probability of Location & Scale Parameters Based on T=1.8........ 55

Table 9: Expected Length of 95% CI. & Coverage Probability of S(t) Based on T=1.8........ 56

Table 10: Real Data Set on Breaking Stress of Carbon Fibers (in Gpa)........ 58

Table 11: An Adaptive Progressive Type-II Censored Samples of Real Data Based on T= (1.60,3.66) 60

Table 12: Point Estimations of the Location & Scale Parameters & S (0.5) Based on a Real Data Set........ 66

Table 13: 95% Confidence Intervals and Length of the Location & Scale Parameters & S (0.5) Based on Real Data Set 67

LIST OF FIGURES

Figure 1: Illustration of an adaptive type-II progressive censoring scheme 5

Figure 2: Plots of the log-logistic pdf for some different parameter's values 21

Figure 3: Plots of the log-logistic survival function for some different parameter's values 22

Figure 4: Trace plot and probability density plot of the location parameter 38

Figure 5: Trace plot and probability density plot of the scale parameter 39

Figure 6: Autocorrelation plot of MCMC output for the location parameter 40

Figure 7: Autocorrelation plot of MCMC output for the scale parameter 40

Figure 8: Smooth probability density plot for the location parameter with burn-in=1000 41

Figure 9: Smooth probability density plot for the scale parameter with burn-in=1000 41

Figure 10: Histogram of the random draws of the location parameter with burn-in=1000 42

Figure 11: Histogram of the random draws of the scale parameter with burn-in=1000 43

Figure 12: Trace plot and probability density plot of the location parameter based on a real data 62

Figure 13: Trace plot and probability density plot of the scale parameter based on a real data 62

Figure 14: Autocorrelation plot of the location parameter based on a real data 63

Figure 15: Autocorrelation plot of the scale parameter based on a real data.................... 64

Figure 16: Smooth probability density plot for the location parameter based on a real data.. 64

Figure 17: Smooth probability density plot for the scale parameter based on a real data.. 65

CHAPTER 1: INTRODUCTION

This chapter provides the study background by introducing the log-logistic distribution, the problem statement, justification for the study, the general and specific objectives, and the definition of a few key terms.

1.1 General Background and key Terms

Balakrishnan N. (1992) stated the probability density function (pdf) of a logistic distribution with random variable X as the following:

$$f(x;\mu,\sigma)=\frac{\pi}{\sigma\sqrt{3}}\frac{e^{\frac{-\pi(x-\mu)}{\sigma\sqrt{3}}}}{\left[1+e^{\frac{-\pi(x-\mu)}{\sigma\sqrt{3}}}\right]^2} \quad ,-\infty<x<\infty\,,-\infty<\mu<\infty\,,\sigma>0$$

Let the random variable X have a logistic distribution with mean μ, variance σ^2, and supposed T=e^x; then, T has a log-logistic distribution. Therefore, the corresponding density function of this distribution is as follows:

$$f(t)=\frac{\pi}{\sigma t\sqrt{3}}\frac{e^{\frac{-\pi(\ln t-\mu)}{\sigma\sqrt{3}}}}{\left[1+e^{\frac{-\pi(\ln t-\mu)}{\sigma\sqrt{3}}}\right]^2} \quad ,\ 0<t<\infty\ , \qquad -\infty<\mu<\infty\,, \qquad 0<\sigma<\infty$$

This form of the distribution was used to compare two unknown estimators of the log-logistic parameters based on censored and un censored samples by AL-Haj Ebrahem and Baklizi (2005). In statistical approach, the log-logistic distribution which noted in economics as the Fisk distribution (Fisk, 1961). This distribution is widely used in life testing experiment. The most common distinctive of this lifetime model is that, the logarithm of the lifetime variable is logistically distributed due to the specific properties of this distribution. Also, it has a resembling shape to the log-normal distribution, although it has heavier tails. It is used in lifetime data analysis as a kind of a parametric model. For instance, the mortality rate from cancer resulting medication. Additionally, it has been used to model the stream flow and precipitation in hydrology and in economics as a simple model for distributing wealth or income. Moreover, this distribution is useful in networking, to model the transmission times of data which keeps in mind both the network and the software. Furthermore, the log-logistic distribution is a highly popular distribution, which can be considered as a substitute to the Weibull distribution in real-life data analysis. Moreover, in this distributional model on contrast to the log-normal distribution, the cumulative distribution function has an explicitly closed written form; this cause it useful for analysing the real-life or clinical data with censoring such as lung cancer data. In life testing experiments, we are faced with censored data (Lawless, 1982). However, the researchers may not often have enough time to observe the life time for all the test units in the experiment. Decreasing the duration time of the experiment and the relate cost is main the sense for censoring. For example, in some of real-life applications, the experimenters must deal with some types of censored sample, due to the time limitation in the experiment which avoid the experimenter to observe the life time of all units. A censoring scheme, which make equity between (i) the total experimental duration time; (ii) the number of experimental units; and (iii) the performance of statistical inference in the experiment result which is adorable. The most regular censoring schemes are type-I censoring – where the experiment stop at a predetermined time T – and type-II censoring – where the experiment stop upon the *m* specified failure times obtained. However, these regular censoring schemes do not have the ability for removing objects at each failure time except at the last failure time observed. Due to this absence, a more public censoring scheme has been imported. The progressive type-II censoring scheme has engaged much application in real-life testing experiment; however, this censoring scheme may not be applicable in some area of real-life testing; due to pre-determined values of progressive censoring scheme. Thus, the continuing censoring scheme of progressive type-II is a type-II hybrid progressive censoring scheme proposed by Kundu and Joarder (2006). According to the primary objective in life testing experiments, which aim to reducing the test duration time and the related expenditure of the experiment which yield a high efficiency in statistical inference, this censoring scheme may not be appropriate or will not be very efficient. Besides, the complete observed failure times *m* is not fixed in an advance (random) and an inadequate observed number *m* may not be satisfied in statistical inference. This thesis has thus suggested a combination of type-I censoring and type-II progressive censoring schemes, known as an adaptive type-II progressive censoring scheme, provided for real life studies that takes in to account a process of adaptation. Consider *n* an identical, independent units in a reliability experiment; let *m* and *n* are pre-determined early. In addition, let the progressive censoring scheme R= $(R_1, \cdots\cdots, R_m)$ provided before starting the experiment; however, some of the removal units during the test may change due to un satisfied *m* observed failure times. Additionally, the experimental total time may run over the pre-fixed time T. According to the research studied by Ng et al. (2009) , the first situation in this censoring scheme explained in Figure 1(a) below as a pre-determined number of observed failure time satisfied before time T (i.e. $X_{m:m:n} < T$), while the second situation explained below in Figure 1(b) as the experimental total time exceed a pre-determined time T due to the assured assumption of obtaining *m* observed failure time (i.e. $X_{m:m:n} > T$). If the test duration time pass the predetermined time T , the duration time of the experiment will not go far from the previous fixed time due to primary concept of order statistics (David & Nagaraja, 2003).This concept implies that the experiment speed to terminate by avoiding removing survival units at points after the experiment pass the predetermined time T except at the time of the last failure observed. Thus, the expected duration time of the test will be smaller (Balakrishnan N. , 2007). Assume J represent the number of failure times observed before the predetermined time T, i.e.,

$$X_{J:m:n} \leq T < X_{J+1:m:n,} \qquad J = 0,1, \cdots \cdots, m$$

where $X_{0:m:n} \equiv 0$ and $X_{m+1:m:n} \equiv \infty$ .When the total time has passed the ideal test time T, we set $R_{J+1} = \cdots \cdots = R_{m-1} = 0 \; and \; R_m = n - m - \sum_{i=1}^{J} R_i$. In this situation, we do not remove any survival units except at the time of m^{th} failure; this allows us to acceleration the experiment to end as soon as possible and this modification on progressive censoring scheme is satisfied when *(J+1)*th observed failure time exceeds the predetermined ideal test time for J+1<*m*. The predetermined value of ideal total test time T is act as a major factor in determining the progressive censoring scheme and as an adjustment between a lower experiment time and a higher number of observing failure times. The first sever situation is occurred when the ideal total test time T approach to infinity ($T \rightarrow \infty$); in this case the time is not important in the experiment – we thus have a normal progressive type-II censoring scheme. While the second sever case can occur when the ideal total test time equal to zero ($T = 0$); in this case, the experiment end quickly, and this censoring scheme tends to the familiar type-II (failure) censoring scheme. Furthermore, if R_i=0, i=1, 2, … ....*m* and *m*=*n*, the censoring scheme reduces to no censoring, i.e., a case of a complete observed sample. Generally, an adaptive progressive type-II censoring scheme plays as a major factor in scale down the duration time of the experiment and the related cost as well as increasing the efficiency in statistical inference in any experimental design. For extensive knowledge on progressive censoring and real applications in reliability and quality see Balakrishnan and Cramer (2014).

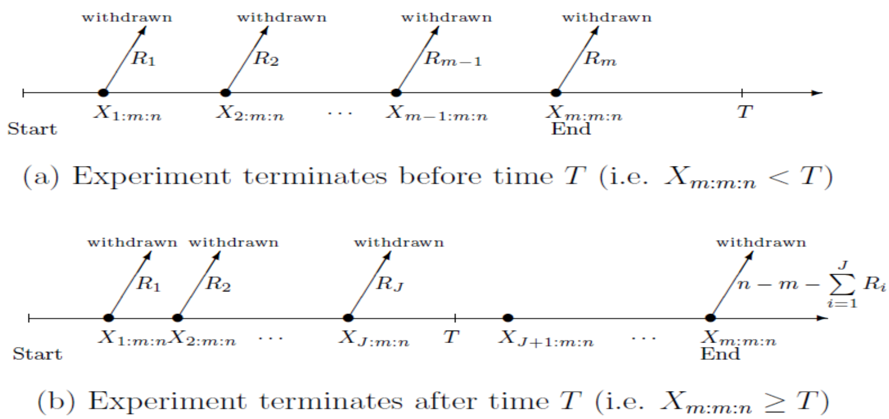

Figure 1: Illustration of an adaptive type-II progressive censoring scheme

1.1.1 Lifetime Analysis

A lifetime is defined as a positive random variable T, referring to the time elapsed until the occurrence of the event under consideration. Examples of such events are as follows: death, breakdown, entry into unemployment, or illness. Lifetime analysis is the study of the delay of the occurrence of the event under study.

1.1.2 Censored Data

It refers to data some of which are only known with a lower or upper bound and not a precise value.

1.1.3 Type I-Censored Data

Under a type-I censoring scheme, the test will be terminated upon a pre-determined time T is reached while the observed sample size m is not provided before the experiment (random).

1.1.4 Type-II Censored Data

Under a type -II censoring scheme, the test will be ended upon the pre-fixed number m is reached. Thus, the ideal test time of the real-life experiment is random.

1.1.5 Progressive Censoring Scheme

The natural concept of progressive censoring scheme is allowing to remove units at each observed failure time.

1.1.6 Progressive Type-I Censored Data

It is a general concept of type-I (time) censoring scheme and applied in limited area in real-life test due to the main purpose of censoring. In this censoring scheme the ideal test times are predetermined in advance and the number of observed sample m are determined until the experiment is terminated. Thus, the effective sample size m is considered as random variable as well as the progressive censoring scheme $(R_i, i = 1 \cdots\cdots, m)$.

1.1.7 Progressive Type-II Censored Data

It is a general concept of type-II (failure) censoring scheme due to lake of flexibility of removing survival units during the experiment. In this censoring scheme the effective sample size m is predetermined in advance and the progressive censoring scheme are provided previously, but the test duration time of the experiment is random. Suppose n independent and identical units are put in the test and at each observed failure time, a predetermined number of survival units is randomly extracted from the test and this manner will continue until the time of last failure observed based on $R_m = R_i - m\,, i = 1, \cdots \cdots m - 1$ removals survival units.

1.1.8 Type-II Progressive Hybrid Censored Data

It is a generalization of type-II progressive censoring scheme due to the main purpose of censoring which is a compromise between saving the total test time and observing many failure times during the experiment. In this censoring scheme, the ideal duration time as well as a progressive censoring scheme are previously determined, but the number of observed failure time m is random. Thus, the statistical inference in this censoring scheme will not gain a high efficiency due to the effective random sample size.

1.1.9 Adaptive Progressive Type-II Censored Data

It is a combination of type-I and progressive type-II censoring schemes which is useful in many real-life tests. In this censoring scheme, the observed sample size m, the removal units at each observed failure time, and the ideal total test time T are fixed in advance by the experimenter. In addition, this censoring scheme called an adaption process in case of modifying some of the values in the predetermined progressive censoring scheme based on the pre-determined time T.

1.2 Literature Review

This section focuses on reviewing works by previous researchers relevant to the study's problem. The primary goal is to offer an overview of the approaches developed so far in approximating the estimate of the parameters of log-logistic distribution under an adaptive progressive type-II censored sample. This helps us gain insight in our research while avoiding repetition.

1.2.1 Inference Based on the Log-Logistic Distribution

Various probability density functions have been proposed as models for lifetime data. The log-logistic distribution (named in economics the Fisk distribution) is the most widely used distribution in analysing lifetime's data. It is suitable substitute for use as a substitute to the Weibull distribution. It is a combination of the Gompertz and Gamma distributions with the mean and variance equal to one.

In 2005, it was noticed from the available writers on the subject distribution; a study considered distinguish between some estimators of the log-logistic model based on uncensored and censored samples. The authors in this study focused on both un-censored and censored sample in case of type-I (time) and type-II (failure) censoring schemes. They derived the maximum likelihood (ML) equations of the two estimators and solving theses equations simultaneously for the two unknown parameters. Thus, they found that the equating equations cannot be solved for the two unknown parameters explicitly and finding the roots need some numerical procedure like Newton-Raphson method. Thus, they suggested an alternative to the MLE. In their research, they used the least squares estimators by regressing certain estimators of the linearized distribution function on a function of the observations themselves, as a first technique to compare between the cases of data (complete and censored). The second technique applied under the type-II censoring scheme based on expanding certain terms in ML equations by using a Taylor series expansion to get a new system of ML equation whose solution exist in closed form. Moreover, they exhibited the estimated model parameters under a different situation. Based on these simulations, some of the estimated criteria are calculated such as bias, mean squared error, and the ratios of the mean square errors of the MLEs were also obtained. They conclude that the biases all very small and consequently the estimators were approximately unbiased. Further, the efficiency of the estimators under type-II (failure) censoring are like to the estimators under type-I (time) censoring (AL-Haj Ebrahem & Baklizi, 2005). In 2015, a study examined a statistical inference based on the log-logistic distribution with right censored data. The author provided the Bayesian estimation based on an informative gamma prior and derived the ML estimators. Additionally, the Bayesian methods performed based on two types of loss functions. Thus, the ratio of two integrals cannot be solved explicitly; therefore, the approximated Bayes estimators were estimated by using the approximation of Lindley technique. The medical life-data set considered as either randomly or non-informatively censored. In the first approach, the ML equations cannot be solved explicitly, and the roots of these simultaneous equations estimated by using an iterative (Newton-Raphson) procedure with a trivial value using a first order Taylor series. In the Bayesian approach, the author first used the squared error loss function technique to estimate the unknown parameters and noticed that the equations are not be solved explicitly. Thereafter, he considered the Lindley approximation procedure to compute the ratio of two integrals. Then, he provided the asymmetric loss function with Lindley's approach to approximate the estimate of unknown parameters. Finally, a simulation study was conducted based on three real-life data sets for analysing the proposed methods. They were concluded that all the estimators are probably sufficient to estimate the two unknown parameters. Additionally, maximum likelihood and Bayes estimation based on symmetric loss function according to a Monte Carlo simulation study provide approximately the same estimate for the scale parameter. Furthermore, the authors noticed that for the shape parameter, Bayes using squared error loss is estimated well than the maximum likelihood estimation. However, both estimation methods have the same value of estimate due to the standard errors which approximately tend to same value in case of large sample size. Additionally, the Bayes estimate for the shape parameter under the linear exponential loss function best estimate when the sample size arranges between small to moderate (Guure, 2015). In 2017, a modified of the ML equations of the log-logistic distribution situated on progressive type-II (failure) censoring scheme with binomial removals was studied. In this study, the modified ML equations were derived due to the non-closed form solutions. Thus, the solution of ML equations obtained by using a Newton-Raphson method. The author modified the ML equations for the two unknown parameters by linearizing some term using the Taylor series expansion. Moreover, the asymptotic matrix of the inverse of the observed Fisher information matrix has obtained to approximate the two-sided normal confidence interval for the parameters. Additionally, he considered the MLEs of the reliability and hazard rate by using the invariance properties of the MLEs. Moreover, the coverage probabilities of the asymptotic intervals for both parameters were derived for purpose of Monte Carlo simulation (Raykundaliya, 2017).

1.2.2 Inference Based on an Adaptive Progressive Type-II Censoring Scheme

Several researchers have worked on an adaptive progressive type-II censoring and different distributions under this censoring scheme have been considered by many authors.

In 2009, a study considered a statistical inference of exponential life times under adaptive type-II progressive censoring scheme. A censoring scheme which combine between type-I and type-II progressively censoring schemes, known an adaptive type-II progressively censoring scheme, was used; the authors provided an algorithm to generate an adaptive progressive type-II censored data from whatever continue distribution. The number of observed failure m, initial time T, and a progressive censoring scheme $(R_i, i = 1, \cdots\cdots, m)$ are needed to predetermine in advance. Additionally, the non-Bayesian estimator of the lifetime parameter was derived, and the estimate of the variance-covariance matrix was constructed. Thereafter, they provided various methods for build up a confidence interval for the parameter. Moreover, they introduced the computational formula for the expected duration time of the experiment and based on some properties of the exponential distribution, they obtained the conditional expectation of $x_{m:m:n}$for$j = 0,1, \cdots, m-1$. Furthermore, they examined the difference between two types of progressive censoring schemes – adaptive PT-II censored scheme and hybrid censoring scheme – proposed recently by Kundu and Joarder (2006) based on the efficiency of the MLE using Monte Carlo simulation. According to their results, the mean for point estimation based on Bayesian approach and the Bayes credible interval are commended when informative prior knowledge exist corresponding to the parameters; contrary, the

MLEs and Bayes interval with non-informative prior for interval estimation should be applied (Ng, Kundu, & Chan, 2009). In 2013, a research of generalized Pareto model based on adaptive progressive type-II censoring was introduced. According to this research, the authors used maximum likelihood, Bayes, and bootstrap estimation techniques. In Bayesian approach, the Bayes estimate based on non-informative prior of the unknown parameters under SELF are founded under a simulated sample from the intractable posterior density by using MCMC algorithm. The comparison between the Bayes estimators and the ML estimators was performed. Furthermore, a real-life example was illustrated using a real data to examine the suggested methods. Finally, the comparison between the different methods were performed using a Monte Carlo simulation study. According to this, the study concluded that the MLEs were nearly close to the Bayes estimators (Mahmoud, Soliman, Abd Ellah, & El-Sagheer, 2013). Additionally, in 2016, a study considered the exponentiated Weibull distribution to estimate based on an adaptive type-II progressive censoring scheme. This research focused on estimating the two unknown parameters of the introduced model, reliability, and hazard functions using Bayesian and frequentist approaches. The approximate confidence intervals and parametric bootstrap confidence intervals in case of small effective sample size were according to the MLEs. Moreover, the Bayes estimate for the unknown estimators under SELF by using posterior samples which was generated via MCMC algorithm. Thus, the corresponding approximate Bayes intervals for the unknown parameters were constructed. The authors also provided a real example to examine the proposed methods of estimation. A Monte Carlo simulation executed to investigate the proposed methods based on the estimated criteria. For this purpose, the authors generate 1000 censored values from the introduced distribution. Moreover, a simulation study was performed based on various values of n, different observed sample of m, and different progressive censoring schemes (CS). They concluded that the achievement of the frequentist estimators was like that of Bayes estimates under non-informative priors. Furthermore, the MSEs of the maximum likelihood estimators and Bayes estimators were higher for the censoring scheme $R_m = n - m, R_i = 0 \; for \; i \neq m$ than the censoring scheme $R_1 = n - m, R_i = 0, i \neq 1$ (AL Sobhi & Soliman, 2016). Moreover, in 2016, the generalized exponential distribution was provided to estimate the model parameters using the MLE and Bayesian approach as well as the survival and hazard functions. Additionally, a study was performed to construct the confidence intervals for these unknown quantities. The authors provided that; the introduced model will tend to the exponential distribution when the shape parameter ϴ=1. Based on the ML equations for the two unknown parameters, the observed Fisher information matrix was constructed. Thus, the inverse of this matrix is constructed. Consequently, the approximate confidence intervals were constructed for the two unknown parameters. By using the ML estimators for deriving the survival and hazard functions, the authors used the delta method to calculate the approximate confidence intervals for these functions. They created a linear approximation of that functions and then calculated the variance due to analytically reasons of variance estimation. Additionally, a study considered a Bayesian approach with informative and non-informative gamma prior for that quantities under SELF procedure to get the Bayesian estimators. Due to the complicity for solving the integrals analytically, they considered a numerical technique to approximate these integrals. In fact, the authors provided the MCMC flexible method as an alternative method for Bayesian estimation for parameters and calculating the Bayes intervals via the Metropolis-Hastings algorithm within Gibbs sampling. Moreover, a Monte Carlo study was performed to examine the different proposed methods of estimation under different choices of removal units with different choices of *n* and *m*. Finally, an illustrative real-life example was considered to examine the proposed methods of estimation. Based on the simulation study, the mean-squared error decreased as *n* and *m* increased. Furthermore, the Bayesian approach with informative gamma priors was considered as the perfect way for parameters estimation under all cases (Mohie El-Din M. M., Amein, Shafay, & Mohamed, 2016). Moreover, in 2017, the exponentiated exponential distribution was stayed for an adaptive progressive type-II censoring scheme. In this study, the quantities were estimated under the maximum likelihood and Bayesian estimation procedure. In Bayes approach the parameters were estimated under two types of loss functions. Additionally, the corresponding intervals based on both methods of parameters estimation were constructed. Moreover, the authors provided a comparison study between the Bayesian and MLE approaches using the estimated risk criterion via a simulation study. Finally, a study presented a real-life example to examine the proposed methods. The simulation study concluded that the progressive type-II censoring scheme was occurred when the ideal total test time approach to infinity ($\mathrm{T} \rightarrow \infty$) while the conventional type-II censoring scheme was obtained when the ideal total test time approach to zero ($\mathrm{T} \rightarrow 0$). In addition, the introduced censoring scheme was reduced to case of un-censored scheme when the ideal total test time equal to the effective sample size ($m = n$). Furthermore, this study considered that, the Bayes estimate based on linear exponential loss function was influenced by shape parameter value. However, the Bayesian approach based on squared error and linear exponential loss functions considered as the best method for estimating the unknown parameters rather than the frequentist approach. In addition, when the shape parameter tended to small value (approach to zero), the estimated risks criterion under Bayesian approach based on both loss functions as nearly close to each other in all situations. Further, the confidence interval approach to the nominal level under large values of sample size, but the Bayes intervals tend in most cases (Ateya & Mohammed, 2017).

1.3 Problem Statement

Over the past few years, the log-logistic distribution has extensively been used to analyse lifetime data, owing to its flexibility. This is a popular distributional model, that can be used as a substitutional model to the Weibull distribution (a non-monotonic hazard function) in lifetime or reliability experiments. Moreover, this life distribution has a characteristic property which is the distribution function can be formed explicitly in closed form in contrast to the log-normal distribution. This allows us to analyse many types of censoring data.

Some related works were done based on log-logistic distribution, for instance, by Guure (2015), the parameters of the log-logistic model were estimated based on right censored data by using Bayesian and classical estimation methods. In addition, in case of complete sample there is a study considered by Al-Shomrani et al. (2016) to estimate the unknown parameters by using numerical MCMC techniques in Bayesian approach.

There is bounded work that can be found under an adaptive progressive type-II censored scheme based on different lifetime distributions, for example, the most important research which was considered early in this filed were performed by Ng et al. (2009) to estimate the failure rate based on MLE method under an adaptive progressive type-II censored data. By Mahmoud et al. (2013), the estimation of the unknown parameters of generalized Pareto was also performed based on this type of censoring scheme by using MLE method, Bayesian estimation and parametric bootstrap method with constructing the corresponding intervals for these methods. Additionally, Mohie El-Din et al. (2016) discussed in their research the MLE and Bayesian approach to estimate the parameters of the generalized exponential distribution and other quantities under this type of censored data. Furthermore, AL Sobhi & Soliman (2016) studied in their research the exponentiated Weibull model to estimate the parameters and other quantities based on this type of censoring scheme by using ML and Bayesian methods. The latest study was considered by Ateya & Mohammed (2017) to estimate the exponentiated exponential model parameters by using the MLE and Bayesian methods based on this type of censoring scheme. However, it has been noticed in the available studies that no work has been done based on an adaptive progressive type-II censoring scheme in case of log-logistic distribution.

Therefore, due to this work limitation, this study will develop the ML and Bayesian estimation methods to estimate the parameters, survival function and the associated intervals under an adaptive progressive type-II censored data.

1.4 Significance of the Study

The results will make the log-logistic distribution applicable in cases where life testing experiments are faced with adaptive progressive type-II censored data. Additionally, the study will contribute and augment the usefulness of survival and reliability analysis.

1.5 Objective of the Study

The primary aim is to explore the maximum likelihood estimators (MLEs) for the log-logistic distribution parameters and for the survival function and calculate the approximate confidence intervals for both the parameters and the survival function. Additionally, the Bayesian method is to be used to approximately estimate the unknown parameters (μ, σ) and approximate the Bayes credible interval for the parameters and survival function based on an adaptive progressive type-II censored data. Furthermore, a Monte Carlo simulation is executed to consider their efficiency. To explain the proposed methods, a data set using a real-life example is analysed.

1.6 Specific Objectives

To achieve the main objective stated above, we have the following specific objectives:

1. Estimate the parameters of the log-logistic model and the survival function based on the maximum likelihood method and construct the corresponding confidence intervals.
2. Estimate the parameters, survival function, and the associated credible intervals based on Bayesian approach under SELF.
3. Compare between the proposed methods of estimation by using a Monte Carlo simulation.
4. Apply the estimation procedure under a life-time data.

1.7 The Scope of the Study

This study considered only the Bayesian and classical approach to estimate the parameters of the log-logistic distribution together with the survival function, constructing the corresponding approximate confidence and credible intervals for each quantity.

CHAPTER 2: MAXIMUM LIKELIHOOD AND BAYES ESTIMATION

2.1 Brief Overview of the Maximum Likelihood Inference

The MLE method is considered as suitable technique for driving estimators. In the background, the primary concept of the MLE is to maximize the probability (likelihood) of the observed sample. Additionally, this method yields with good efficiency in statistical inference. Moreover, the method can be utilized to various distributional model and various types of data. The Newton-Raphson method is considered as numerical techniques to find these estimators in case of non-closed form solution of ML equations. The Newton-Raphson method (also called Newton's method) is one of the flexible numerical techiques for solving a non-linear equations and used for optimization. It considered fixed point iteration scheme for approximating the roots and requires that the function be continuous and differentiable by using the concept of low-order terms of Taylor series. In general, this method is need only one initial true value for each parameter and it is different than other methods such as bisection and false methods which require two initial true values for each parameter. To explain the method of Newton-Rapson method of solving a Nonlinear equation, let $f(a)$ be a function and let a a root of the equation $f(a) = 0$. Suppose x_0 is an initial guess of the root, let x_1 is a next estimate value of the same parameter and we continue to produce a good estimate until we approach to close estimate of the root. This procedure in the Newton Raphson method is called an iterative procedure or an iterative root-finding procedure. In specific, this method is efficient when the initial value is close to the true value of the parameter and let $a = x_0 + h$ denoted that the initial value with the error of estimate the parameter. Thus, let $h = a - x_0$, where h measures how much the x_0

far from the true value of the parameter. By using the a few low-order terms of the Taylor series expansion when the value of $(a - x_0)$ is quite close to zero, we can write the function $f(a)$ in $(a - x_0)$ as follows:

$$f(a) = f\,(x_0 + h) \approx f(x_0) + h\,f'(x_0) = 0$$

and

let $h \approx \frac{-f(x_0)}{f'(x_0)}$, then

$a = x_0 + h \approx x_0 - \frac{f(x_0)}{f'(x_0)}$ and let x_1 is a first estimate of a which denoted as $x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$ and let $x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$ be a next estimate of the true value and consequently we continue to performed this procedure until convergence is obtained. Thus, the Newton Rapson formula can be written as follows:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

2.1.1 Log-Logistic Distribution

The log-logistic distribution has been proposed by Bain (1974) via a transformation of a well-known logistic variate and its characteristic have been handled by Ragab and Green (1984). Assume that the random variable T of the unit follows the log-logistic distribution with two unknown parameters μ and σ. Moreover, the probability function (pdf), cumulative distribution function (cdf), survival function and hazard rate function of the log-logistic distribution are respectively as follows:

$$f(t) = \frac{\pi}{\sigma t\sqrt{3}} \frac{e^{\frac{-\pi(\ln t - \mu)}{\sigma\sqrt{3}}}}{\left[1 + e^{\frac{-\pi(\ln t - \mu)}{\sigma\sqrt{3}}}\right]^2} \quad , \quad 0 < t < \infty \;, \qquad -\infty < \mu < \infty \,, \qquad 0 < \sigma < \infty$$

$$F(t) = \frac{1}{1 + e^{\frac{-\pi(lnt - \mu)}{\sigma\sqrt{3}}}} \qquad , \qquad 0 < t < \infty$$

$$R(t) = 1 - \frac{1}{1 + e^{\frac{-\pi(lnt - \mu)}{\sigma\sqrt{3}}}} \qquad , \qquad 0 < t < \infty$$

$$h(t) = \frac{f(t)}{R(t)} = \frac{\pi}{\sigma t\sqrt{3}} \frac{1}{\left[1 + e^{\frac{-\pi(lnt - \mu)}{\sigma\sqrt{3}}}\right]} \qquad , \qquad 0 < t < \infty$$

Where μ and σ are the location and scale parameters respectively. Figures 2 and 3 below show the plots of the log-logistic probability density function and the survival function for some distinct values of the parameters μ and σ respectively.

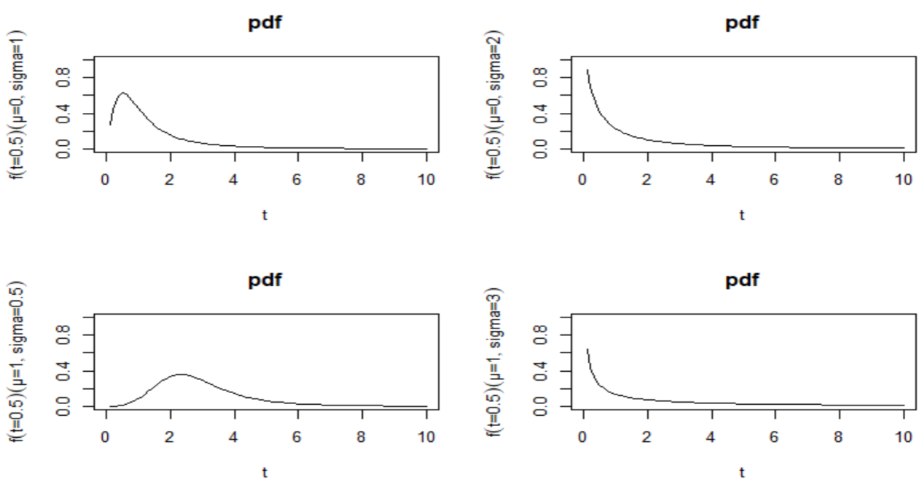

Figure 2*:* Plots of the log-logistic pdf for some different parameter's values

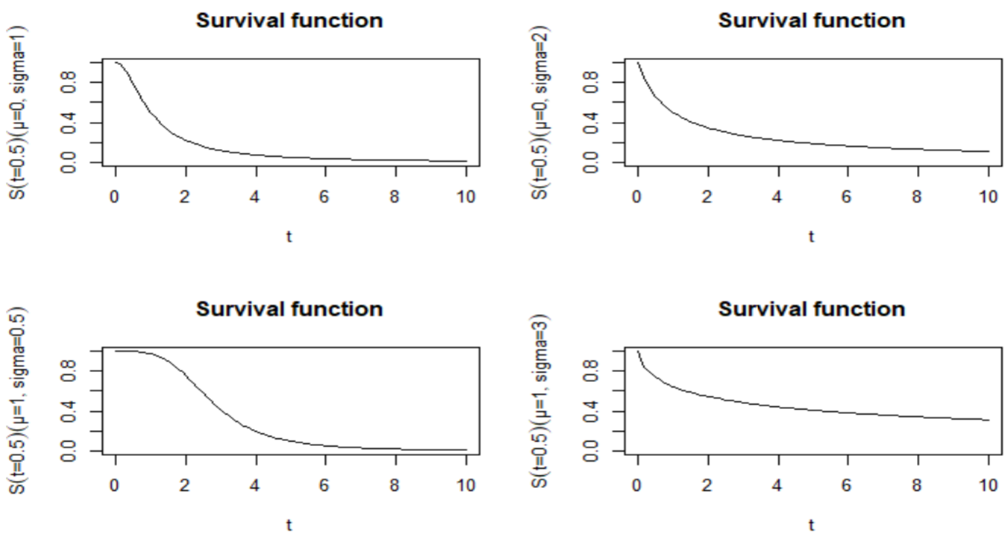

Figure 3: Plots of the log-logistic survival function for some different parameter's values

2.1.2 Likelihood Based on an Adaptive PT-II Censored Data

Consider n units on a real life experiment are from a lifetime distribution with cdf $\boldsymbol{F}(\boldsymbol{x};\boldsymbol{\Theta})$, pdf $\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\Theta})$ by using the above-mentioned assumptions, the conditional likelihood function of the vector of parameters Θ given the vector of observations $\boldsymbol{t}$ with progressive censoring scheme $\boldsymbol{R}$=$(\boldsymbol{R_1},\cdots\cdots,\boldsymbol{R_m})$ under an observed sample. This was defined on its introduction by Ng et al. (2009) as follows:

$$L(\Theta;\underline{t}) = d_J\left(\prod_{i=1}^{m} f(t_i\,;\theta)\right)\left(\prod_{i=1}^{J}(1-F(t_i\,;\theta))^{R_i}\right)\left(1-F(t_m;\theta)\right)^{n-m-\sum_{i=1}^{J}Ri}$$

$$0 < t_1 < t_2 < \cdots < t_m < \infty$$

where

$$d_J = \prod_{i=1}^{m}\left[n-i+1-\sum_{k=1}^{max\{i-1,J\}} R_k\right]$$

and the associated conditional likelihood function of log-logistic distribution is as follows:

$$L(\theta;\boldsymbol{t})=d_J\left(\prod_{i=1}^{m}\frac{\pi}{\sigma t_i\sqrt{3}}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left[1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right]^2}\right)\left(\prod_{i=1}^{J}(1-\frac{1}{\left[1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right]})^{R_i}\right)\left(1-\frac{1}{\left[1+e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}\right]}\right)^{n-m-\sum_{i=1}^{J}R_i}$$

$$L(\theta;\boldsymbol{t})=d_J\left(\prod_{i=1}^{m}\frac{\pi}{\sigma t_i\sqrt{3}}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left[1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right]^2}\right)\left(\prod_{i=1}^{J}\left(\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left[1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right]}\right)^{R_i}\right)\left(\frac{e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}}{\left[1+e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}\right]}\right)^{n-m-\sum_{i=1}^{J}R_i}$$

2.1.3 Maximum Likelihood Estimation

The MLEs for the two unknown parameters μ and σ have very interesting asymptotic properties, such as Lehmann (1998) derived. Based on this method, the estimators are consistent, asymptotically unbiased, and best asymptotically normal. Let $\boldsymbol{t_{1:m:n}},\cdots\cdots,\boldsymbol{t_{m:m:n}}$ denote an adaptive progressive type-II censored sample, with $(\boldsymbol{R_1},\cdots\cdots,\boldsymbol{R_m})$ representing the progressive censoring scheme. The maximum likelihood equation based on this observed data can be obtain by taking ln for the likelihood function as follows:

$$ln\,L(\theta;\boldsymbol{t})=\textbf{Constant}+\sum_{i=1}^{m}ln\frac{\pi}{\sigma t_i\sqrt{3}}-\frac{\pi[\sum_{i=1}^{m}(lnt_i-\mu)]}{\sigma\sqrt{3}}-2\sum_{i=1}^{m}ln\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)$$
$$-\frac{\pi}{\sigma\sqrt{3}}\sum_{i=1}^{J}R_i(lnt_i-\mu)-\sum_{i=1}^{J}R_i\,ln\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)$$
$$+\left(n-m-\sum_{i=1}^{J}R_i\right)\left[\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}-ln\left(1+e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}\right)\right] \qquad (1)$$

The first partial derivatives of equation (1) with respect to μ and σ are respectively as follows:

$$\frac{\partial\, ln\, L(\theta;\boldsymbol{t})}{\partial\mu}=\frac{m\pi}{\sigma\sqrt{3}}-\frac{2\pi}{\sigma\sqrt{3}}\sum_{i=1}^{m}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left(1+e^{\frac{-(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)}+\frac{\pi}{\sigma\sqrt{3}}\sum_{i=1}^{J}R_i\left[1-\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)}\right]$$

$$+\left(n-m-\sum_{i=1}^{J}R_i\right)\left(\frac{\pi}{\sigma\sqrt{3}}-\frac{e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}\frac{\pi}{\sigma\sqrt{3}}}{(1+e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}}\right)=0$$

$$\frac{\partial\, ln\, L(\theta;\boldsymbol{t})}{\partial\mu}=\frac{m\pi}{\sigma\sqrt{3}}-\frac{2\pi}{\sigma\sqrt{3}}\sum_{i=1}^{m}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)}+\frac{\pi}{\sigma\sqrt{3}}\sum_{i=1}^{J}R_i\left[\frac{1}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)}\right]$$

$$+\frac{\pi}{\sigma\sqrt{3}}\left(n-m-\sum_{i=1}^{J}R_i\right)\left(\frac{1}{(1+e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}}\right)=0 \qquad (2)$$

$$\frac{\partial\, ln\, L(\theta;\boldsymbol{t})}{\partial\sigma}=\sum_{i=1}^{m}\frac{\sqrt{3}\,\sigma t_i}{\pi}\frac{-\pi}{\sqrt{3}\sigma^2 t_i}+\frac{\pi\sum_{i=1}^{m}lnt_i-m\pi\mu}{\sigma^2\sqrt{3}}-2\sum_{i=1}^{m}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\frac{\pi(lnt_i-\mu)}{\sigma^2\sqrt{3}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)}+\frac{\pi\sum_{i=1}^{J}R_i(lnt_i-\mu)}{\sigma^2\sqrt{3}}$$

$$-\sum_{i=1}^{J}R_i\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\frac{\pi(lnt_i-\mu)}{\sqrt{3}\sigma^2}}{\left(1+e^{\frac{-\pi(lnt_i-\mu}{\sigma\sqrt{3}}}\right)}+\left(n-m-\sum_{i=1}^{J}R_i\right)\left[\frac{\pi(lnt_m-\mu)}{\sigma^2\sqrt{3}}-\frac{e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}\frac{\pi(lnt_m-\mu)}{\sqrt{3}\sigma^2}}{\left(1+e^{\frac{-\pi(lnt_m-\mu}{\sigma\sqrt{3}}}\right)}\right]$$

$$=0$$

$$\frac{\partial\, ln\, L(\theta;\boldsymbol{t})}{\partial\sigma}=-\sigma^{-m}+\frac{\pi\sum_{i=1}^{m}(lnt_i-\mu)}{\sigma^2\sqrt{3}}\left(1-2\sum_{i=1}^{m}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)}\right)$$

$$+\frac{\pi\sum_{i=1}^{J}R_i(lnt_i-\mu)}{\sigma^2\sqrt{3}}\left(1-\sum_{i=1}^{m}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)}\right)$$

$$+\frac{\pi(lnt_m-\mu)}{\sigma^2\sqrt{3}}\left(n-m-\sum_{i=1}^{J}R_i\right)\left[1-\frac{e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}}{\left(1+e^{\frac{-\pi(lnt_m-\mu}{\sigma\sqrt{3}}}\right)}\right]=0 \qquad (3)$$

Equations (2) & (3) cannot be solved for μ, σ explicitly. So, these equations required to solving numerically. The ML estimator for the reliability function by using the invariance property of ML estimator is as follows:

$$\widehat{s(t)} = 1 - \frac{1}{1+e^{\frac{-\pi(lnt_i - \hat{\mu})}{\hat{\sigma}\sqrt{3}}}} \quad , \quad t > 0$$

2.2 Asymptotic Confidence Intervals for μ and σ

The Fisher's changing were discussed by Aldrich (1997) and the consequently observed Fisher information matrix of the parameters μ and σ for large *n*, is given as follows:

$$I(\hat{\mu},\hat{\sigma}) = \begin{pmatrix} -\frac{\partial^2 \ln L(\theta,\underline{t})}{\partial\mu^2} & -\frac{\partial^2 \ln L(\theta,\underline{t})}{\partial\mu\partial\sigma} \\ -\frac{\partial^2 \ln L(\theta,\underline{t})}{\partial\sigma\partial\mu} & -\frac{\partial^2 \ln L(\theta,\underline{t})}{\partial\sigma^2} \end{pmatrix}_{\mu=\hat{\mu},\sigma=\hat{\sigma}}$$

where

$$\frac{\partial^2 \ln L(\theta,\underline{t})}{\partial\mu^2} = \frac{-2\pi}{\sigma\sqrt{3}} \sum_{i=1}^{m} \frac{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right) e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}} \frac{\pi}{\sigma\sqrt{3}} - e^{\frac{-2\pi(lnt_i-\mu)}{\sigma\sqrt{3}}} \frac{\pi}{\sigma\sqrt{3}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)^2}$$

$$-\frac{\pi}{\sigma\sqrt{3}} \sum_{i=1}^{J} R_i \frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}} \frac{\pi}{\sigma\sqrt{3}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)^2} - \frac{\pi}{\sigma\sqrt{3}} \frac{\left(n-m-\sum_{i=1}^{J} R_i\right) e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}} \frac{\pi}{\sigma\sqrt{3}}}{\left(1+e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}\right)^2} \quad (4)$$

$$\frac{\partial^2 \ln L(\theta,\underline{t})}{\partial\sigma^2} = m\sigma^{-m-1} - \pi\sum_{i=1}^{m}\frac{(lnt_i-\mu)}{\sigma^4\sqrt{3}}\left(1-2\sum_{i=1}^{m}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)}\right) + \pi\sum_{i=1}^{m}\frac{(lnt_i-\mu)}{\sigma^2\sqrt{3}}\frac{-2e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\frac{\pi(lnt_i-\mu)}{\sigma^2\sqrt{3}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)^2} -$$

$$\pi\sum_{i=1}^{J}\frac{R_i(lnt_i-\mu)}{\sigma^2\sqrt{3}}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\frac{\pi(lnt_i-\mu)}{\sigma^2\sqrt{3}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)^2} - \left(1-\sum_{i=1}^{m}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)}\right)\frac{\pi\sum_{i=1}^{J}R_i(lnt_i-\mu)}{\sigma^4\sqrt{3}} - \frac{\pi(lnt_m-\mu)}{\sigma^2\sqrt{3}}\Big(n-$$

$$m-\sum_{i=1}^{J}R_i\Big)\frac{e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}\frac{\pi(lnt_m-\mu)}{\sigma^2\sqrt{3}}}{\left(1+e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}\right)^2} - \left(1-\frac{e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}}{\left(1+e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}\right)}\right)\frac{\pi(lnt_m-\mu)\left(n-m-\sum_{i=1}^{J}R_i\right)}{\sigma^4\sqrt{3}} \tag{5}$$

$$\frac{\partial^2 \ln L(\theta,\underline{t})}{\partial\mu\partial\sigma} = \frac{-m\pi}{\sigma^2\sqrt{3}} - \frac{2\pi}{\sigma\sqrt{3}}\sum_{i=1}^{m}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\frac{\pi(lnt_i-\mu)}{\sigma^2\sqrt{3}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)^2} + \frac{2\pi}{\sigma^2\sqrt{3}}\sum_{i=1}^{m}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)} - \frac{\pi}{\sigma\sqrt{3}}\sum_{i=1}^{m}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\frac{\pi(lnt_i-\mu)}{\sigma^2\sqrt{3}}}{\left(1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right)^2} -$$

$$\frac{\pi}{\sigma^2\sqrt{3}}\sum_{i=1}^{J}R_i\left(\frac{1}{1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}\right) - \frac{\pi}{\sigma\sqrt{3}}\left(n-m-\sum_{i=1}^{J}R_i\right)\frac{e^{\frac{-\pi((lnt_m-\mu))}{\sigma\sqrt{3}}}\frac{\pi(lnt_m-\mu)}{\sigma^2\sqrt{3}}}{\left(1+e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}\right)^2} -$$

$$\frac{\pi}{\sigma^2\sqrt{3}}\frac{\left(n-m-\sum_{i=1}^{J}R_i\right)}{\left(1+e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}\right)} \tag{6}$$

Regrettably, it is difficult to find the expected Fisher information analytically (it does not exist). Therefore, by using the concept of large sample theory and the variance-covariance matrix, which is the inverse of the observed Fisher information matrix $I^{-1}(\hat{\mu},\hat{\sigma})$, the approximate 100(1-$\alpha$) % normal confidence intervals for the parameters μ and σ are given respectively as follows:

$$\left(\hat{\mu} - z_{\alpha/2}\sqrt{\widehat{V(\hat{\mu})}}, \hat{\mu} + z_{\alpha/2}\sqrt{\widehat{V(\hat{\mu})}}\right)$$

and

$$\left(\hat{\sigma} - z\alpha_{/2}\sqrt{\widehat{V(\hat{\sigma})}}, \hat{\sigma} + z\alpha_{/2}\sqrt{\widehat{V(\hat{\sigma})}}\right)$$

Where V($\hat{\mu}$) and V($\hat{\sigma}$) are the estimate variance of $\hat{\mu}\ and\ \hat{\sigma}$, given by the main diagonal elements of $I^{-1}(\hat{\mu}, \hat{\sigma})$, and $z\alpha_{/2}$ represent the right tail probability $\alpha/_2$ for standard distribution.

2.3 Approximate Confidence Interval for Survival Function

The Delta method (Greene, 2010), is applied to evaluate the approximate confidence intervals for the survival functions (S(t)). This is a natural way for calculating the confidence interval for functions of the MLEs, in which these functions are intractable to analytically calculate the variance. Then, by creating linear approximations of this survival function and then calculating the variance of linear approximation as follows:

$$G = \left(\frac{\partial S(t)}{\partial \mu} \quad \frac{\partial S(t)}{\partial \sigma}\right)$$

where,

$$\frac{\partial S(t)}{\partial \mu} = \frac{\left(1+e^{\frac{-\pi(lnt-\mu}{\sigma\sqrt{3}}}\right)e^{\frac{-\pi(lnt-\mu)}{\sigma\sqrt{3}}}\frac{\pi}{\sigma\sqrt{3}} - e^{\frac{-\pi(lnt-\mu)}{\sigma\sqrt{3}}}e^{\frac{-\pi(lnt-\mu)}{\sigma\sqrt{3}}}\frac{\pi}{\sigma\sqrt{3}}}{\left(1+e^{\frac{-\pi(nt-\mu)}{\sigma\sqrt{3}}}\right)^2}$$

$$\frac{\partial S(t)}{\partial \mu} = \frac{\frac{\pi}{\sigma\sqrt{3}}e^{\frac{-\pi(lnt-\mu)}{\sigma\sqrt{3}}}}{\left(1+e^{\frac{-\pi(lnt-\mu)}{\sigma\sqrt{3}}}\right)^2} \qquad (7)$$

$$\frac{\partial S(t)}{\partial \sigma} = \frac{\left(1+e^{\frac{-\pi(lnt-\mu)}{\sigma\sqrt{3}}}\right)e^{\frac{-\pi(lnt-\mu)}{\sigma\sqrt{3}}}\frac{\pi(lnt-\mu)}{\sigma^2\sqrt{3}} - e^{\frac{-\pi(lnt-\mu)}{\sigma\sqrt{3}}}e^{\frac{-\pi(lnt-\mu)}{\sigma\sqrt{3}}}\frac{\pi(lnt-\mu)}{\sigma^2\sqrt{3}}}{\left(1+e^{\frac{-\pi(lnt-\mu)}{\sigma\sqrt{3}}}\right)^2}$$

$$\frac{\partial S(t)}{\partial \sigma} = \frac{e^{\frac{-\pi(lnt-\mu)}{\sigma\sqrt{3}}}\frac{\pi(lnt-\mu)}{\sigma^2\sqrt{3}}}{\left(1+e^{\frac{-\pi(lnt-\mu)}{\sigma\sqrt{3}}}\right)^2} \qquad (8)$$

and the approximate estimate of the variance of $\widehat{S(t)}$ is given by the following:

$\widehat{Var}(\widehat{S(t)}) \approx [G^t I^{-1}(\mu,\sigma)G]_{(\hat{\mu},\hat{\sigma})}$

Then the approximate confidence interval for S(t) is as follows:

$$\left(\widehat{S(t)} - z\alpha_{/2}\sqrt{\widehat{Var}(\widehat{S(t)})}\,, \widehat{S(t)} + z\alpha_{/2}\sqrt{\widehat{Var}(\widehat{S(t)})}\right)$$

2.4 Brief Overview of the Bayesian Approach

In Bayesian analysis, the parameters of interest are considered as some random variables and follow some prior distributions. The informative prior distribution is considered when we have previous information about the model parameters. When no previous information for the parameters accessible, it is more suitable to consider the non-informative prior for the Bayesian analysis. Additionally, Jeffreys' prior, is defined as a non-informative prior since $p(\theta) \propto [I(\theta)]^{\frac{1}{2}}$, where $I(\theta) = -E_\theta\left[\frac{d^2 log(p(y\backslash\theta))}{d\theta^2}\right]$. Furthermore, a conjugate prior distribution $p(\theta)$ for a given sampling distribution is one where the prior density as well as the likelihood density function have the same functional structure. Moreover, the comfortable analytical results for conjugate prior make it is useful to use, but it is not desirable to handling for this relaxed. A prior distribution captures all the known information about the parameters before collecting data and it is updated when this sample information is collected. The Bayes' Rule is the behind of this updating which is called posterior distribution. Before introducing the Bayesian estimation procedure, we began to state Bayes' Theorem first, the idea behind Bayesian approach. Bayes' Theorem is also known as Bayes' rule and was introduced by the Thomas Bayes (Casella & Berger , 2002). Suppose A & B are two events in a sample space and the probability of these events denoted as $P(A)$ & $P(B)$, respectively; then, the conditional probability formulas can be written as $p(A\backslash B)=\frac{p(A\cap B)}{p(B)}$ and $p(B\backslash A)=\frac{p(A\cap B)}{p(A)}$, where $p(A)\neq 0\;\&\;p(B)\neq 0.$ The conditional probability formulas are helpful in deriving Bayes' rule when substituting $p(A\cap B)=p(B\backslash A)p(A)$ in the conditional probability formula as $p(A\backslash B)=\frac{p(B\backslash A)p(A)}{p(B)}$. The Bayes' rule can be reported as $p(A_i\backslash B)=\frac{P(B\backslash A_i)P(A_i)}{\sum_{j=1}^{\infty}P(B\backslash A_j)p(A_j)}$ when A_i represent a set of mutually exclusive events. Thus, the Bayes' rule is considered as posterior = $\frac{likelihood\ x\ prior}{marginal\ likelihood}$.

2.4.1 Bayesian Estimation

Bayesian method provides substitutional procedure to estimate the parameters, principally when previous knowledge about μ and σ is available. Hence, the Bayes estimates for the unknown parameters (μ , σ) and the survival function, that represents the probability for an observation to stay alive at the end of duration test time. Thus, the corresponding credible intervals for a parameters and survival function based on an adaptive progressive type-II censored data under SELF are constructed. Because the posterior distributions result from more complicated models and cannot be written in closed form, a MCMC algorithm has been provided in this chapter to simulate samples from posterior distribution. Therefore, the non-informative priors for both parameters μ and σ are considered as $\pi_1(\mu)\propto 1$ and $\pi_2(\sigma\backslash\mu)\propto\frac{1}{\sigma}$. When multiplying $\pi_1(\mu)$ by $\pi_2(\sigma\backslash\mu)$, the corresponding prior density of μ and σ are considered as follows $\pi(\mu,\sigma)\propto\pi_1(\mu)\pi_2(\sigma\backslash\mu)$, clearly $\pi(\mu,\sigma)\propto\frac{1}{\sigma}$. Additionally, an informative conjugate logistic prior for μ is considered and because the scale parameter σ of the log-logistic distribution is nonnegative, the gamma prior for this parameter is taken, which is not necessarily the conjugate prior; this prior distribution is more applicable in the sense that it is more flexible and popular (Guure, 2015). The prior density is considered for μ as $\pi_1(\mu) \propto \frac{\exp\left(\frac{-(\mu-\theta)}{\beta}\right)}{\left[1+\exp\left(\frac{-(\mu-\theta)}{\beta}\right)\right]^2}$ with known hyper parameters $(\theta = 0, \beta = 1)$ and for parameter σ as $\pi_2(\sigma\backslash\mu) \propto \sigma^{a-1}\exp(\frac{-\sigma}{b})$, where a and b represent the hyper parameters that are assumed known (a=1, $b = 1$). The known hyper parameters are considered in such a way that the expected value by using these known values equal the initial true value of the corresponding parameters which known as prior mean of the suggested prior distribution. Consequently, the joint prior density of μ and σ are considered as follows:

$$\pi(\mu,\sigma) \propto \frac{\sigma^{a-1}\exp(\frac{-\sigma}{b}-\frac{(\mu-\theta)}{\beta})}{\left[1+\exp(\frac{-(\mu-\theta)}{\beta})\right]^2}.$$

Subsequently, the general form of the posterior density is proportionally to the likelihood function times the prior density function as the following:

$$p(\mu,\sigma\backslash\boldsymbol{t}) \propto \prod_{i=1}^{m}[\ \ likelihood\ \ ]\{prior\}$$

and the corresponding joint posterior conditional density function with non-informative priors is

$$p(\mu,\sigma)\propto\left(\prod_{i=1}^{m}\frac{\pi}{\sigma t_i\sqrt{3}}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left[1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right]^2}\right)\left(\prod_{i=1}^{J}\left(\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left[1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right]}\right)^{R_i}\right)\left(\frac{e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}}{\left[1+e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}\right]}\right)^{n-m-\sum_{i=1}^{J}R_i}\times\frac{1}{\sigma}$$

while the corresponding joint posterior conditional density function with an informative prior is

$$p(\mu,\sigma)\propto\left(\prod_{i=1}^{m}\frac{\pi}{\sigma t_i\sqrt{3}}\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left[1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right]^2}\right)\left(\prod_{i=1}^{J}\left(\frac{e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}}{\left[1+e^{\frac{-\pi(lnt_i-\mu)}{\sigma\sqrt{3}}}\right]}\right)^{R_i}\right)\left(\frac{e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}}{\left[1+e^{\frac{-\pi(lnt_m-\mu)}{\sigma\sqrt{3}}}\right]}\right)^{n-m-\sum_{i=1}^{J}R_i}\times$$

$$\frac{\sigma^{a-1}\exp(\frac{-\sigma}{b}-\frac{(\mu-\theta)}{\beta})}{\left[1+\exp(\frac{-(\mu-\theta)}{\beta})\right]^2}$$

Hence, the Bayes estimates of any function of μ and σ such as h (μ, σ), based on squared error loss function is as follows:

$$\widehat{h(\mu,\sigma)}=\{E_{\mu,\sigma\backslash t}(h(\mu,\sigma))\}$$

$$=\frac{\int_{\mu=-\infty}^{\infty}\int_{\sigma=0}^{\infty}h(\mu,\sigma)X\,L(\underline{t};\mu,\sigma)X\pi(\mu,\sigma)d\sigma d\mu}{\int_{\mu=-\infty}^{\infty}\int_{\sigma=0}^{\infty}L(\underline{t};\mu,\sigma)X\pi(\mu,\sigma)d\sigma d\mu} \qquad (9)$$

All the Bayesian estimators under the SEL function considered as ratio of two integrals and it is not possible to compute equation (9) in closed form. MCMC is one of the best numerical approximation for estimating these unknown parameters and provides flexibility way for extracting posterior samples from their respective posterior distributions. Therefore, the MCMC algorithm needs to perform as a numerical method or we require some approximation techniques (Lindley’s or the Tierney-Kadane method) to approximate the above integrals for evaluating the approximate Bayes estimates of the parameters and the reliability function. Using MCMC by choosing Gibbs sampling or the Metropolis procedure has been used to simulate random values from the intractable posterior density function, approximate the Bayesian estimator, and build the associated credible interval for the parameters and reliability function. See Hamada et al. (2008) for more information about the MCMC techniques and Bayesian reliability examples. Here, the Metropolis-Hastings algorithm has been considered to simulate samples from the full conditional posterior distribution and the proposal proceeds by proposing a joint move on (μ, σ). Thus, the approximate Bayes estimate for the parameters and survival function are computed, in turn, to construct the associated credible intervals. The Metropolis-Hasting algorithm is illustrated below; which provides a flexible way for obtaining random values from a target distribution with the logistic candidate for parameter μ and the inverse-gamma candidate for parameter σ; we assume that Θ is a 2-dimensional, real-valued parameter vector as follows:

1) Initialize $j = 0$, $\mu^{(j)} = 0, \sigma^{(j)} = 1$

2) Set $j = 1$

3) Draw μ^* from a logistic ($\mu^{(j-1)}, s^2$) candidate distribution.

4) Draw $\sigma^*$from an inverse gamma($\left(\sigma^{(j-1)}\right)^2 + 2$), $\left(\sigma^{(j-1)}\right)^2 + 1$) candidate distribution.

5) Compute the acceptance probability $r = min\left(1, \frac{p(\theta^*/data)}{p(\theta^{(j-1)}/data)} \frac{f(\theta^{(j-1)}/\theta^*)}{f(\theta^*/\theta^{(j-1)})}\right)$

6) Draw u from a uniform (0,1) density.

7) If $u \leq r$, $set\ \theta^{(j)} = \theta^*$.Otherwise, set $\theta^{(j)} = \theta^{(j-1)}$

8) Increment j and repeat steps 3 to 7 for N=11000 times.

9) Approximate Bayes estimates of μ and σ using MCMC samples based on the SEL function as $\hat{\mu}_{BS}=\frac{1}{N-M}\sum_{i=M+1}^{N} \mu^{(i)}$ and $\hat{\sigma}_{BS}=\frac{1}{N-M}\sum_{i=M+1}^{N} \sigma^{(i)}$,where M is burn-in.

10) Substitute $\mu^{(i)}$ and $\sigma^{(i)}$ in to equation $S(t) = 1 - \frac{1}{1+e^{\frac{-\pi(lnt-\mu)}{\sigma\sqrt{3}}}}$, $0 < t < \infty$ to compute $S^{(1)}(t), S^{(2)}(t), \cdots\cdots, S^{(N)}(t)$.

11) An approximate Bayesian estimates of the $S(t)$, based on the SEL function, can be

found as $\widehat{S}_{Bs}(t) = \frac{\sum_{i=M+1}^{N} S^{(i)}(t)}{N-M}$.

12) Compute the credible intervals of μ and σ, order $\mu_{M+1}, \mu_{M+2}, \cdots\cdots, \mu_N$ and $\sigma_{M+1}, \sigma_{M+2}, \cdots\cdots, \sigma_N$ as $\mu_1, \mu_2, \cdots\cdots, \mu_{N-M}$ and $\sigma_1, \sigma_2, \cdots\cdots, \sigma_{N-M}$.Then, the $100(1-\alpha)$ % symmetric credible intervals of μ and σ constructed as $\left(\mu_{((N-M)(\alpha/2))}, \mu_{((N-M)(1-\alpha/2))}\right)$ and $\left(\sigma_{((N-M)(\alpha/2))}(t), \sigma_{((N-M)(1-\alpha/2))}\right)$.

13) Compute the credible intervals of $S(t)$, order $S_{M+1}(t), S_{M+2}(t), \cdots\cdots, S_N(t)$ as $S_1(t) < S_2(t) < \cdots\cdots < S_{N-M}(t)$. Then, the $100(1-\alpha)$ % symmetric credible intervals of S(t) as $\left(S_{((N-M)(\alpha/2))}(t), S_{((N-M)(1-\alpha/2))}(t)\right)$.

CHAPTER 3: SIMULATION STUDY, RESULTS AND COMPARISONS

3.1 Simulation Study

A Monte Carlo simulation study was executed to examine the efficiency of different estimators, comparing the suggested methods of estimates of μ, σ, and S(t). This simulation study was implemented using the R program. R is an interactive software application designed specifically to perform calculations, manipulate data, and produce graphical displays of data and results. It is a free software project, part on an international effort to share software without charge (Linder, Seefeld, & Ed, 2015). Generate an adaptive progressive type-II censored sample with pre-determined number of n and m and the progressive censoring schemes with given values of the ideal censoring time T from the log-logistics distribution is described below using the procedure described by Balakrishnan and Sandhu (1995) and by Ng et al. (2009) . For illustration, the algorithm to generate an adaptive progressively type-II censored sample from any continuous life time distribution is considered as follows:

1. Define the values of n, m, μ, σ, T and $(R_1, R_2, \cdots, R_m)$.
2. Simulate m random variables from uniform (0,1) as $W_1, W_2 \cdots \cdots W_m$.
3. Set $V_i = W_i^{1/(i+R_m+R_{m-1}+\cdots+Rm-i+1)}$for $i = 1,2,\cdots m$.
4. Set $Ui = 1 - VmVm - 1 \ldots.Vm - i + 1$ for $i = 1,2,\ldots,m$ .Then $U1, U2, \ldots.Um$, is the m progressive type-II observed sample from uniform distribution.
5. Set Xi = F^{-1}(Ui, θ) for $i = 1,2,\ldots,m$,where $F^{-1}(.,\theta)$ represent the quantile function of the log-logistic distribution. Thus, $X1, X2, \ldots, Xm$, is the needed

progressive type-II observed sample from the specified distribution F(.) by using inverse transformation method.

6. Identify the value of J, where $x_{J:m:n} < T < x_{J+1:m:n}$, and discard the sample $x_{j+2:m:n},\ldots\ldots,x_{m:m:n}$.

   Simulate the first $m-J-1$ order random values from a truncated distribution considered as $\frac{f(x)}{[1-F(x_{j+1:m:n})]}$ with sample size $\left(n-\sum_{i=1}^{j} R_i - J - 1\right)$ as $x_{j+2:m:n}, x_{j+3:m:n}, \cdots, x_{m:m:n}$.

A Monte Carlo simulation is performed under different numbers of total sample size *n*, observed sample size *m*, and different cases of progressive censoring schemes (CS) for each choice of *m* and *n* which tabulated in Table 1.

Table 1: The Different Progressive Censoring Scheme R with Different Choices of n and m

Scheme	n	m	CS
1	50	30	(0*29,20)
2			(0*10,2*10,0*10)
3			(20,0*29)
4	50	40	(0*39,10)
5			(0*15,1*10,0*15)
6			(10,0*39)
7	70	40	(0*39,30)
8			(0*10,2*15,0*15)
9			(30,0*39)
10	70	50	(0*49,20)
11			(0*20,2*10,0*20)
12			(20,0*49)
13	90	50	(0*49,40)
14			(0*15,2*20,0*15)
15			(40,0*49)

16	90	60	(0*59,30)
17			(0*20,2*15,0*25)
18			(30,0*59)

Hence, a simulation study is executed with respect to two distinct values of ideal total test time T as $(1, 1.8)$ where these censoring times are calculated in the form of $F^{-1}(0.5)$ =1 and $F^{-1}(0.75) = 1.8$ by using the quantile function of the log-logistic distribution as shown in Appendix A. To generate the data, we suppose that the initial true values of the parameters (μ , σ) are (0,1) respectively, for the survival function with different values of t (0.5,1,2) as (0.7784,0.5,0.2215) corresponding respectively to each value of t. For prior information, the non-informative priors for both parameters are considered as the flat prior for parameter μ and the $\frac{1}{\sigma}$ Jeffrey prior for parameter σ, which is given as MCMC0. Additionally, an informative prior (MCMC1) is considered as μ following the logistic distribution with known hyper parameters (ϴ=0, β=1) and σ following the gamma distribution with known hyper parameters (a=1, b=1). To find the Bayesian estimate and the 95% Bayes interval for the unknown parameters, we simulate 11000 MCMC values from the target distribution using the Metropolis-Hastings algorithm. Generally, successive samples (values) from the target distribution are correlated; however, this autocorrelation approach to disappear as the MCMC algorithm run for long time. In detail; For each choice of n and m with each choice of progressive censoring scheme we replicate the process 2000 times to generate an adaptive progressive type-II censored sample.

3.1.1 Convergence Diagnostic

To use the MCMC samples for inference we must ensure that the Markov chains have reached stationarity. The convergence diagnostic helps us in determining when this convergence is achieved and guide us to determine the number of values needs to remove from the beginning of the chain. However, most diagnostics are designed to verify a necessary yet insufficient condition for convergence. Moreover, the initial values of the chains are not simulated from the target distribution, so these simulated values of the parameters do not represent the required distribution. Thus, after the MCMC run for long time, the effect of correlated values will die and the distribution after the burn-in period tends the posterior distribution (target distribution). The simplest path to estimate the approximate number of values to be removed is to draw the time series plots of MCMC draws for each parameter against the iteration number. A coda package (Bayesian package) was used, an R package providing several functions for the plotting and analysis of generated posterior samples. For these functions to work, we need to transform the MCMC samples to an object of class “mcmc” to allow us the available functions in the package coda. As seen from Figures 2 and 3, the two rows correspond to the μ and σ parameters, respectively; thus, there are two plots for each parameter. The left plot in both figures indicate that a burn-in period is needed; the chains are mixing well because the mean and variance are relatively constant – this means that the chains have reached stationarity when MCMC runs for much longer. The word ‘mixing’ connotes how well the algorithm reach to the parameter’s distribution. The burn-in is considered as M=1000 to generate independent random values and to try to enter a region with high posterior probability; this burn-in does not represent samples from the posterior distribution. The right plots for both figures are called the marginal density plots for the posterior parameters. Thus, the marginal plots represent the distribution of values of the parameters in the MCMC chain. It is clear that, the marginal density plot for parameters μ and σ are unsmoothed; this is because the bandwidth function (bwf) in the plot function in the coda package is omitted and, consequently, the bandwidth is calculated by default as:

$$bw = 0.9n^{\frac{-1}{5}}.\min\left(sd(x), \frac{IQR(x)}{1.34}\right).$$

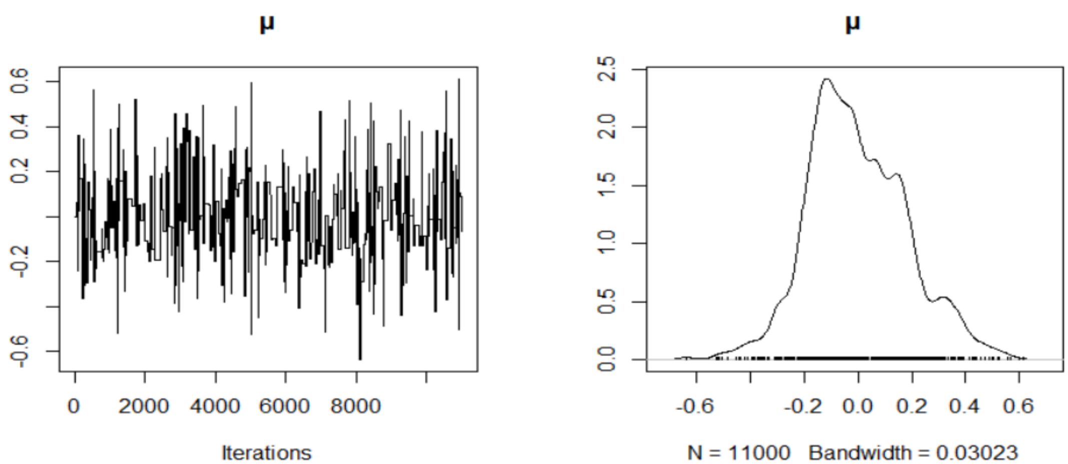

Figure 4*:* Trace plot and probability density plot of the location parameter

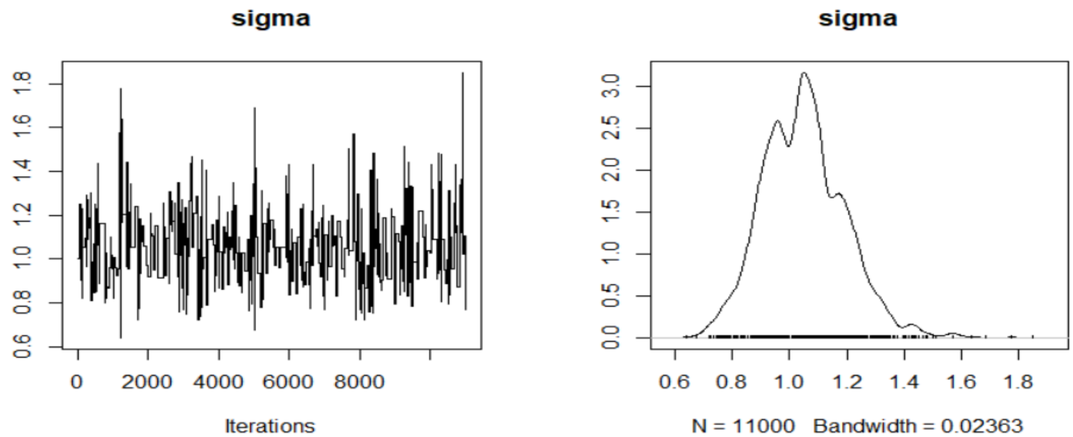

Figure 5: Trace plot and probability density plot of the scale parameter

Generally, the choice of the bandwidth value controls the smoothness of plotting the marginal density function. One of the techniques to smooth this density can be done by using R program. Thus, the correct bandwidth approximately obtained by giving the density function to an approximately independent subsample of the data. To know where the autocorrelation is approximately not significant, we draw the autocorrelation plots for each parameter, as shown in Figures 4 and 5. Moreover, the 'autocorr function' in the package coda provides the output for the lags and the corresponding autocorrelations which noted as lag 50 gives the nearly uncorrelated data (value near to zero). Thus, we sub-sample the output at spacing 50; then, we draw again the marginal density plot for both parameters with the new bandwidth calculated using the R code, as shown below in Figures 6 and 7; this approach is provided in detailing by Geyer (2012).

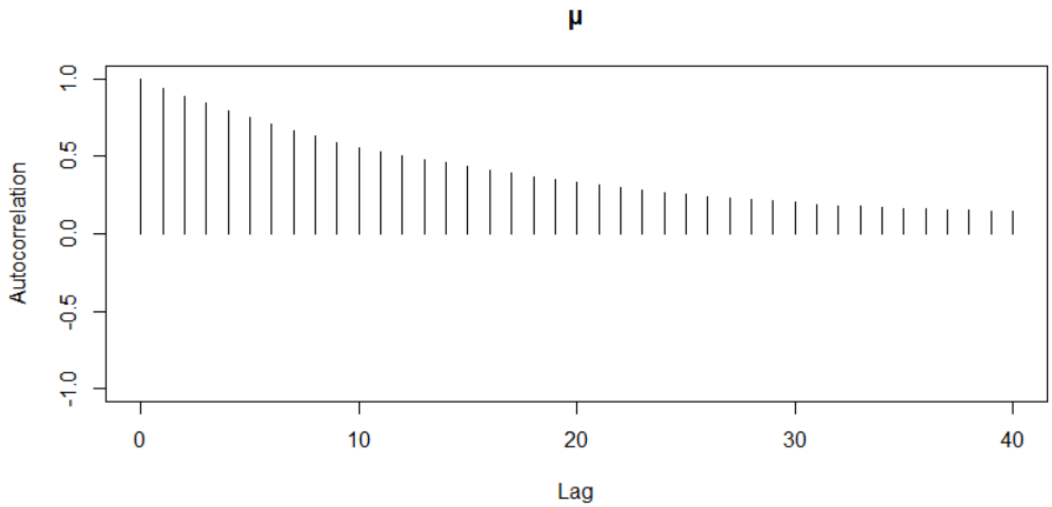

Figure 6: Autocorrelation plot of MCMC output for the location parameter

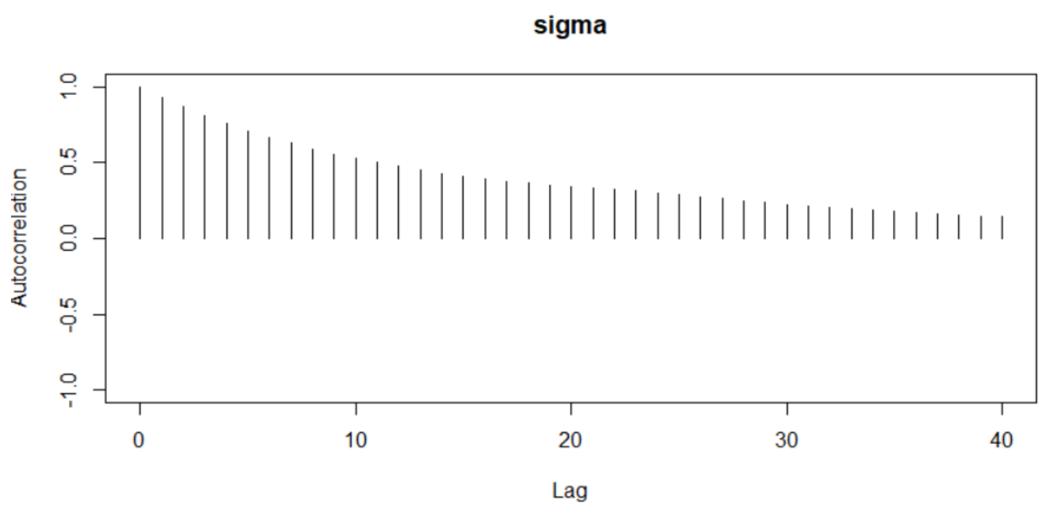

Figure 7*:* Autocorrelation plot of MCMC output for the scale parameter

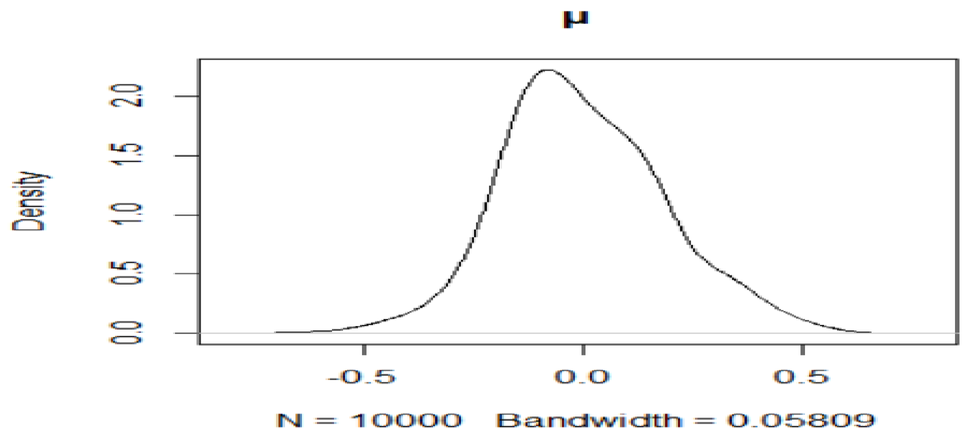

Figure 8: Smooth probability density plot for the location parameter with burn-in=1000

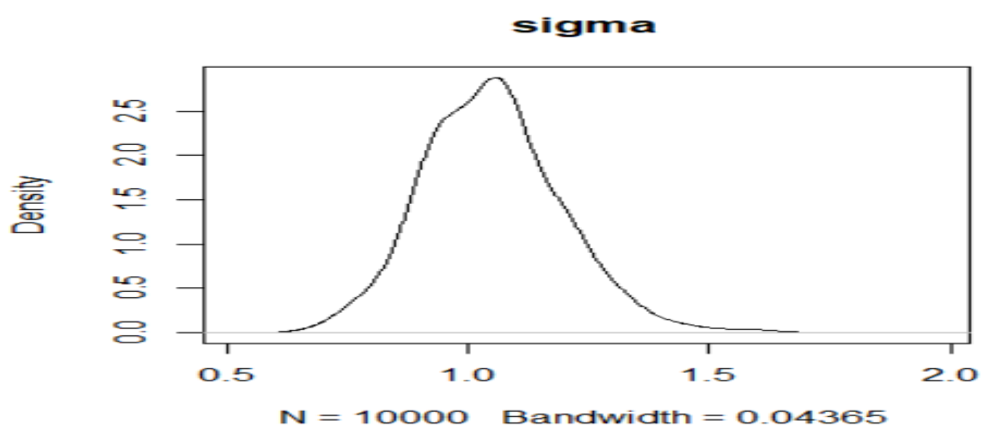

Figure 9: Smooth probability density plot for the scale parameter with burn-in=1000

Additionally, Figures 8 and 9 show the histograms of the values in the MCMC chain for parameters μ and σ, respectively after iteration 1000. In addition, if the chain does not mix well, we can increase the discard values, increase the simulated sample or changing the choice of the proposal function. For more information about convergence diagnostics with coda package in R, see Hartig (2011). Appendix B presents the R code for generating an adaptive progressive type-II censored sample, convergence diagnostic, and a Monte Carlo simulation study for one choice of *n*, *m*, and T, based on one case of progressive censoring scheme for different methods of estimation.

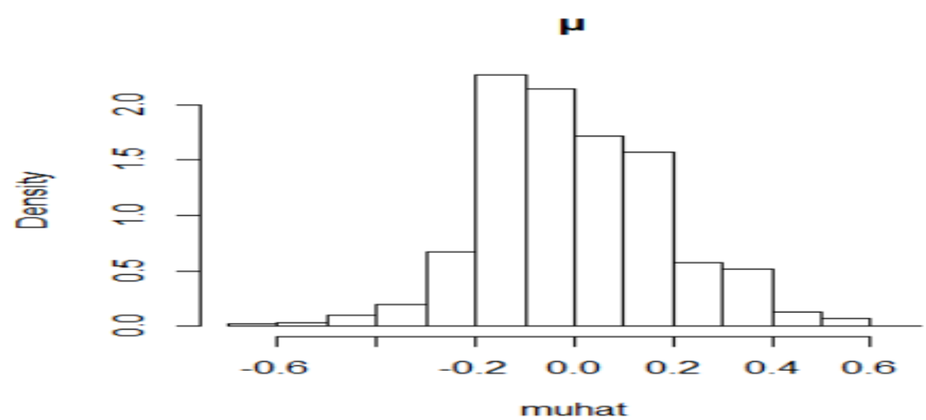

Figure 10*:* Histogram of the random draws of the location parameter with burn-in=1000

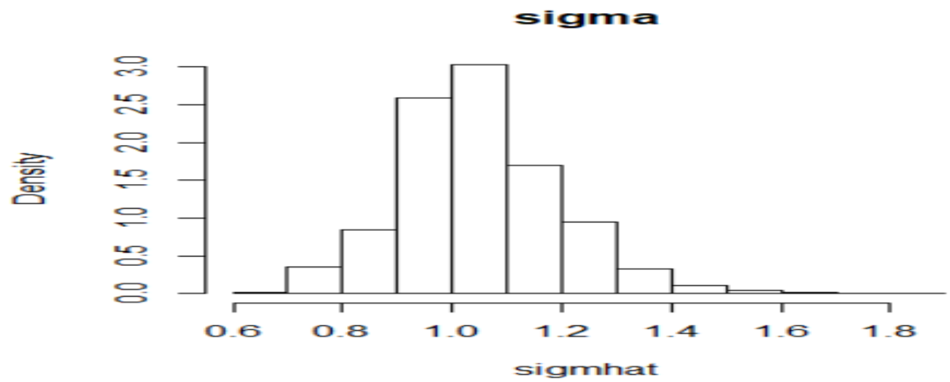

Figure 11*:* Histogram of the random draws of the scale parameter with burn-in=1000

3.2 Results and Comparisons

For both methods, maximum likelihood and Bayesian estimation, the process was replicated 2000 times. For each generated sample, we construct a 95% confidence interval, determined whether the initial true value fall inside the interval, and compute the width of the constructed interval. The coverage probability was measured as the count of intervals contained the initial true value divided by 2000, while the estimated average length of the confidence and credible intervals were evaluated as the average of lengths for all intervals over 2000. The mean, bias, and MSE of μ, σ, and S(t) parameters for each method are tabulated in Tables 2–5 for the two values of T. From the simulation study, obviously the MLEs are near to that of the Bayes estimators (MCMC0) in case of non-informative priors. Thus, it is preferable to use the MLEs in return of the Bayes estimation when no reliable information is available, since the Bayes estimators are more intractable. Furthermore, considered from the tables below, the bias was being very small, not presented in all cases of the progressive censoring schemes with all methods of estimations; consequently, the estimators are approximately unbiased. It is obvious that the mean squared errors (MSEs) are reduced in all methods when the values of *n* and *m* are increased. Moreover, where *n* is fixed and m increases, the MSEs are decreased in all situations with all methods. Furthermore, the MSEs in non-classical method with informative prior (MCMC1) have the smallest values in most cases placed on different situations. Besides, the MSEs via different progressive censoring schemes were compered, it is considerable that the MSE values are small and near to each other in each different set of *n* and *m*, but the progressive censoring scheme $R = (0, \dots \dots \dots, n - m)$ is most efficient for all choices and provides approximately the smallest MSE for all estimators. It is considerable that, there is no big difference between the two distinct values of ideal total test time based on the estimated criteria in all cases.

Table 2: Mean, Bias and MSE of Location & Scale Parameters Based on T=1

$(\boldsymbol{n}, \boldsymbol{m})$	CS	MLE		MCMC0		MCMC1	
(50,30)	$(0^{29}, 20)$	**μ**	**σ**	**μ**	**σ**	**μ**	**σ**
Mean		-0.0094	0.9789	0.0005	1.0145	0.0018	1.0120
Bias		-0.0094	-0.0210	0.0005	0.0145	0.0018	0.0120
MSE		0.0199	0.0261	0.0207	0.0260	0.0212	0.0243
	$(0^{10}, 2^{10}, 0^{10})$						
Mean		-0.0105	0.9815	0.0078	1.0109	-0.0013	1.0069
Bias		-0.0105	-0.0184	0.0078	0.0109	-0.0013	0.0069
MSE		0.0240	0.0230	0.0244	0.0239	0.0239	0.0219
	$(20, 0^{29})$						
Mean		-0.0038	0.9873	0.0006	1.0092	0.0035	1.0102
Bias		-0.0038	-0.0126	0.0006	0.0092	0.0035	0.0102
MSE		0.0291	0.0220	0.0312	0.0221	0.0309	0.0221

(n, m)	CS	MLE		MCMC0		MCMC1	
(50,40)	$(0^{39}, 10)$	**μ**	**σ**	**μ**	**σ**	**μ**	**σ**
Mean		0.0015	0.9800	0.0005	1.0110	0.0015	1.0079
Bias		0.0015	-0.0199	0.0005	0.0110	0.0015	0.0079
MSE		0.0187	0.0179	0.0181	0.0176	0.0173	0.0171
	$(0^{15}, 1^{10}, 0^{15})$						
Mean		-0.0022	0.9813	-0.0010	1.0089	-0.0052	1.0057
Bias		-0.0022	-0.0186	-0.0010	0.0089	-0.0052	0.0057
MSE		0.0181	0.0176	0.0213	0.0181	0.0198	0.0163
	$(10, 0^{39})$						
Mean		0.0028	0.9866	0.0050	1.0132	-0.0008	1.0032
Bias		0.0028	-0.0133	0.0050	0.0132	-0.0008	0.0032
MSE		0.0228	0.0169	0.0242	0.0182	0.0227	0.0165
(70,40)	$(0^{39}, 30)$						
Mean		-0.0098	0.9761	0.0024	1.0109	0.0015	1.0088
Bias		-0.0098	-0.0238	0.0024	0.0109	0.0015	0.0088
MSE		0.0148	0.0181	0.0155	0.0200	0.0143	0.0190
	$(0^{10}, 2^{15}, 0^{15})$						
Mean		-0.0080	0.9856	0.0018	1.0073	-0.0016	1.0036
Bias		-0.0080	-0.0143	0.0018	0.0073	-0.0016	0.0036
MSE		0.0181	0.0163	0.0183	0.0167	0.0183	0.0151
	$(30, 0^{39})$						
Mean		-0.0055	0.9833	0.0026	1.0135	0.0025	1.0065
Bias		-0.0055	-0.0166	0.0026	0.0135	0.0025	0.0065
MSE		0.0224	0.0160	0.0229	0.0163	0.0223	0.0163
(70,50)	$(0^{49}, 20)$						
Mean		-0.0053	0.9844	0.0052	1.0069	-0.0004	1.0046
Bias		-0.0053	-0.0155	0.0052	0.0069	-0.0004	0.0046
MSE		0.0125	0.0136	0.0136	0.0141	0.0139	0.0142
	$(0^{20}, 2^{10}, 0^{20})$						
Mean		-0.00005	0.9880	0.0006	1.0062	0.0008	1.0048
Bias		-0.00005	-0.0119	0.0006	0.0062	0.0008	0.0048
MSE		0.0144	0.0143	0.0156	0.0146	0.0146	0.0130
	$(20, 0^{49})$						
Mean		-0.0011	0.9939	0.0005	1.0050	-0.0009	1.0042
Bias		-0.0011	-0.0060	0.0005	0.0050	-0.0009	0.0042
MSE		0.0186	0.0138	0.0185	0.0135	0.0176	0.0136
(90,50)	$(0^{49}, 40)$						
Mean		-0.0083	0.9829	0.00008	1.0032	0.0036	1.0106
Bias		-0.0083	-0.0170	0.00008	0.0032	0.0036	0.0106
MSE		0.0116	0.0152	0.0116	0.0155	0.0116	0.0159
	$(0^{15}, 2^{20}, 0^{15})$						
Mean		-0.0040	0.9869	-0.0018	1.0048	0.0009	1.0019
Bias		-0.0040	-0.0130	-0.0018	0.0048	0.0009	0.0019
MSE		0.0142	0.0131	0.0142	0.0143	0.0140	0.0132
	$(40, 0^{49})$						
Mean		0.0026	0.9973	-0.0027	1.0069	-0.0046	1.0032
Bias		0.0026	-0.0026	-0.0027	0.0069	-0.0046	0.0032
MSE		0.0183	0.0134	0.0180	0.0132	0.0183	0.0126

(90,60)	$(0^{59}, 30)$						
Mean		-0.0048	0.9873	0.0042	0.9983	-0.0009	1.0029
Bias		-0.0048	-0.0126	0.0042	0.0016	-0.0009	0.0029
MSE		0.0110	0.0119	0.0107	0.0114	0.0114	0.0119
	$(0^{20}, 2^{15}, 0^{25})$						
Mean		-0.0064	0.9918	0.0020	1.0039	-0.0005	0.9991
Bias		-0.0064	-0.0081	0.0020	0.0039	-0.0005	-0.0008
MSE		0.0121	0.0109	0.0123	0.0114	0.0122	0.0113
	$(30, 0^{59})$						
Mean		0.0018	0.9923	0.0027	1.0018	0.0004	1.0062
Bias		0.0018	-0.0076	0.0027	0.0018	0.0004	0.0062
MSE		0.0156	0.0116	0.0153	0.0108	0.0151	0.0115

Table 3: Mean, Bias and MSE of S(t) Based on T=1

$(\boldsymbol{n}, \boldsymbol{m})$	**CS**	**MLE**			**MCMC0**			**MCMC1**		
$(\mathbf{50}, \mathbf{30})$	$(0^{29}, 20)$	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**
Mean		0.7800	0.4937	0.2133	0.7750	0.4958	0.2234	0.7756	0.4970	0.2237
Bias		0.0015	−0.0062	−0.0081	−0.0034	−0.0041	0.0018	-0.0027	-0.0029	0.0022
MSE		0.0027	0.0044	0.0036	0.0024	0.0043	0.0035	0.0025	0.0044	0.0034
	$(0^{10}, 2^{10}, 0^{10})$									
Mean		0.7779	0.4926	0.2146	0.7762	0.4984	0.2262	0.7740	0.4945	0.2230
Bias		−0.0005	−0.0073	−0.0069	−0.0021	−0.0015	0.0047	-0.0043	-0.0054	0.0014
MSE		0.0027	0.0052	0.0040	0.0026	0.0049	0.0039	0.0026	0.0050	0.0037
	$(20, 0^{29})$									
Mean		0.7782	0.4974	0.2190	0.7728	0.4989	0.2269	0.7736	0.5004	0.2280
Bias		−0.0001	-0.0025	−0.0024	−0.0055	−0.0010	0.0053	-0.0048	0.0004	0.0064
MSE		0.0036	0.0060	0.0040	0.0037	0.0061	0.0040	0.0037	0.0060	0.0040
(50,40)	$(0^{39}, 10)$									
Mean		0.7825	0.4998	0.2181	0.7753	0.4997	0.2247	0.7762	0.4998	0.2243
Bias		0.0040	-0.0001	−0.0034	−0.0030	−0.0002	0.0032	-0.0022	-0.0001	0.0028
MSE		0.0024	0.0040	0.0029	0.0024	0.0036	0.0025	0.0022	0.0035	0.0025
	$(0^{15}, 1^{10}, 0^{15})$									
Mean		0.7811	0.4983	0.2171	0.7742	0.4980	0.2243	0.7737	0.4962	0.2223
Bias		0.0026	−0.0016	−0.0044	−0.0041	−0.0019	0.0028	-0.0047	-0.0037	0.0008
MSE		0.0024	0.0038	0.0027	0.0026	0.0043	0.0029	0.0024	0.0040	0.0027
	$(10, 0^{39})$									
Mean		0.7808	0.5008	0.2207	0.7746	0.5021	0.2283	0.7749	0.4993	0.2243
Bias		0.0024	0.0008	−0.0007	−0.0037	0.0021	0.0067	-0.0034	-0.0006	0.0028
MSE		0.0029	0.0047	0.0031	0.0030	0.0048	0.0031	0.0028	0.0045	0.0029
(70,40)	$(0^{39}, 30)$									
Mean		0.7803	0.4939	0.2130	0.7766	0.4975	0.2233	0.7768	0.4972	0.2227
Bias		0.0018	−0.0060	−0.0085	−0.0018	−0.0024	0.0017	-0.0015	-0.0027	0.0011
MSE		0.0018	0.0033	0.0027	0.0018	0.0032	0.0027	0.0017	0.0030	0.0025
	$(0^{10}, 2^{15}, 0^{15})$									
Mean		0.7779	0.4943	0.2162	0.7754	0.4966	0.2239	0.7749	0.4954	0.2223
Bias		−0.0004	-0.0056	−0.0052	−0.0030	−0.0033	0.0023	-0.0034	-0.0045	0.0008
MSE		0.0020	0.0039	0.0030	0.0019	0.0037	0.0029	0.0019	0.0037	0.0028

$(\boldsymbol{n},\boldsymbol{m})$	**CS**	**MLE**			**MCMC0**			**MCMC1**		
	$(30,0^{39})$	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**
Mean		0.7788	0.4969	0.2175	0.7735	0.5002	0.2276	0.7751	0.5004	0.2261
Bias		0.0003	−0.0030	−0.0040	−0.0049	0.0002	0.0060	-0.0032	0.0048	0.0045
MSE		0.0028	0.0047	0.0030	0.0027	0.0045	0.0030	0.0027	0.0044	0.0029
(70,50)	$(0^{49},20)$									
Mean		0.7799	0.4970	0.2165	0.7780	0.5015	0.2247	0.7766	0.4987	0.2225
Bias		0.0014	-0.0029	−0.0049	−0.0003	0.0015	0.0032	-0.0017	-0.0012	0.0009
MSE		0.0017	0.0026	0.0020	0.0018	0.0027	0.0019	0.0018	0.0028	0.0020
	$(0^{20},2^{10},0^{20})$									
Mean		0.7804	0.4990	0.2190	0.7759	0.4981	0.2235	0.7763	0.4985	0.2234
Bias		0.0020	−0.0009	−0.0024	−0.0025	−0.0018	0.0019	-0.0021	-0.0014	0.0018
MSE		0.0018	0.0030	0.0023	0.0018	0.0032	0.0024	0.0017	0.0030	0.0022
	$(20,0^{49})$									
Mean		0.7782	0.4991	0.2208	0.7753	0.4998	0.2247	0.7753	0.4993	0.2239
Bias		−0.0001	-0.0008	−0.0007	−0.0030	−0.0001	0.0031	-0.0030	-0.0006	0.0024
MSE		0.0023	0.0038	0.0025	0.0023	0.0037	0.0024	0.0022	0.0035	0.0023
(90,50)	$(0^{49},40)$									
Mean		0.7794	0.4944	0.2147	0.7776	0.4970	0.2211	0.7772	0.4987	0.2238
Bias		0.0009	−0.0055	−0.0067	−0.0007	−0.0029	-0.0004	-0.0012	-0.0013	0.0022
MSE		0.0013	0.0025	0.0023	0.0014	0.0024	0.0021	0.0014	0.0024	0.0021
	$(0^{15},2^{20},0^{15})$									
Mean		0.7791	0.4963	0.2176	0.7753	0.4954	0.2218	0.7769	0.4972	0.2222
Bias		0.0007	−0.0036	−0.0039	−0.0030	−0.0045	0.0003	-0.0015	-0.0027	0.0007
MSE		0.0015	0.0030	0.0024	0.0014	0.0029	0.0024	0.0015	0.0029	0.0023
	$(40,0^{49})$									
Mean		0.7787	0.5009	0.2228	0.7739	0.4979	0.2240	0.7740	0.4976	0.2229
Bias		0.00027	0.0009	0.0012	−0.0045	−0.0020	0.0025	-0.0043	-0.0023	0.0013
MSE		0.0022	0.0038	0.0024	0.0021	0.0036	0.0024	0.0023	0.0037	0.0022
(90,60)	$(0^{59},30)$									
Mean		0.7793	0.4967	0.2172	0.7797	0.5005	0.2223	0.7770	0.4982	0.2217
Bias		0.0009	−0.0032	−0.0042	0.0012	0.0005	0.0008	-0.0014	-0.0017	0.0002
MSE		0.0013	0.0023	0.0019	0.0013	0.0022	0.0017	0.0014	0.0023	0.0018
	$(0^{20},2^{15},0^{25})$									
Mean		0.7775	0.4959	0.2179	0.7770	0.4990	0.2233	0.7772	0.4976	0.2215
Bias		−0.0008	−0.0040	−0.0035	−0.0013	−0.0009	0.0018	-0.0011	-0.0023	0.0002
MSE		0.0013	0.0025	0.0019	0.0014	0.0025	0.0018	0.0013	0.0025	0.0019
	$(30,0^{59})$									
Mean		0.7797	0.5004	0.2212	0.7771	0.5008	0.2244	0.7756	0.5000	0.2245
Bias		0.0012	0.0004	−0.0002	−0.0013	0.0008	0.0028	-0.0028	0.0001	0.0030
MSE		0.0019	0.0032	0.0021	0.0018	0.0030	0.0020	0.0019	0.0030	0.0019

Table 4: Mean, Bias and MSE of Location & Scale Parameters Based on T=1.8

(n,m)	CS	MLE		MCMC0		MCMC1	
(50,30)	$(0^{29},20)$	μ	σ	μ	σ	μ	σ
Mean		-0.0150	0.9751	0.0056	1.0138	0.0038	1.0138
Bias		-0.0150	-0.0248	0.0056	0.0138	0.0038	0.0138
MSE		0.0198	0.0245	0.0217	0.0259	0.0205	0.0244
	$(0^{10},2^{10},0^{10})$						
Mean		-0.0116	0.9772	0.0004	1.0071	0.0040	1.0041
Bias		-0.0116	-0.0227	0.0004	0.0071	0.0040	0.0041
MSE		0.0242	0.0227	0.0252	0.0234	0.0247	0.0219
	$(20,0^{29})$						
Mean		-0.0088	0.9904	0.0061	1.0082	0.0044	1.0107
Bias		-0.0088	-0.0095	0.0061	0.0082	0.0044	0.0107
MSE		0.0297	0.0207	0.0300	0.0222	0.0306	0.0195
(50,40)	$(0^{39},10)$						
Mean		-0.0018	0.9802	-0.0022	1.0061	-0.0024	1.0081
Bias		-0.0018	-0.0197	-0.0022	0.0061	-0.0024	0.0081
MSE		0.0185	0.0174	0.0193	0.0180	0.0184	0.0171
	$(0^{15},1^{10},0^{15})$						
Mean		0.0011	0.9867	0.0055	1.0034	0.0020	1.0064
Bias		0.0011	-0.0132	0.0055	0.0034	0.0020	0.0064
MSE		0.0206	0.0182	0.0212	0.0169	0.0198	0.0175
	$(10,0^{39})$						
Mean		-0.0032	0.9873	0.0032	1.0077	-0.0068	1.0029
Bias		-0.0032	-0.0126	0.0032	0.0077	-0.0068	0.0029
MSE		0.0220	0.0169	0.0239	0.0176	0.0241	0.0160
(70,40)	$(0^{39},30)$						
Mean		-0.0094	0.9823	-0.0021	1.0036	-0.0042	1.0011
Bias		-0.0094	-0.0176	-0.0021	0.0036	-0.0042	0.0011
MSE		0.0150	0.0184	0.0151	0.0185	0.0153	0.0188
	$(0^{10},2^{15},0^{15})$						
Mean		-0.0063	0.9838	0.0026	1.0094	0.0050	1.0070
Bias		-0.0063	-0.0161	0.0026	0.0093	0.0050	0.0070
MSE		0.0187	0.0166	0.0193	0.0163	0.0186	0.0169
	$(30,0^{39})$						
Mean		-0.0016	0.9858	-0.0052	1.0054	0.0037	1.0024
Bias		-0.0016	-0.0141	-0.0052	0.0054	0.0037	0.0024
MSE		0.0228	0.0153	0.0225	0.0159	0.0215	0.0153
(70,50)	$(0^{49},20)$						
Mean		-0.0015	0.9850	0.0036	1.0053	-0.0001	1.0096
Bias		-0.0015	-0.0149	0.0036	0.0053	-0.0001	0.0096
MSE		0.0134	0.0139	0.0136	0.0140	0.0137	0.0139
	$(0^{20},2^{10},0^{20})$						
Mean		-0.0043	0.9903	0.0006	1.0106	0.0037	1.0033
Bias		-0.0043	-0.0096	0.0006	0.0106	0.0037	0.0033
MSE		0.0145	0.0141	0.0150	0.0144	0.0175	0.0133

(n, m)	CS	MLE		MCMC0		MCMC1	
	$(20, 0^{49})$	**μ**	**σ**	**μ**	**σ**	**μ**	**σ**
Mean		-0.0016	0.9855	0.0041	1.0033	-0.0004	1.0030
Bias		-0.0016	-0.0144	0.0041	0.0033	-0.0004	0.0030
MSE		0.0176	0.0135	0.0184	0.0127	0.0178	0.0131
(90,50)	$(0^{49}, 40)$						
Mean		-0.0073	0.9827	0.0048	1.0054	0.0036	1.0048
Bias		-0.0073	-0.0172	0.0048	0.0054	0.0036	0.0048
MSE		0.0117	0.0154	0.0127	0.0155	0.0118	0.0148
	$(0^{15}, 2^{20}, 0^{15})$						
Mean		0.0016	0.9887	-0.0016	1.0073	0.0032	1.0068
Bias		0.0016	-0.0112	-0.0016	0.0073	0.0032	0.0068
MSE		0.0136	0.0016	0.0142	0.0133	0.0146	0.0137
	$(40, 0^{49})$						
Mean		-0.0047	0.9950	0.0030	1.0036	-0.0003	1.0103
Bias		-0.0047	-0.0049	0.0030	0.0036	-0.0003	0.0103
MSE		0.0179	0.0132	0.0179	0.0131	0.0189	0.0133
(90,60)	$(0^{59}, 30)$						
Mean		-0.0008	0.9906	0.0004	1.0031	0.0021	1.0072
Bias		-0.0008	-0.0093	0.0004	0.0031	0.0021	0.0072
MSE		0.0111	0.0119	0.0110	0.0121	0.0108	0.0120
	$(0^{20}, 2^{15}, 0^{25})$						
Mean		-0.0054	0.9866	-0.0012	1.0003	-0.0009	1.0051
Bias		-0.0054	-0.0133	-0.0012	0.0003	-0.0009	0.0051
MSE		0.0118	0.0116	0.0130	0.0116	0.0126	0.0114
	$(30, 0^{59})$						
Mean		-0.0020	0.9929	-0.0045	1.0069	-0.0022	1.0046
Bias		-0.0020	-0.0070	-0.0045	0.0069	-0.0022	0.0046
MSE		0.0152	0.0110	0.0159	0.0117	0.0157	0.0104

Table 5: Mean, Bias and MSE of S(t) Based on T=1.8

(n, m)	CS	MLE			MCMC0			MCMC1		
(50, 30)	$(0^{29}, 20)$	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**
Mean		0.7788	0.4909	0.2111	0.7767	0.4984	0.2248	0.7760	0.4978	0.2246
Bias		0.0003	-0.0090	−0.0104	-0.0017	−0.0015	0.0033	0.0024	-0.0021	0.0030
MSE		0.0025	0.0045	0.0036	0.0026	0.0044	0.0036	0.0024	0.0043	0.0034
	$(0^{10}, 2^{10}, 0^{10})$									
Mean		0.7784	0.4918	0.2132	0.7745	0.4950	0.2234	0.7761	0.4966	0.2241
Bias		−0.0000	-0.0081	−0.0082	-0.0038	−0.0049	0.0018	−0.0023	-0.0033	0.0025
MSE		0.0027	0.0051	0.0041	0.0027	0.0051	0.0039	0.0026	0.0051	0.0038
	$(20, 0^{29})$									
Mean		0.7757	0.4953	0.2185	0.7749	0.5016	0.2283	0.7734	0.5006	0.2286
Bias		−0.0026	-0.0046	−0.0029	-0.0035	0.0016	0.0067	-0.0049	0.0006	0.0070
MSE		0.0037	0.0061	0.0039	0.0036	0.0060	0.0039	0.0035	0.0060	0.0039
(50,40)	$(0^{39}, 10)$									
Mean		0.7815	0.4987	0.2172	0.7754	0.4983	0.2229	0.7748	0.4983	0.2233
Bias		0.0031	-0.0012	−0.0043	-0.0030	-0.0016	0.0013	-0.0035	-0.0016	0.0017
MSE		0.0025	0.0039	0.0027	0.0025	0.0038	0.0027	0.0024	0.0037	0.0025
	$(0^{15}, 1^{10}, 0^{15})$									
Mean		0.7806	0.4996	0.2196	0.7772	0.5004	0.2252	0.7759	0.4991	0.2245
Bias		0.0022	-0.0003	−0.0019	-0.0011	0.0004	0.0036	-0.0025	-0.0008	0.0029
MSE		0.0026	0.0043	0.0030	0.0024	0.0043	0.0030	0.0024	0.0039	0.0029
	$(10, 0^{39})$									
Mean		0.7788	0.4981	0.2189	0.7751	0.5007	0.2265	0.7728	0.4964	0.2227
Bias		0.0003	-0.0018	−0.0026	-0.0032	0.0007	0.0049	-0.0055	-0.0035	0.0011
MSE		0.0028	0.0046	0.0030	0.0029	0.0047	0.0031	0.0029	0.0048	0.0030
(70,40)	$(0^{39}, 30)$									
Mean		0.7791	0.4938	0.2143	0.7766	0.4956	0.2205	0.7765	0.4945	0.2194
Bias		0.0007	-0.0061	−0.0072	-0.0017	-0.0043	-0.0009	-0.0019	-0.0054	−0.0021
MSE		0.0018	0.0032	0.0028	0.0018	0.0032	0.0026	0.0018	0.0032	0.0027
	$(0^{10}, 2^{15}, 0^{15})$									
Mean		0.7788	0.4949	0.2164	0.7749	0.4968	0.2247	0.7764	0.4980	0.2248
Bias		0.0003	-0.0050	−0.0050	-0.0034	-0.0031	0.0031	-0.0019	-0.0019	0.0033
MSE		0.0020	0.0040	0.0031	0.0020	0.0038	0.0030	0.0019	0.0038	0.0030
	$(30, 0^{39})$									
Mean		0.7794	0.4987	0.2194	0.7728	0.4966	0.2234	0.7763	0.5008	0.2255
Bias		0.0010	-0.0012	−0.0021	-0.0055	-0.0033	0.0019	-0.0020	0.0008	0.0040
MSE		0.0028	0.0047	0.0030	0.0027	0.0044	0.0029	0.0026	0.0043	0.0028
(70,50)	$(0^{49}, 20)$									
Mean		0.7808	0.4986	0.2179	0.7778	0.5003	0.2239	0.7757	0.4989	0.2237
Bias		0.0024	-0.0013	−0.0035	-0.0006	0.0003	0.0023	-0.0027	-0.0010	0.0022
MSE		0.0017	0.0028	0.0021	0.0017	0.0028	0.0021	0.0018	0.0028	0.0020
	$(0^{20}, 2^{10}, 0^{20})$									
Mean		0.7786	0.4970	0.2183	0.7750	0.4980	0.2243	0.7768	0.5013	0.2252
Bias		0.0001	-0.0029	−0.0031	-0.0033	-0.0019	0.0027	-0.0015	0.0013	0.0036
MSE		0.0018	0.0031	0.0023	0.0017	0.0030	0.0023	0.0022	0.0035	0.0023
	$(20, 0^{49})$									
Mean		0.7800	0.4991	0.2188	0.7767	0.5014	0.2256	0.7756	0.4995	0.2239
Bias		0.0016	-0.0008	−0.0026	-0.0016	0.0014	0.0040	-0.0027	-0.0004	0.0023
MSE		0.0023	0.0037	0.0023	0.0022	0.0037	0.0024	0.0022	0.0035	0.0023

(n, m)	CS	MLE			MCMC0			MCMC1		
(90,50)	$(0^{49}, 40)$	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**
Mean		0.7798	0.4949	0.2150	0.7785	0.4989	0.2231	0.7782	0.4986	0.2227
Bias		0.0014	-0.0050	−0.0065	0.0008	-0.0010	0.0016	-0.0001	-0.0013	0.0011
MSE		0.0014	0.0025	0.0023	0.0014	0.0026	0.0023	0.0013	0.0024	0.0021
	$(0^{15}, 2^{20}, 0^{15})$									
Mean		0.7805	0.4989	0.2197	0.7748	0.4956	0.2226	0.7764	0.4978	0.2239
Bias		0.0020	-0.0010	−0.0017	-0.0035	-0.0043	0.0010	-0.0020	-0.0021	0.0024
MSE		0.0016	0.0031	0.0025	0.0015	0.0029	0.0023	0.0015	0.0029	0.0024
	$(40, 0^{49})$									
Mean		0.7770	0.4976	0.2200	0.7763	0.5004	0.2251	0.7738	0.4991	0.2256
Bias		-0.0014	-0.0023	−0.0015	-0.0020	0.0004	0.0035	-0.0046	-0.0008	0.0041
MSE		0.0023	0.0037	0.0024	0.0021	0.0036	0.0024	0.0023	0.0037	0.0024
(90,60)	$(0^{59}, 30)$									
Mean		0.7799	0.4987	0.2192	0.7775	0.4988	0.2221	0.7772	0.4995	0.2234
Bias		0.0014	-0.0012	−0.0022	-0.0008	-0.0011	0.0005	-0.0012	-0.0004	0.0019
MSE		0.0014	0.0023	0.0018	0.0014	0.0022	0.0017	0.0013	0.0022	0.0017
	$(0^{20}, 2^{15}, 0^{25})$									
Mean		0.7791	0.4963	0.2170	0.7766	0.4970	0.2215	0.7757	0.4972	0.2226
Bias		0.0006	-0.0036	−0.0045	-0.0017	-0.0029	0.0001	-0.0026	-0.0027	0.0010
MSE		0.0014	0.0025	0.0020	0.0014	0.0026	0.0021	0.0014	0.0025	0.0020
	$(30, 0^{59})$									
Mean		0.7783	0.4987	0.2202	0.7737	0.4976	0.2232	0.7748	0.4985	0.2236
Bias		−0.0005	-0.0012	−0.0013	-0.0047	-0.0023	0.0017	-0.0035	-0.0014	0.0020
MSE		0.0019	0.0031	0.0020	0.0020	0.0032	0.0020	0.0019	0.0031	0.0020

Additionally, a simulation study is executed to consider the coverage probability and average length as shown below in Tables 6–9 for the two distinct values of T. From these tables, it is considerable that the expected width of the confidence interval (CI) and the credible interval decrease for all estimators in all methods as *n* and *m* increase as well as *n* is fixed and *m* increases. Furthermore, it is clear that the values of average length are small in all methods with different cases which means that the better performance of the confidence and Bayes intervals. The coverage probabilities of the estimate confidence intervals in the MLEs are close to the nominal level of 0.95 for μ, σ, and S (t=0.5, 1, 2) as *n* become large, but fail to reach to the desired level as *n* become small. Alternately, it is clear that in most cases the coverage probabilities of the credible intervals are approach to the nominal level of 0.95 for μ, σ, and S (t=0.5, 1, 2). In addition, it is obvious that there is no big difference between the two distinct values based on estimated criteria.

Table 6: Expected Length of 95% CI. & Coverage Probability for Location & Scale Parameters Based on T=1

(n, m)	CS	MLE		MCMC0		MCMC1	
(50,30)	$(0^{29}, 20)$	**μ**	**σ**	**μ**	**σ**	**μ**	**σ**
EL.		0.5467	0.6056	0.5821	0.6459	0.5765	0.6320
CP.		0.9375	0.9155	0.9465	0.9405	0.9365	0.9515
	$(0^{10}, 2^{10}, 0^{10})$						
EL.		0.5925	0.5788	0.6207	0.6089	0.6128	0.5991
CP.		0.9275	0.9200	0.9460	0.9475	0.9415	0.9555
	$(20, 0^{29})$						
EL.		0.6719	0.5630	0.6940	0.5823	0.6908	0.5773
CP.		0.9475	0.9275	0.9485	0.9495	0.9470	0.9390
(50,40)	$(0^{39}, 10)$						
EL.		0.5225	0.5083	0.5428	0.5327	0.5390	0.5255
CP.		0.936	0.926	0.9480	0.9515	0.9550	0.9455
	$(0^{15}, 1^{10}, 0^{15})$						
EL.		0.5397	0.5051	0.5588	0.5271	0.5547	0.5214
CP.		0.9485	0.927	0.9385	0.9440	0.9410	0.9525
	$(10, 0^{39})$						
EL.		0.5835	0.5013	0.6051	0.5214	0.5946	0.5114
CP.		0.9445	0.9275	0.9380	0.9420	0.9505	0.9525
(70,40)	$(0^{39}, 30)$						
EL.		0.4677	0.5262	0.4926	0.5536	0.4898	0.5462
CP.		0.9405	0.9275	0.9380	0.9460	0.9410	0.9450
	$(0^{10}, 2^{15}, 0^{15})$						
EL.		0.5213	0.4962	0.5378	0.5143	0.5307	0.5076
CP.		0.9340	0.9325	0.9460	0.9520	0.9460	0.9530
	$(30, 0^{39})$						
EL.		0.5793	0.4869	0.6004	0.5046	0.5940	0.4972
CP.		0.9410	0.9280	0.9490	0.9465	0.9490	0.9415
(70,50)	$(0^{49}, 20)$						
EL.		0.4482	0.4619	0.4607	0.4763	0.4581	0.4724
CP.		0.9500	0.9385	0.9450	0.9505	0.9395	0.9495
	$(0^{20}, 2^{10}, 0^{20})$						
EL.		0.4755	0.4543	0.4864	0.4667	0.4824	0.4623
CP.		0.948	0.9345	0.9455	0.9425	0.9430	0.9485
	$(20, 0^{49})$						
EL.		0.5250	0.4489	0.5314	0.4554	0.5290	0.4513
CP.		0.9425	0.9300	0.9440	0.9470	0.9435	0.9475
(90,50)	$(0^{49}, 40)$						
EL.		0.4194	0.4756	0.4324	0.4902	0.4349	0.4882
CP.		0.9435	0.9325	0.9480	0.9450	0.9490	0.9420

	$(0^{15}, 2^{20}, 0^{15})$						
EL.		0.4574	0.4484	0.4666	0.4609	0.4630	0.4546
CP.		0.9285	0.9385	0.9385	0.9390	0.9420	0.9485
	$(40,0^{49})$						
EL.		0.5259	0.4422	0.5313	0.4480	0.5258	0.4418
CP.		0.9405	0.9390	0.9465	0.9490	0.9455	0.9445
(90,60)	$(0^{59}, 30)$						
EL.		0.4007	0.4266	0.4059	0.4335	0.4074	0.4332
CP.		0.9390	0.9415	0.9420	0.9500	0.9310	0.9490
	$(0^{20}, 2^{15}, 0^{25})$						
EL.		0.4349	0.4132	0.4398	0.4202	0.4367	0.4141
CP.		0.9430	0.9445	0.9435	0.9490	0.9405	0.9420
	$(30, 0^{59})$						
EL.		0.4787	0.4078	0.4826	0.4130	0.4825	0.4120
CP.		0.9395	0.9350	0.9425	0.9400	0.9400	0.9400

Table 7: Expected Length of 95% CI. & coverage Probability for S(t) Based on T=1

$(\boldsymbol{n}, \boldsymbol{m})$	**CS**	**MLE**			**MCMC0**			**MCMC1**		
(50,30)	$(0^{29}, 20)$	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**
EL.		0.1955	0.2523	0.2303	0.1946	0.2487	0.2321	0.1927	0.2471	0.2298
CP.		0.9245	0.9405	0.9145	0.9500	0.9465	0.9440	0.9405	0.9365	0.9490
	$(0^{10}, 2^{10}, 0^{10})$									
EL.		0.1984	0.2724	0.2420	0.1957	0.2668	0.2425	0.1954	0.2653	0.2393
CP.		0.9335	0.9345	0.9090	0.9455	0.9460	0.9420	0.9485	0.9415	0.9460
	$(20, 0^{29})$									
EL.		0.2358	0.3041	0.2440	0.2322	0.2962	0.2432	0.2302	0.2950	0.2419
CP.		0.9305	0.943	0.9170	0.9395	0.9485	0.9465	0.9445	0.9470	0.9435
(50,40)	$(0^{39}, 10)$									
EL.		0.1931	0.2399	0.1997	0.1918	0.2354	0.1996	0.1909	0.2349	0.1985
CP.		0.9315	0.9345	0.9170	0.9510	0.9480	0.9550	0.9545	0.9550	0.9495
	$(0^{15}, 1^{10}, 0^{15})$									
EL.		0.1929	0.2478	0.2075	0.1917	0.2430	0.2067	0.1912	0.2423	0.2054
CP.		0.9335	0.946	0.9285	0.9410	0.9385	0.9365	0.9450	0.9410	0.9490
	$(10, 0^{39})$									
EL.		0.2088	0.2652	0.2127	0.2073	0.2600	0.2124	0.2059	0.2586	0.2100
CP.		0.9305	0.9420	0.9330	0.9395	0.9380	0.9480	0.9420	0.9505	0.9490
(70,40)	$(0^{39}, 30)$									
EL.		0.1659	0.2170	0.2028	0.1641	0.2135	0.2033	0.1638	0.2131	0.2023
CP.		0.9285	0.9435	0.9300	0.9380	0.9380	0.9380	0.9505	0.9410	0.9490
	$(0^{10}, 2^{15}, 0^{15})$									
EL.		0.1705	0.2387	0.2145	0.1690	0.2349	0.2134	0.1678	0.2331	0.2113
CP.		0.9360	0.9370	0.9190	0.9465	0.9460	0.9430	0.9375	0.9460	0.9500
	$(30, 0^{39})$									
EL.		0.2054	0.2641	0.2116	0.2031	0.2586	0.2122	0.2017	0.2577	0.2106
CP.		0.9350	0.9390	0.922	0.9465	0.9490	0.9440	0.9440	0.9490	0.9490

(n,m)	CS	MLE			MCMC0			MCMC1		
(70,50)	$(0^{49},20)$	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**
EL.		0.1654	0.2056	0.1782	0.1628	0.2018	0.1784	0.1629	0.2012	0.1766
CP.		0.9415	0.9480	0.9380	0.9420	0.9450	0.9475	0.9380	0.9395	0.9470
	$(0^{20},2^{10},0^{20})$									
EL.		0.1646	0.2171	0.1892	0.1633	0.2133	0.1878	0.1624	0.2125	0.1861
CP.		0.9380	0.9455	0.9325	0.9350	0.9455	0.9395	0.9480	0.9430	0.9445
	$(20,0^{49})$									
EL.		0.1873	0.2373	0.1910	0.1847	0.2329	0.1892	0.1838	0.2319	0.1877
CP.		0.9400	0.9390	0.9320	0.9440	0.9440	0.9475	0.9415	0.9435	0.9455
(90,50)	$(0^{49},40)$									
EL.		0.1468	0.1934	0.1837	0.1450	0.1904	0.1830	0.1449	0.1897	0.1827
CP.		0.9380	0.9445	0.9235	0.9415	0.9480	0.9465	0.9320	0.9490	0.9495
	$(0^{15},2^{20},0^{15})$									
EL.		0.1495	0.2096	0.1919	0.1482	0.2064	0.1895	0.1468	0.2048	0.1888
CP.		0.9385	0.9370	0.9195	0.9455	0.9385	0.9315	0.9345	0.9420	0.9475
	$(40,0^{49})$									
EL.		0.1848	0.2368	0.1918	0.1828	0.2323	0.1889	0.1818	0.2309	0.1869
CP.		0.9305	0.9345	0.9285	0.9530	0.9465	0.9435	0.9405	0.9455	0.9475
(90,60)	$(0^{59},30)$									
EL.		0.1466	0.1835	0.1636	0.1440	0.1804	0.1627	0.1442	0.1801	0.1619
CP.		0.9455	0.9410	0.925	0.9395	0.9420	0.9505	0.9355	0.9310	0.9375
	$(0^{20},2^{15},0^{25})$									
EL.		0.1475	0.1981	0.1751	0.1449	0.1941	0.1737	0.1448	0.1939	0.1722
CP.		0.9520	0.9435	0.9285	0.9400	0.9435	0.9440	0.9405	0.9405	0.9395
	$(30,0^{59})$									
EL.		0.1706	0.2170	0.1749	0.1686	0.2127	0.1729	0.1677	0.2118	0.1722
CP.		0.9355	0.9370	0.9230	0.9440	0.9425	0.9465	0.9425	0.9400	0.9460

Table 8: Expected Length of 95% CI. & Coverage Probability of Location & Scale Parameters Based on T=1.8

(n, m)	CS	MLE		MCMC0		MCMC1	
(50,30)	$(0^{29}, 20)$	**μ**	**σ**	**μ**	**σ**	**μ**	**σ**
EL.		0.5446	0.6033	0.5804	0.6445	0.5765	0.6334
CP.		0.9350	0.9140	0.9430	0.9440	0.9455	0.9555
	$(0^{10}, 2^{10}, 0^{10})$						
EL.		0.5942	0.5771	0.6212	0.6098	0.6163	0.5986
CP.		0.9345	0.9190	0.9430	0.9500	0.9440	0.9520
	$(20, 0^{29})$						
EL.		0.6738	0.5652	0.6942	0.5831	0.6897	0.5769
CP.		0.9455	0.9315	0.9495	0.9445	0.9430	0.9570
(50,40)	$(0^{39}, 10)$						
EL.		0.5226	0.5085	0.5410	0.5308	0.5414	0.5255
CP.		0.9385	0.9285	0.9425	0.9515	0.9500	0.9475
	$(0^{15}, 1^{10}, 0^{15})$						
EL.		0.5460	0.5098	0.5614	0.5267	0.5592	0.5216
CP.		0.9380	0.9210	0.9340	0.9475	0.9445	0.9465
	$(10, 0^{39})$						
EL.		0.5836	0.5019	0.6009	0.5179	0.5932	0.5110
CP.		0.9450	0.9280	0.9390	0.9370	0.9365	0.9505
(70,40)	$(0^{39}, 30)$						
EL.		0.4707	0.5293	0.4889	0.5507	0.4856	0.5399
CP.		0.9420	0.9265	0.9455	0.9450	0.9445	0.9495
	$(0^{10}, 2^{15}, 0^{15})$						
EL.		0.5219	0.4962	0.5410	0.5162	0.5355	0.5100
CP.		0.9365	0.9340	0.9430	0.9555	0.9440	0.9440
	$(30, 0^{39})$						
EL.		0.5810	0.4877	0.5960	0.5004	0.5914	0.4945
CP.		0.9380	0.9335	0.9450	0.9485	0.9420	0.9480
(70,50)	$(0^{49}, 20)$						
EL.		0.4484	0.4623	0.4614	0.4778	0.4609	0.4752
CP.		0.9430	0.9340	0.9430	0.9515	0.9385	0.9510
	$(0^{20}, 2^{10}, 0^{20})$						
EL.		0.4782	0.4564	0.4896	0.4693	0.5292	0.4519
CP.		0.9465	0.9385	0.9480	0.9445	0.9475	0.9425
	$(20, 0^{49})$						
EL.		0.5207	0.4450	0.5322	0.4547	0.5290	0.4513
CP.		0.9505	0.931	0.9445	0.9520	0.9410	0.9475
(90,50)	$(0^{49}, 40)$						
EL.		0.4193	0.4754	0.4332	0.4923	0.4330	0.4872
CP.		0.9370	0.9265	0.9310	0.9405	0.9455	0.9510
	$(0^{15}, 2^{20}, 0^{15})$						
EL.		0.4616	0.4508	0.4719	0.4635	0.4697	0.4597
CP.		0.9420	0.937	0.9455	0.9485	0.9425	0.9360
	$(40, 0^{49})$						
EL.		0.5246	0.4411	0.5323	0.4466	0.5337	0.4467
CP.		0.9470	0.9330	0.9445	0.9440	0.9445	0.9415

(90,60)	$(0^{59}, 30)$						
EL.		0.4021	0.4279	0.4071	0.4356	0.4095	0.4352
CP.		0.9350	0.9420	0.9370	0.9495	0.9430	0.9455
	$(0^{20}, 2^{15}, 0^{25})$						
EL.		0.4332	0.4113	0.4377	0.4185	0.4392	0.4179
CP.		0.9460	0.9230	0.9360	0.9400	0.9450	0.9425
	$(30, 0^{59})$						
EL.		0.4788	0.4082	0.4851	0.4141	0.4813	0.4104
CP.		0.9450	0.9390	0.9335	0.9440	0.9470	0.9490

Table 9: Expected Length of 95% CI. & Coverage Probability of S(t) Based on T=1.8

(n, m)	**CS**	**MLE**			**MCMC0**			**MCMC1**		
(50,30)	$(0^{29}, 20)$	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**	**S (0.5)**	**S (1)**	**S (2)**
EL.		0.1960	0.2526	0.2296	0.1935	0.2482	0.2320	0.1927	0.2473	0.2295
CP.		0.9280	0.9425	0.9095	0.9435	0.9430	0.9370	0.9465	0.9455	0.9435
	$(0^{10}, 2^{10}, 0^{10})$									
EL.		0.1980	0.2738	0.2437	0.1963	0.2682	0.2434	0.1941	0.2663	0.2420
CP.		0.9310	0.9380	0.9125	0.9405	0.9430	0.9455	0.9430	0.9440	0.9460
	$(20, 0^{29})$									
EL.		0.2371	0.3039	0.2437	0.2317	0.2968	0.2441	0.2303	0.2943	0.2419
CP.		0.9360	0.9430	0.9210	0.9465	0.9495	0.9485	0.9445	0.9430	0.9470
(50,40)	$(0^{39}, 10)$									
EL.		0.1933	0.2399	0.1996	0.1920	0.2353	0.1988	0.1916	0.2351	0.1980
CP.		0.9270	0.9385	0.9250	0.9405	0.9425	0.9450	0.9490	0.9500	0.9475
	$(0^{15}, 1^{10}, 0^{15})$									
EL.		0.1923	0.2486	0.2107	0.1909	0.2444	0.2098	0.1903	0.2432	0.2078
CP.		0.9215	0.9350	0.9245	0.9465	0.9340	0.9380	0.9435	0.9445	0.9420
	$(10, 0^{39})$									
EL.		0.2093	0.2651	0.2122	0.2070	0.2599	0.2117	0.2061	0.2579	0.2090
CP.		0.9320	0.9415	0.9295	0.9440	0.9390	0.9390	0.9450	0.9365	0.9445
(70,40)	$(0^{39}, 30)$									
EL.		0.1662	0.2170	0.2031	0.1645	0.2142	0.2031	0.1633	0.2128	0.2005
CP.		0.9325	0.9430	0.9230	0.9440	0.9455	0.9450	0.9420	0.9445	0.9420
	$(0^{10}, 2^{15}, 0^{15})$									
EL.		0.1701	0.2390	0.2154	0.1690	0.2353	0.2149	0.1675	0.2334	0.2129
CP.		0.9300	0.9365	0.9245	0.9375	0.9430	0.9490	0.9495	0.9440	0.9490
	$(30, 0^{39})$									
EL.		0.2051	0.2641	0.2125	0.2033	0.2591	0.2107	0.2013	0.2577	0.2103
CP.		0.9305	0.9325	0.9285	0.9475	0.9450	0.9520	0.9510	0.9420	0.9485
(70,50)	$(0^{49}, 20)$									
EL.		0.1650	0.2055	0.1785	0.1635	0.2022	0.1782	0.1630	0.2018	0.1772
CP.		0.9330	0.9400	0.9275	0.9475	0.9430	0.9445	0.9395	0.9385	0.9435
	$(0^{20}, 2^{10}, 0^{20})$									
EL.		0.1649	0.2178	0.1903	0.1634	0.2141	0.1890	0.1834	0.2318	0.1886
CP.		0.9375	0.9455	0.936	0.9530	0.9480	0.9420	0.9415	0.9475	0.9490
	$(20, 0^{49})$									
EL.		0.1867	0.2374	0.1903	0.1845	0.2329	0.1898	0.1839	0.2323	0.1883
CP.		0.9355	0.9475	0.9385	0.9480	0.9445	0.9430	0.9415	0.9410	0.9415

(90,50)	$(0^{49}, 40)$									
EL.		0.1467	0.1933	0.1838	0.1445	0.1899	0.1833	0.1447	0.1904	0.1830
CP.		0.9400	0.9440	0.9245	0.9460	0.9310	0.9340	0.9515	0.9455	0.9435
	$(0^{15}, 2^{20}, 0^{15})$									
EL.		0.1488	0.2103	0.1945	0.1483	0.2077	0.1924	0.1472	0.2059	0.1910
CP.		0.9290	0.939	0.9245	0.9445	0.9455	0.9505	0.9370	0.9425	0.9420
	$(40,0^{49})$									
EL.		0.1854	0.2369	0.1909	0.1824	0.2332	0.1904	0.1827	0.2322	0.1893
CP.		0.9400	0.9440	0.9300	0.9480	0.9445	0.9445	0.9430	0.9445	0.9425
(90,60)	$(0^{59}, 30)$									
EL.		0.1463	0.1834	0.1642	0.1437	0.1798	0.1623	0.1442	0.1806	0.1629
CP.		0.9420	0.9375	0.9265	0.9415	0.9370	0.9440	0.9430	0.9430	0.9435
	$(0^{20}, 2^{15}, 0^{25})$									
EL.		0.1470	0.1983	0.1752	0.1450	0.1944	0.1731	0.1452	0.1940	0.1724
CP.		0.9395	0.9465	0.9315	0.9400	0.9360	0.9350	0.9505	0.9450	0.9395
	$(30, 0^{59})$									
EL.		0.1712	0.2170	0.1745	0.1695	0.2129	0.1729	0.1684	0.2117	0.1717
CP.		0.9375	0.9405	0.9260	0.9325	0.9335	0.9340	0.9430	0.9470	0.9390

CHAPTER 4: REAL DATA ANALYSIS

The data set was found originally by Nichols and Padgett (2006). It has been analyzed by Lemonte (2014) and by AL Sobhi and Soliman (2016) . The uncensored data on breaking stress of carbon fibers (in Gpa) is composed of 100 observations as shown below in Table 10. Carbon fiber is composed of carbon atoms bounded together to form along chain. Carbon fibers are extremely stiff, strong, and light. Although carbon fiber has many significant benefits over other materials, it is more expensive than traditional materials such as steel, aluminum, and plastic. It is used in many processes to create excellent building materials such as solid carbon sheets and carbon tubes. The most common uses for carbon fiber in applications are in sports equipment and robotics.

Table 10: Real Data Set on Breaking Stress of Carbon Fibers (in Gpa)

Real Data Set on Breaking Stress of Carbon Fibers (in Gpa)									
0.39	0.81	0.85	0.98	1.08	1.12	1.17	1.18	1.22	1.25
1.36	1.41	1.47	1.57	1.57	1.59	1.59	1.61	1.61	1.69
1.69	1.71	1.73	1.80	1.84	1.84	1.87	1,89	1.92	2.00
2.03	2.03	2.05	2.12	2.17	2.17	2.17	2.35	2.38	2.41
2.43	2.48	2.48	2.50	2.53	2.55	2.55	2.56	2.59	2.67
2.73	2.74	2.76	2.77	2.79	2.81	2.81	2.82	2.83	2.85
2.87	2.88	2.93	2.95	2.96	2.97	2.97	3.09	3.11	3.11
3.15	3.15	3.19	3.19	3.22	3.22	3.27	3.28	3.31	3.31

3.33	3.39	3.39	3.51	3.56	3.60	3.65	3.68	3.68	3.68
3.70	3.75	4.20	4.38	4.42	4.70	4.90	4.91	5.08	5.56

In this illustrative example, the data set was used to simulate an adaptive progressive type-II censored sample with *m*=60 with two distinct values of ideal total test time T as (1.60,3.66); the progressive censoring scheme was considered as $R = (30, 0^{*58}, 10)$. For clarity $R = (1, 0^{*4}, 3)$is a short form of $R = (1,0,0,0,0,3)$. Moreover, the function sample in the R program was used to remove randomly 30 survival units from 99 patients at the first failure; then, the remaining 10 survival units at the last failure were removed. Thus, the observed adaptive progressive type-II censored samples are shown below in Table 11 for two different number of T and two distinct number of J. Since J=13 represents that only 13 observed failure times were observed before time T=1.60 and J=60 represents that all the observed failure times were observed before time T=3.66, this implies that the experiment ends before time T.

Table 11: An Adaptive Progressive Type-II Censored Samples of Real Data Based on T= (1.60,3.66)

An Adaptive Progressive Type-II Censored Samples of Real Data Based on T= (1.60,3.66)					
T=1.60, J=13					
0.39	0.81	0.85	0.98	1.12	1.17
1.22	1.25	1.36	1.41	1.57	1.59
1.59	1.61	1.69	1.69	1.71	1.73
1.80	1.84	1.84	1.89	1.92	2.00
2.03	2.05	2.12	2.17	2.17	2.35
2.41	2.43	2.48	2.48	2.50	2.55
2.59	2.76	2.79	2.81	2.81	2.81
2.83	2.85	2.87	2.93	2.95	2.96
2.97	3.09	3.11	3.15	3.19	3.19
3.22	3.31	3.31	3.33	3.39	3.51
T=3.66, J=60					
0.39	0.85	1.08	1.17	1.22	1.25
1.36	1.57	1.59	1.59	1.61	1.71
1.73	1.80	1.84	1.84	1.87	1.89
1.92	2.03	2.03	2.05	2.12	2.17
2.17	2.38	2.41	2.43	2.48	2.53
2.55	2.55	2.67	2.73	2.74	2.76
2.77	2.79	2.81	2.81	2.82	2.83
2.87	2.95	2.96	2.97	3.09	3.11
3.11	3.15	3.19	3.19	3.22	3.27
3.28	3.31	3.39	3.51	3.60	3.65

The MLE's for the unknown quantities are computed for the complete sample (un censored) i.e.$(n = m = 100)$ as $(\hat{\mu}_{ML}, \hat{\sigma}_{ML})$= $(0.9156, 0.4404)$ and $(\hat{S}(t = 0.5)) = (0.9986)$. To know if a sample follow a log-logistic distribution, we need to apply the Kolmogorov-Smirnov test for one sample, the estimate of the parameters for the complete sample was used to standardize the data and transform it to logistic distribution namely x=(**π**/sqrt(3))*(log(y)-μ)/σ, since the non-standard form of the parameters in the log-logistic model is different from that adopted in R. It was noted that under the significance level (0.05), the p-value=0.3927, is greater than the significance level and the test statistics value equal 0.090001, which is too small. This implies that, the proposed log-logistic model fits the sample data fully and this sample followed a log-logistic distribution. Based on the observed samples, corresponding respectively to predetermined ideal test times and according to the proposed MCMC algorithm described above in the previous chapter, different estimators and the related 95% confidence intervals of μ, σ, S (0.5) are computed to describe the provided methods of estimation. Since, there are no previous knowledge available for the unknown parameters, the diffuse priors for both μ and σ were used to generate 11000 MCMC samples as $((\mu_i, \sigma_i), i = 1,2, \dots \dots, 11000)$ and then discard the first 1000 random values according to the convergence diagnostic procedure explained in the previous chapter. The trace plot is shown below on the left side of Figures 10 and 11 for each parameter, which indicates the successive draw for μ and σ at each iteration, respectively. Moreover, as it is clear from the figures on the right, the marginal posterior density plot from both parameters were unsmoothed.

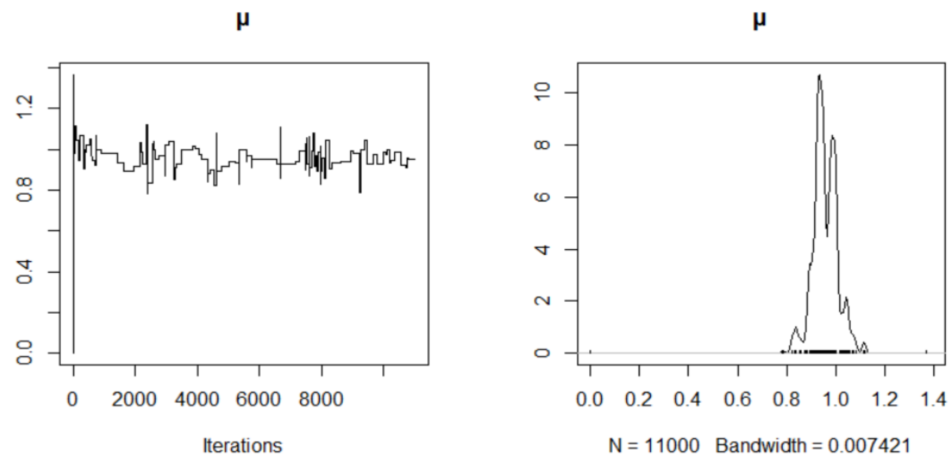

Figure 12: Trace plot and probability density plot of the location parameter based on a real data

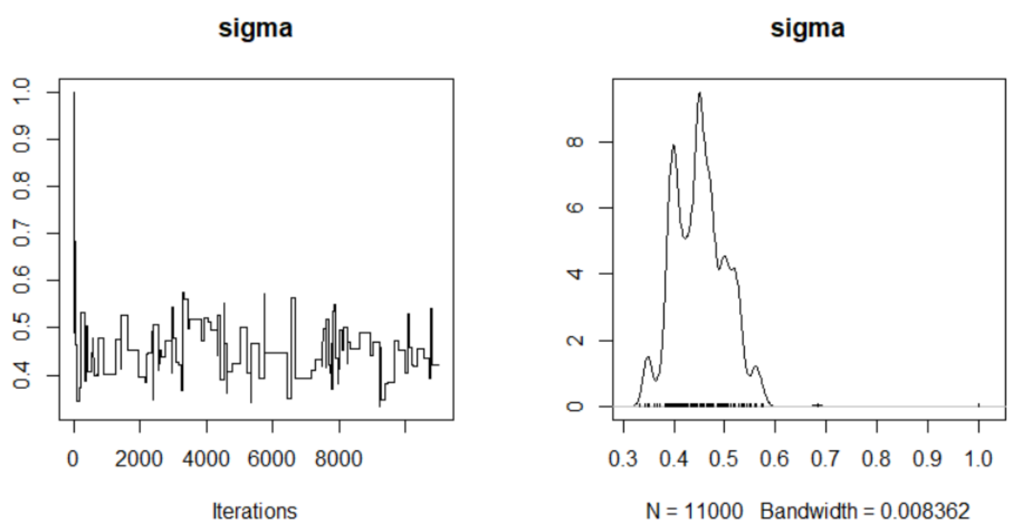

Figure 13: Trace plot and probability density plot of the scale parameter based on a real data

Therefore, based on autocorrelation plots shown below in Figures 12 and 13 the modified plots for the marginal density for both parameters are those shown in Figures 14 and 15 after removing the first 1000 values from the chains, taking the same approach as explained in the previous chapter.

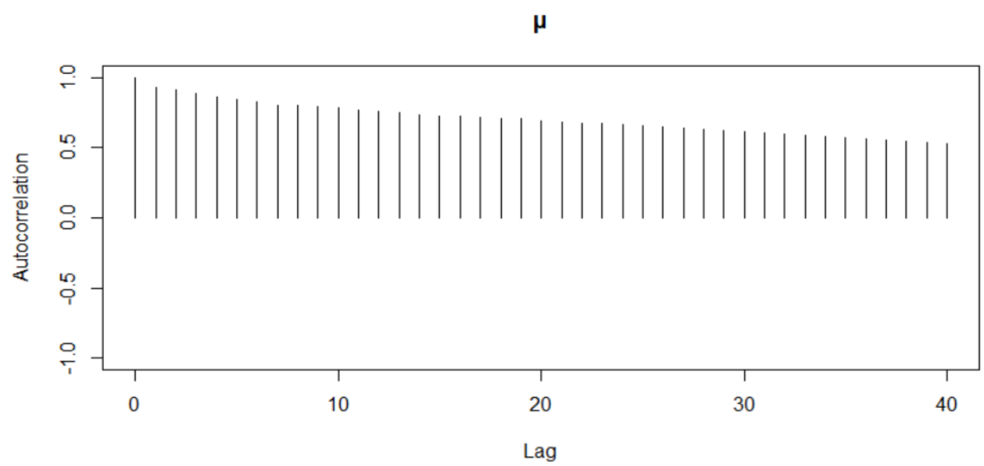

Figure 14: Autocorrelation plot of the location parameter based on a real data

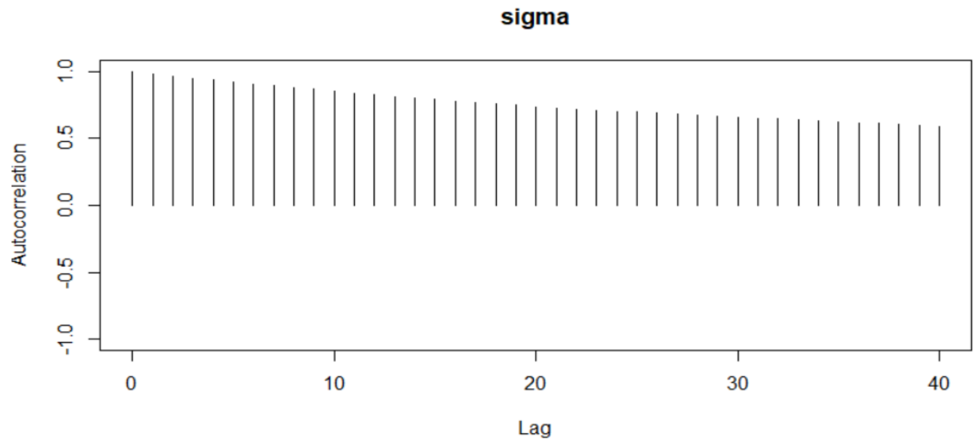

Figure 15: Autocorrelation plot of the scale parameter based on a real data

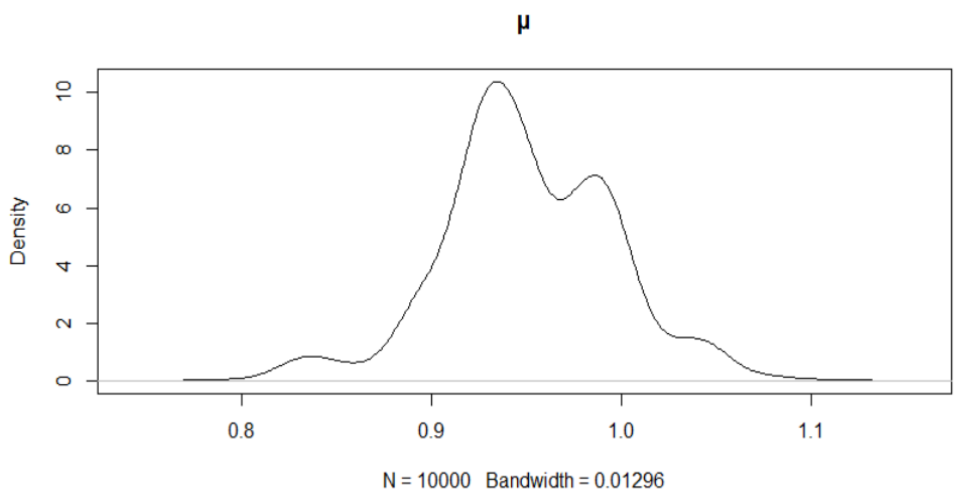

Figure 16: Smooth probability density plot for the location parameter based on a real data

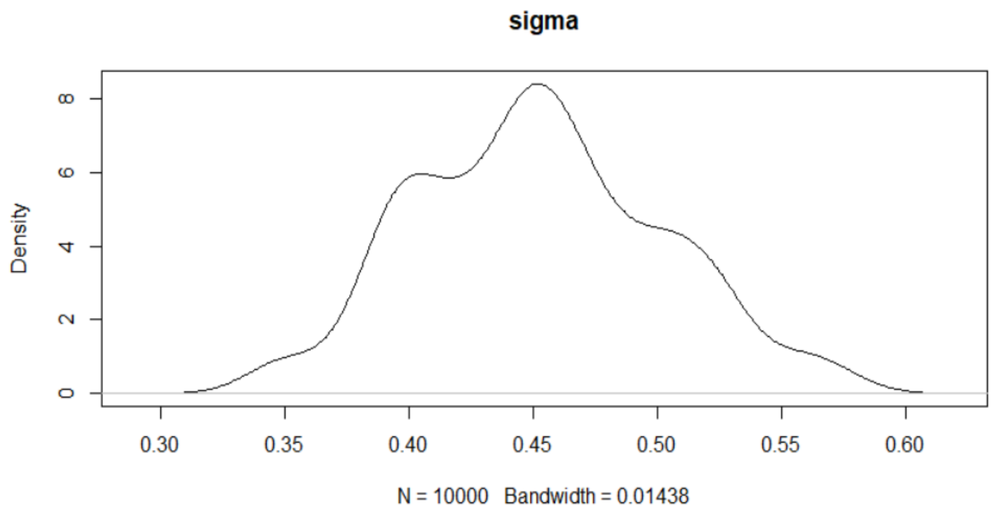

Figure 17: Smooth probability density plot for the scale parameter based on a real data

Table 12 summaries the MLEs for parameters μ, σ, and S (0.5) via the censored sample, which provides a close estimate to the estimate of the same parameters' by using the complete sample. This implies that the experiments ending at time $x_{m:m:n} = 3.65$ or $x_{m:m:n} = 3.51$ provide close estimates for the parameters where the experiment ends at time $x_{m:m:n} = 5.56$, which is desirable for obtain a highly efficient of estimation in cases of time reduction. Furthermore, from this table, the Bayes estimates under the MCMC0 prior and the MLE are near to each other assuming two distinct ideal total test times.

Table 12: Point Estimations of the Location & Scale Parameters & S (0.5) Based on a Real Data Set

	T=1.60		T=3.66	
	MLE	MCMC0	MLE	MCMC0
μ	0.8814	0.8810	0.9425	0.9508
σ	0.4846	0.4959	0.4448	0.4523
S (0.5)	0.9972	0.9964	0.9987	0.9983

The approximate 95% confidence intervals have been computed, as well as the corresponding length for each interval as reported below in Table 13. It is obvious that the length of the MLE method is nearly as short as the MCMC0 method, which means that the realization of the confidence interval is better and this illustrative example in a real data set is approximately consistent with the results obtained in the simulation study. The R code for the illustrative real-life example is given in full in Appendix C.

Table 13: 95% Confidence Intervals and Length of the Location & Scale Parameters & S (0.5) Based on Real Data Set

		T=1.60		T=3.66	
		Interval	Length	Interval	Length
MLE	μ	(0.7731,0.9897)	0.2166	(0.8436,1.0414)	0.1977
	σ	(0.3827,0.5866)	0.2038	(0.3503,0.5393)	0.1890
	S (0.5)	(0.9936,1.0008)	0.0072	(0.9968,1.0006)	0.0037
MCMC0					
	μ	(0.7947,0.9868)	0.1921	(0.8370,1.0470)	0.2100
	σ	(0.3708,0.5946)	0.2237	(0.3508,0.5605)	0.2096
	S (0.5)	(0.9911,0.9995)	0.0084	(0.9946,0.9997)	0.0051

CHAPTER 5: SUMMARY, CONCLUSION AND SUGGESTIONS FOR FURTHER STUDY

5.1 Summary

This study provided the maximum likelihood and Bayesian approach to estimate the parameters of the log-logistic model and survival function under an adaptive progressive type-II censored data. The approximate MLEs of the parameters and the survival function were computed using the Newton-Raphson numerical method (owing to the non-closed form equations). Additionally, the asymptotic confidence intervals for μ and σ parameters via the variance-covariance matrix ($I^{-1}(\hat{\mu},\hat{\sigma})$) were constructed. Moreover, the Delta method was considered to approximate the confidence interval for the reliability function. Furthermore, the Bayesian approach presented was based on non-informative priors for both the unknown parameters and, in another case, for an informative logistic conjugate prior for μ and gamma prior for σ. The Bayes estimates under the SELF cannot be solved analytically, due to the complexity of the ratio of two integrals. Thus, the Metropolis Hastings algorithm was provided to generate 11000 samples and then the first 1000 draws was removed as discarded values based on a convergence diagnostic via the coda package. The two unknown parameters are approximated using Bayesian approach and, consequently, the corresponding credible intervals for these quantities and for the reliability function were computed. Next, a simulation study examined a case of 2000, replicated to investigate the realization of the derived methods for various values of sample sizes *n*, effective sample sizes *m*, and the three different progressive censoring schemes for each different choice of *n* and *m*. The proposed methods were examined on the basis of real-life example.

5.2 Conclusion

Based on results, the non-classical method (Bayesian approach) for the parameters and credible intervals is recommended when informative prior information exist for the unknown parameters; else, the classical method (MLE) for estimating the parameters and Bayes interval according to non-informative prior for interval estimation is better to perform. In addition, based on the estimated coverage probability, it was obvious that the intervals based on MLE were consequently reached to the nominal level by increasing the sample size and the effective sample size while the credible intervals approach to the nominal level based on different choices of sample size n and m. Moreover, the Biases were small in all situations based in all methods which tends to approximately unbiased estimators. Furthermore, from comparing the effect of different progressive censoring schemes according to the estimated criteria, it has been noted that the MSEs were close for the three progressive chosen censoring schemes in each choice of n and m. However, it is suggested to avoid using censoring schemes with (n-m) removal units at time of the first failure of the experiment, because if we remove more units at the beginning, we will lose more information and the MSE will be high. Additionally, the duration total test time will be large based on the behaviour of order statistics. Based on this study, it was clear that the MSEs have the smallest values in MCMC1 method in most situations with various sample sizes n and different effective sample sizes m. In addition, the expected width under all choices were small. Also, the coverage probability estimated criterion indicated that the credible intervals based in all cases were approach to near to the nominal level in MCMC1 better than ML estimation method. As a result, we can conclude that the Bayesian approach based on the suggested informative prior is a good substitutional to the MLE. Application to the real data set was also considered to estimate the parameters and survival function. Therefore, the data was used to simulate two samples based on an adaptive progressive type-II censored scheme according to two various of ideal test time to explain the two situations mentioned in the chapter introduction for this censoring scheme. Then, the point and interval estimated were computed based on this simulated samples. Hence, the results were consistent approximately with that results obtained in the simulation study which is the MLE method is quite close to the estimate using Bayesian approach under non informative prior.

5.3 Suggestions for Further Study

Owing to time limitations, further study is suggested as follows. First, the same study could be repeated with any other loss function, such as the linear exponential (LINEX) loss function; another different informative prior distribution might also be tried. Second, getting the ideal experimental structure for predetermined ideal total test time (T), the effective sample size (m), and the total sample size (n) would be interesting for further study. Third, it might be helpful to repeat this study with other estimations of the characteristics of the distribution, such as hazard function or cumulative hazard function, quantile function ...etc. and constructing other types of confidence intervals, such as the parametric bootstrap confidence interval, likelihood ratio-based confidence interval, and highest posterior density interval...etc. In addition, it might be useful to fit the real data set in the illustrative example with any life distribution model to compare the adequacy with other continuous life time distribution. Finally, the value of total test time(T) act an important base in determining the values of progressive censoring schemes (R) and intermediary between a shorter experimental time and a larger number of units used in the test. This indicates that the expected duration test time is affected by the value of the removal units; further studies should calculate the same, aiming to significantly reduce the experimental time taking in order the efficiency of statistical inference.

REFERENCES

AL Sobhi, M. M., & Soliman, A. A. (2016). Estimation for the exponentiated Weibull model with adaptive Type-II progressive censored schemes. *Applied Mathematical Modelling, 40*(2), 1180-1192.

Aldrich, J. (1997). R. A. Fisher and the Making of Maximun Likelihood 1912-1922. *Statistical Science* , 162-176.

AL-Haj Ebrahem, M., & Baklizi, A. (2005). Comparison of Simple Estimators of the Log-Logisitic Parameters Based on Complete and Censored Samples. *Journal of Statistical Theory and Applications*, *4*, 151-160.

Al-Shomrani, A. A., Shawky, A. l., Arif, O. H., & Aslam, M. (2016). Log-logistic distribution for survival data analysis using MCMC. *SpringerPlus*, *5*(1).

Ateya, S. F., & Mohammed, H. S. (2017). Statistical inferences based on an adaptive progresive type-II censoring from exponentiated exponential distribution. *Journal of the Egyptian Mathematical Society*, 393-399.

Bain, L. J. (1974). Analysis for the linear failure rate life-testing distribution. *Technometrics*, *16*, 551-559.

Balakrishnan, N. (1992). Handbook of the logistic distribution. New York:Dekker: CRC Press.

Balakrishnan, N. (2007). Progressive censoring methodology: an appraisal. *Test*, *16*(2), 287-289 .

Balakrishnan, N., & Cramer, E. (2014). *The Art of Progressive Censoring.* Statistics for Industry and Technology.

Balakrishnan, N., & Sandhu, R. (1995). A simple Simulational Algorithm for Generating Progressive Type-II Censored Samples. *American Statisticial Association , 49*, 229-230.

Casella, G., & Berger , R. L. (2002). *Statistical Inference.* Thomson Learning.

David, H. A., & Nagaraja, H. N. (2003). *Order Statistics* (3rd ed.). New York: Wiley.

Fisk, P. R. (1961). The graduation of income distributions. *Econometrica, 29*, 171-185.

Geyer, C. J. (2012). Markov Chain Monte Carlo and Bayesian Inference. *Journal of statistical* .

Greene, W. H. (2010). *Econometric analysis* (7th ed.). New York: Pearson.

Guure, C. B. (2015). Inference on the Loglogistic Model with Right Censored Data. *Austin Biom and Biostat, 2*(1), 1015.

Hamada, M. S., Wilson, A. G., & Reese, S. C. (2008). *Bayesian Relibility.* Springer Science+Business Media,LLC.

Hartig, F. (2011). MCMC chain analysis and convergence diagnostics with coda in R. *theoretical ecology*.

Kundu, D., & Joarder, A. (2006). Analysis of Type -II progressively hybrid censored data. *Computational Statistics and Data Analysis, 50*(10), 2509-2528.

Lawless, J. F. (1982). *Statistical Models and Methods for lifetime Data.* New York :Wiley: John Wiley & Sons, Inc.

Lehmann, E. L. (1998). *Theory of Point estimation* (2 ed.). New York.

Lemonte, A. J. (2014). The beta log-logistic distribution. *Brazilian Journal of Probability and Statistics, 28*(3), 313-332.

Linder, E., Seefeld, K., & Ed, M. (2015). Statistics Using R with Biological Examples. (January).

Mahmoud, M. A., Soliman, A. A., Abd Ellah, A. H., & El-Sagheer, R. M. (2013). Estimation of Generalized Pareto under an Adaptive Type-II Progressive Censoring. *Intelligent Information Management, 5*, 73-83.

Mohie El-Din, M. M., Amein, M. M., Shafay, A. R., & Mohamed, S. (2016). Estimation of generalized exponential distribution based on an adaptive progressively type-II censored sample. *Jornal of Statistical Computation and Simulation*, 1563-5163.

Ng, H. K., Kundu, D., & Chan, P. S. (2009). Statistical Analysis of Exponential Lifetimes under an adaptive Type-II Progressive Censoring Scheme. *Naval Research Logistics*, 687-698.

Nichols, M. D., & Padgett, W. J. (2006). A Bootstrap Control Chart for Weibull Percentiles. *Quality and Reliability Engineering International, 22*(2), 141-151.

Ragab, A., & Green, J. (1984). On order statistics from the Log-Logistic distribution and their properties. *Communication in statistics -Theory and Methods*, *21*, pp. 2713-2724.

Raykundaliya, D. P. (2017). Modified maximum likelihood estimation of parameters in the log-logistic distribution under progressive Type II censored data with binomial removals. *International Journal of Engineering,Pure and Applied Sciences*, *2*, 1-6.

APPENDIX

APPENDIX A: QUANTILE FUNCTION OF LOG-LOGISTIC DISTRIBUTION

$$F(t) = \frac{1}{1 + e^{\frac{-\pi(\ln t - \mu)}{\sigma\sqrt{3}}}}$$

$$u = \frac{1}{1 + e^{\frac{-\pi(\ln t - \mu)}{\sigma\sqrt{3}}}}$$

$$1 + e^{\frac{-\pi(\ln t - \mu)}{\sigma\sqrt{3}}} = \frac{1}{u}$$

$$\frac{-\pi(\ln t - \mu)}{\sigma\sqrt{3}} = \ln(\frac{1}{u} - 1)$$

$$\ln t - \mu = \frac{\sigma\sqrt{3}}{-\pi}\ln(\frac{1}{u} - 1)$$

$$\ln t - \mu = \frac{\sigma\sqrt{3}}{-\pi}\ln(\frac{1}{u} - 1)$$

$$\ln t = \frac{\sigma\sqrt{3}}{-\pi}\ln(\frac{1}{u} - 1) + \mu$$

$$t = F^{-1}(u) = e^{\frac{\sigma\sqrt{3}}{-\pi}\ln\left(\frac{1}{u} - 1\right) + \mu}$$

$$t = F^{-1}(u) = (\frac{1}{u} - 1)^{\frac{\sigma\sqrt{3}}{-\pi}} e^{\mu}$$

APPENDIX B: R CODE FOR A MONTE CARLO SIMULATION STUDY

```
# Convergence Diagnostic for n=50, m=30 & T=1 .

nsim=11000
muhat=c()
sigmhat=c()
mmuhat=rep(0,nBoot)
ssigmhat=rep(0,nBoot)
Low1=rep(0,nBoot)
Upper1=rep(0,nBoot)
Exact1=rep(0,nBoot)
Low2=rep(0,nBoot)
Upper2=rep(0,nBoot)
Exact2=rep(0,nBoot)
S11=rep(0,nBoot)
S22=rep(0,nBoot)
S33=rep(0,nBoot)
LowS1=rep(0,nBoot)
UpperS1=rep(0,nBoot)
LowS2=rep(0,nBoot)
UpperS2=rep(0,nBoot)
LowS3=rep(0,nBoot)
UpperS3=rep(0,nBoot)
ExactS1=rep(0,nBoot)
ExactS2=rep(0,nBoot)
ExactS3=rep(0,nBoot)
count1=0;count2=0;countS1=0;countS2=0;countS3=0
T=1.8
n=50
m=30
mu=0
sigm=1
pi=3.14
t1=0.5
t2=1
t3=2
w=c()
w=runif(m)     #step 1
w
r=rep(0,m)            # empty vector
#progressive censored scheme
p=rep(0,m)
```

```
v=rep(0,m)
x=c()
u=rep(0,m)
y=rep(0,m)
r=c(20,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
length(r)
p=r[m]
p
for(i in 1:m)
{
  v[i]=w[i]^(1/(i+p))
  p=p+r[m-i]
}
x=v[m]                  #step2
x

for(i in 1:m){        #step 3
  u[i]=1-x
  x=x*v[m-i]
}

u      # progressive Type II censored sample from the uniform.

# The required progressive Type-II censored sample from the quantile function of log-
logistic distribution is
for(i in 1:m){
    y[i]= exp(mu)*(1/u[i]-1)^(-sigm*sqrt(3)/pi)
}
y     # the required sample from the quantile function of log-logistic distribution
d=c()

for(i in 1:m ){

  if(y[i]<T){
    d[i]=y[i]

  } else {d[i]==0 }
}

d

j=length(d)           # Determine the value of j (j is the number of failure observation before
time T)
j
```

```
if(j+1<m){

  rs=c()

  for(i in 1:j)
  {
   rs[i]=r[i]
  }
  rs                        #progressive censored scheme for J

  R=sum(rs)
  R
  #using invers transformation method to get sample from truncated distribution.
  g=c()
  U=c()
  a=y[j+1]
   aa=1/(1+a^(-pi/(sigm*sqrt(3)))*exp(mu*pi/(sigm*sqrt(3))))  #CDF for Log logistic
dist. F(a)=1/(1+exp(-pi*log(a)+mu*pi)/sigm*sqrt(3))
  bb=1
  U=runif(n-R-j-1,aa,bb)
  U
  A=sort(U)
   for(i in 1:n-R-j-1){
       g[i]=exp(mu)*(1/A[i]-1)^(-sigm*sqrt(3)/pi)
   # quantile function with specific range instead of U(0,1)to generated r.v
  }
  g
  f=c()              #the first order statistics m-j-1
  for(i in 1:m-j-1)
  {
   f[i]=g[i]
  }
    ds=c()
   for(i in 1:j){
   ds[i]=y[i]
     }
  ds
   dss=c()
   for(i in 0:j+1){
   dss[i]=y[i]
     }
  dss
  D=c()
  D=c(dss,f)  # use function combine to add the two vectors together
  D           #The adaptive progressive type II censored data according to the algorithm
```

```
 dm=D[m]        # last value in  an adaptive progressive type II censored data
 dm
 RR=(n-m-sum(rs))     #or    R=sum(rs)
 RR
  da=D
 da
  #MCMC(MH): Bayesian method
 muhat[1]=mu
 sigmhat[1]=sigm
   posterior=function(muu,sigmm){
   exp(m*log(pi)-(m*log(sigmm)+sum(log(da))+m*log(sqrt(3)))-
sum(log(da))*pi/(sigmm*sqrt(3))+(m*muu*pi)/(sigmm*sqrt(3))-2*sum(log(1+exp((-
pi*log(da)+pi*muu)/(sigmm*sqrt(3)))))-pi/(sigmm*sqrt(3))*sum(rs*(log(ds)-muu))-
sum(rs*log(1+exp((-pi*log(ds)+pi*muu)/(sigmm*sqrt(3)))))+RR*(-
pi*log(dm)+muu*pi)/(sigmm*sqrt(3))-RR*log(1+exp((-
pi*log(dm)+pi*muu)/(sigmm*sqrt(3))))-log(sigmm))
  }

 dcand = function(muu,sigmm) {
   dlogis(muu,mu,1)*dgamma(1/sigmm,sigm^2+2,rate=sigm*(sigm^2+1))/sigmm^2
  }

 for (i in 2:nsim) {
   cand = c(rlogis(1,muhat[i-1],1),1/rgamma(1,sigmhat[i-1]^2+2,rate=(sigmhat[i-
1]^2+1))) #generate samlpe from candidate or proposal (not prior)  ,need to install
package of inverse gamma

   accep = min(posterior(cand[1], cand[2])*dcand(muhat[i-1],sigmhat[i-
1])/(posterior(muhat[i-1], sigmhat[i-1])*dcand(cand[1],cand[2])), 1)

   dad=runif(1)

   rho = (dad < accep)
   muhat[i] = cand[1] * rho + muhat[i - 1] * (1 - rho)
   sigmhat[i] = cand[2] * rho + sigmhat[i - 1] * (1 - rho)

  }

}else{if((j==m)||(j+1==m)){
 da=y
 dm=y[m]        # last value in an adaptive progressive type II censored data
 dm
 rs=c()

 for(i in 1:j)
```

```
 {
  rs[i]=r[i]
 }
 rs #progressive censored scheme for J

 R=sum(rs)
 R
 RR=(n-m-sum(rs))     #or    R=sum(rs)
 RR
 ds=c()

 for(i in 1:j){
  ds[i]=y[i]

 }
 ds


 #MCMC(MH) : Bayesian method

 muhat[1]=mu
 sigmhat[1]=sigm
  posterior=function(muu,sigmm){
   exp(m*log(pi)-(m*log(sigmm)+sum(log(da))+m*log(sqrt(3)))-
sum(log(da))*pi/(sigmm*sqrt(3))+(m*muu*pi)/(sigmm*sqrt(3))-2*sum(log(1+exp((-
pi*log(da)+pi*muu)/(sigmm*sqrt(3)))))-pi/(sigmm*sqrt(3))*sum(rs*(log(ds)-muu))-
sum(rs*log(1+exp((-pi*log(ds)+pi*muu)/(sigmm*sqrt(3)))))+RR*(-
pi*log(dm)+muu*pi)/(sigmm*sqrt(3))-RR*log(1+exp((-
pi*log(dm)+pi*muu)/(sigmm*sqrt(3))))-log(sigmm))
  }

 dcand = function(muu,sigmm) {
   dlogis(muu,mu,1)*dgamma(1/sigmm,sigm^2+2,rate=sigm*(sigm^2+1))/sigmm^2

 }
 for (i in 2:nsim) {
   cand = c(rlogis(1,muhat[i-1],1),1/rgamma(1,sigmhat[i-1]^2+2,rate=(sigmhat[i-
1]^2+1))) #generate samlpe from candidate or proposal (not prior)  ,need to install
package of inverse gamma

   accep = min(posterior(cand[1], cand[2])*dcand(muhat[i-1],sigmhat[i-
1])/(posterior(muhat[i-1], sigmhat[i-1])*dcand(cand[1],cand[2])), 1)

   dad=runif(1)
```

```
    rho = (dad < accep)
    muhat[i] = cand[1] * rho + muhat[i - 1] * (1 - rho)
    sigmhat[i] = cand[2] * rho + sigmhat[i - 1] * (1 - rho)

  }


}
}
muhat
sigmhat
###########use coda package
cm=as.mcmc(muhat)
cs=as.mcmc(sigmhat)
plot(cm,main="µ")
plot(cs,main="sigma")
summary(cm)
autocorr(cm)# indicate at what lag the autocorrelation decrease
autocorr(cs)
autocorr.plot(cm,main="µ")  #plot the autocorrelation
autocorr.plot(cs,main="sigma")

#####R code to draw again the marginal density plot with correct bandwidth and with
burn in
M=1000
muhat=muhat[-1:-M]
muhat
sigmhat=sigmhat[-1:-M]
sigmhat
###############################use R code after burn-in
i=seq(1,length(muhat),by=50)
out.sub=density(muhat[i])
out=density(muhat,bw=out.sub$bw)
plot(out,main=" µ")
ii=seq(1,length(sigmhat),by=50)
out.sub=density(sigmhat[ii])
out=density(sigmhat,bw=out.sub$bw)
plot(out,main="sigma'')
##################################### hist after burn-in
hist(muhat,main="µ",freq = FALSE)
hist(sigmhat,main="sigma",freq = FALSE)
#######################################################################

# A Monte Carlo Simulation Study for n=50,m=30 & T=1 based on MLE method .
#ML case 1
```

```
nBoot=2000
nn=array(0,dim=c(nBoot,1))
mu1=array(0,dim=c(nBoot,1))
sigm1=array(0,dim=c(nBoot,1))
S1=array(0,dim=c(nBoot,1))
S2=array(0,dim=c(nBoot,1))
S3=array(0,dim=c(nBoot,1))
var1=array(0,dim=c(nBoot,1))
var2=array(0,dim=c(nBoot,1))
Low1=array(0,dim=c(nBoot,1))
Low2=array(0,dim=c(nBoot,1))
Upp1=array(0,dim=c(nBoot,1))
Upp2=array(0,dim=c(nBoot,1))
ExactL1=array(0,dim=c(nBoot,1))
ExactL2=array(0,dim=c(nBoot,1))
count1=0;count2=0;countS1=0;countS2=0;countS3=0
varS1=array(0,dim=c(nBoot,1))
varS1e=array(0,dim=c(nBoot,1))
LowS1=array(0,dim=c(nBoot,1))
UppS1=array(0,dim=c(nBoot,1))
varS2=array(0,dim=c(nBoot,1))
varS2e=array(0,dim=c(nBoot,1))
LowS2=array(0,dim=c(nBoot,1))
UppS2=array(0,dim=c(nBoot,1))
varS3=array(0,dim=c(nBoot,1))
varS3e=array(0,dim=c(nBoot,1))
LowS3=array(0,dim=c(nBoot,1))
UppS3=array(0,dim=c(nBoot,1))
ExactLS1=array(0,dim=c(nBoot,1))
ExactLS2=array(0,dim=c(nBoot,1))
ExactLS3=array(0,dim=c(nBoot,1))
for(ii in 1:nBoot){
  T=1
  n=50
  m=30
  mu=0
  sigm=1
  pi=3.14
  t1=0.5
  t2=1
  t3=2
  w=c()
  w=runif(m)      #step 1
  w
  r=rep(0,m)              # empty vector
```

```
 p=rep(0,m)
 v=rep(0,m)
 x=c()
 u=rep(0,m)
 y=rep(0,m)
 r=c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,20)
 p=r[m]
 p
 for(i in 1:m)
 {
   v[i]=w[i]^(1/(i+p))
   p=p+r[m-i]
 }
 x=v[m]                 #step2
 x

 for(i in 1:m){        #step 3
   u[i]=1-x
   x=x*v[m-i]
 }

 u      # progressive type II censored sample from the uniform.

 for(i in 1:m){# exp(mu)*(1/u[i]-1)^(-sigm*sqrt(3)/pi)# y[i]=exp((sigm*sqrt(3)/-
pi)*log(1/u[i]-1)+mu)
   y[i]= exp(mu)*(1/u[i]-1)^(-sigm*sqrt(3)/pi)
 }
 y     # the required sample from the quantile function of log-logestic distribution
 d=c()
 for(i in 1:m ){

   if(y[i]<T){
     d[i]=y[i]

   } else {d[i]==0 }
 }
  d
   j=length(d)     #Determine the value of j (j is the number of failure observation before
time T)
 j

 if(j+1<m){

   rs=c()
```

```
for(i in 1:j)
{
  rs[i]=r[i]
}
rs #progressive censored scheme for J

R=sum(rs)
R
#using invers transformation method to get sample from truncated distribution.
g=c()
U=c()
a=y[j+1]
aa=1/(1+a^(-pi/(sigm*sqrt(3)))*exp(mu*pi/(sigm*sqrt(3)))) #CDF for Log logestic dist
F(a)=1/(1+exp(-pi*log(a)+mu*pi)/sigm*sqrt(3))
bb=1
U=runif(n-R-j-1,aa,bb)
U
A=sort(U)

for(i in 1:n-R-j-1){
        g[i]=exp(mu)*(1/A[i]-1)^(-sigm*sqrt(3)/pi)
  # quntile function with spcific range instead of U(0,1)to generated r.v from uniform
}
g
f=c()#the first order statistics m-j-1
for(i in 1:m-j-1)
{
  f[i]=g[i]
}
ds=c()
for(i in 1:j){
  ds[i]=y[i]
      }
ds
dss=c()
 for(i in 0:j+1){
  dss[i]=y[i]
   }
dss
D=c()
D=c(dss,f)  # use function combine to add the two vectors together
D          #The adaptive progressive type II censored data according to the both
algorithms
dm=D[m]        # last value in the adaptive progressive type II censored data)
dm
```

```
RR=(n-m-sum(rs))     #or    R=sum(rs)
RR
da=D
da
like=function(b)
{-(m*log(pi)-(m*log(b[2])+sum(log(da))+m*log(sqrt(3)))-
sum(log(da))*pi/(b[2]*sqrt(3))+(m*b[1]*pi)/(b[2]*sqrt(3))-2*sum(log(1+exp((-
pi*log(da)+pi*b[1])/(b[2]*sqrt(3)))))-pi/(b[2]*sqrt(3))*sum(rs*(log(ds)-b[1]))-
sum(rs*log(1+exp((-pi*log(ds)+pi*b[1])/(b[2]*sqrt(3)))))+RR*(-
pi*log(dm)+b[1]*pi)/(b[2]*sqrt(3))-RR*log(1+exp((-
pi*log(dm)+pi*b[1])/(b[2]*sqrt(3))))))}

nn=nlm(like,c(mu,sigm),hessian=TRUE)

mu1[ii]=nn$estimate[1]
sigm1[ii]=nn$estimate[2]

S1[ii]=1-1/(1+exp(-pi*(log(t1)-mu1[ii])/(sigm1[ii]*sqrt(3))))

S2[ii]=1-1/(1+exp(-pi*(log(t2)-mu1[ii])/(sigm1[ii]*sqrt(3))))

S3[ii]=1-1/(1+exp(-pi*(log(t3)-mu1[ii])/(sigm1[ii]*sqrt(3))))

#95% (alpha=0.05)CI (100(1-alpha)% two sided approx.CI for the parameters mu &
sigma)or qnorm(alpha/2)=qnorm(0.025)

inv=solve(nn$hessian)
var1[ii]=inv[1]# mu
var1
var2[ii]=inv[4]#sigma
var2
Low1[ii]=mu1[ii]-qnorm(0.975)*sqrt(var1[ii])

Upp1[ii]=mu1[ii]+qnorm(0.975)*sqrt(var1[ii])

Low2[ii]=sigm1[ii]-qnorm(0.975)*sqrt(var2[ii])

Upp2[ii]=sigm1[ii]+qnorm(0.975)*sqrt(var2[ii])
###############CI for survival function S1(0.5)
G1=c(((pi/(sigm1[ii]*sqrt(3)))*exp(-pi*(log(t1)-mu1[ii])/(sigm1[ii]*sqrt(3))))/(1+exp(-
pi*(log(t1)-mu1[ii])/(sigm1[ii]*sqrt(3))))^2,((pi*(log(t1)-
mu1[ii])/((sigm1[ii])^2*sqrt(3)))*exp(-pi*(log(t1)-mu1[ii])/(sigm1[ii]*sqrt(3))))/(1+exp(-
pi*(log(t1)-mu1[ii])/(sigm1[ii]*sqrt(3))))^2)
G11=rbind(G1)
varS1=G11%*%inv
```

```
varS1e[ii]=varS1%*%t(G11)

LowS1[ii]=S1[ii]-qnorm(0.975)*sqrt(varS1e[ii])

UppS1[ii]=S1[ii]+qnorm(0.975)*sqrt(varS1e[ii])
###############CI for survival function S2(1)
G2=c(((pi/(sigm1[ii]*sqrt(3)))*exp(-pi*(log(t2)-mu1[ii])/(sigm1[ii]*sqrt(3))))/(1+exp(-
pi*(log(t2)-mu1[ii])/(sigm1[ii]*sqrt(3))))^2,((pi*(log(t2)-
mu1[ii])/((sigm1[ii])^2*sqrt(3)))*exp(-pi*(log(t2)-mu1[ii])/(sigm1[ii]*sqrt(3))))/(1+exp(-
pi*(log(t2)-mu1[ii])/(sigm1[ii]*sqrt(3))))^2)
G22=rbind(G2)
varS2=G22%*%inv

varS2e[ii]=varS2%*%t(G22)

LowS2[ii]=S2[ii]-qnorm(0.975)*sqrt(varS2e[ii])

UppS2[ii]=S2[ii]+qnorm(0.975)*sqrt(varS2e[ii])
############################
G3= G1=c(((pi/(sigm1[ii]*sqrt(3)))*exp(-pi*(log(t3)-
mu1[ii])/(sigm1[ii]*sqrt(3))))/(1+exp(-pi*(log(t3)-
mu1[ii])/(sigm1[ii]*sqrt(3))))^2,((pi*(log(t3)-mu1[ii])/((sigm1[ii])^2*sqrt(3)))*exp(-
pi*(log(t3)-mu1[ii])/(sigm1[ii]*sqrt(3))))/(1+exp(-pi*(log(t3)-
mu1[ii])/(sigm1[ii]*sqrt(3))))^2)
G33=rbind(G3)
varS3=G33%*%inv

varS3e[ii]=varS3%*%t(G33)

LowS3[ii]=S3[ii]-qnorm(0.975)*sqrt(varS3e[ii])

UppS3[ii]=S3[ii]+qnorm(0.975)*sqrt(varS3e[ii])
###############################
ExactL1[ii]=Upp1[ii]-Low1[ii]#Average length (AL) is the summation of all length of
confdence intervals divided by total number of iteractons

ExactL2[ii]=Upp2[ii]-Low2[ii]
###################
ExactLS1[ii]=UppS1[ii]-LowS1[ii]#Average length (AL) is the summation of all
length of confidence intervals divided by total number of iterations

ExactLS2[ii]=UppS2[ii]-LowS2[ii]

ExactLS3[ii]=UppS3[ii]-LowS3[ii]
```

```
 }else{if ((j==m)||(j+1==m)){
  da=y
  dm=y[m]        # last value in the adaptive progressive type II censored data)
  dm
  rs=c()

  for(i in 1:j)
  {
   rs[i]=r[i]
  }
  rs #progressive censored scheme for J

  R=sum(rs)
  R
  RR=(n-m-sum(rs))     #or    R=sum(rs)
  RR
  ds=c()

  for(i in 1:j){
   ds[i]=y[i]

  }
  ds

  like=function(b)
  {-(m*log(pi)-(m*log(b[2])+sum(log(da))+m*log(sqrt(3)))-
sum(log(da))*pi/(b[2]*sqrt(3))+(m*b[1]*pi)/(b[2]*sqrt(3))-2*sum(log(1+exp((-
pi*log(da)+pi*b[1])/(b[2]*sqrt(3)))))-pi/(b[2]*sqrt(3))*sum(rs*(log(ds)-b[1]))-
sum(rs*log(1+exp((-pi*log(ds)+pi*b[1])/(b[2]*sqrt(3)))))+RR*(-
pi*log(dm)+b[1]*pi)/(b[2]*sqrt(3))-RR*log(1+exp((-
pi*log(dm)+pi*b[1])/(b[2]*sqrt(3))))))}

  nn=nlm(like,c(mu,sigm),hessian=TRUE)

  mu1[ii]=nn$estimate[1]
  sigm1[ii]=nn$estimate[2]

  #no need to store matrix because outside the loop print the last hessian of nBoot=2000

  S1[ii]=1-1/(1+exp(-pi*(log(t1)-mu1[ii])/(sigm1[ii]*sqrt(3))))

  S2[ii]=1-1/(1+exp(-pi*(log(t2)-mu1[ii])/(sigm1[ii]*sqrt(3))))

  S3[ii]=1-1/(1+exp(-pi*(log(t3)-mu1[ii])/(sigm1[ii]*sqrt(3))))
```

```
#95% (alpha=0.05)CI (100(1-alpha)% two sided approx.CI for the parameters mu & sigma)or qnorm(alpha/2)=qnorm(0.025)

inv=solve(nn$hessian)

var1[ii]=inv[1]# mu
var1
var2[ii]=inv[4]#sigma
var2
Low1[ii]=mu1[ii]-qnorm(0.975)*sqrt(var1[ii])

Upp1[ii]=mu1[ii]+qnorm(0.975)*sqrt(var1[ii])

Low2[ii]=sigm1[ii]-qnorm(0.975)*sqrt(var2[ii])

Upp2[ii]=sigm1[ii]+qnorm(0.975)*sqrt(var2[ii])
###############CI for survival function S1(0.5)
G1=c(((pi/(sigm1[ii]*sqrt(3)))*exp(-pi*(log(t1)-mu1[ii])/(sigm1[ii]*sqrt(3)))))/(1+exp(-pi*(log(t1)-mu1[ii])/(sigm1[ii]*sqrt(3)))))^2,((pi*(log(t1)-mu1[ii])/((sigm1[ii])^2*sqrt(3)))*exp(-pi*(log(t1)-mu1[ii])/(sigm1[ii]*sqrt(3)))))/(1+exp(-pi*(log(t1)-mu1[ii])/(sigm1[ii]*sqrt(3)))))^2)
G11=rbind(G1)
varS1=G11%*%inv

varS1e[ii]=varS1%*%t(G11)

LowS1[ii]=S1[ii]-qnorm(0.975)*sqrt(varS1e[ii])

UppS1[ii]=S1[ii]+qnorm(0.975)*sqrt(varS1e[ii])
###############CI for survival function S2(1)
G2=c(((pi/(sigm1[ii]*sqrt(3)))*exp(-pi*(log(t2)-mu1[ii])/(sigm1[ii]*sqrt(3)))))/(1+exp(-pi*(log(t2)-mu1[ii])/(sigm1[ii]*sqrt(3)))))^2,((pi*(log(t2)-mu1[ii])/((sigm1[ii])^2*sqrt(3)))*exp(-pi*(log(t2)-mu1[ii])/(sigm1[ii]*sqrt(3)))))/(1+exp(-pi*(log(t2)-mu1[ii])/(sigm1[ii]*sqrt(3)))))^2)
G22=rbind(G2)
varS2=G22%*%inv

varS2e[ii]=varS2%*%t(G22)

LowS2[ii]=S2[ii]-qnorm(0.975)*sqrt(varS2e[ii])

UppS2[ii]=S2[ii]+qnorm(0.975)*sqrt(varS2e[ii])
###########################
G3= G1=c( ((pi/(sigm1[ii]*sqrt(3)))*exp(-pi*(log(t3)-mu1[ii])/(sigm1[ii]*sqrt(3)))))/(1+exp(-pi*(log(t3)-
```

```
mu1[ii])/(sigm1[ii]*sqrt(3))))^2,((pi*(log(t3)-mu1[ii])/((sigm1[ii])^2*sqrt(3)))*exp(-
pi*(log(t3)-mu1[ii])/(sigm1[ii]*sqrt(3))))/(1+exp(-pi*(log(t3)-
mu1[ii])/(sigm1[ii]*sqrt(3))))^2)
    G33=rbind(G3)
    varS3=G33%*%inv

    varS3e[ii]=varS3%*%t(G33)

    LowS3[ii]=S3[ii]-qnorm(0.975)*sqrt(varS3e[ii])

    UppS3[ii]=S3[ii]+qnorm(0.975)*sqrt(varS3e[ii])
    ##############################
    ExactL1[ii]=Upp1[ii]-Low1[ii]
    ExactL2[ii]=Upp2[ii]-Low2[ii]
    ###################
    ExactLS1[ii]=UppS1[ii]-LowS1[ii]
    ExactLS2[ii]=UppS2[ii]-LowS2[ii]

    ExactLS3[ii]=UppS3[ii]-LowS3[ii]
  }
  }
}#end of loop

######################
#CII for mu
CI1=c(mean(Low1),mean(Upp1))
CI1
#CII for sigm
CI2=c(mean(Low2),mean(Upp2))
CI2
###################
AL1=sum(ExactL1)/nBoot
AL1
AL2=sum(ExactL2)/nBoot
AL2
#######################################################
#coverage prob. for mu
for(i in 1:nBoot){

  if(Low1[i]<=mu & Upp1[i]>=mu){
    count1=count1+1}

}
count1
prop1=sum(count1)/nBoot
```

```
prop1
################coverage prob. for sigma
for(i in 1:nBoot){

  if(Low2[i]<=sigm & Upp2[i]>=sigm)
    count2=count2+1

}
count2
prop2=sum(count2)/nBoot
prop2
###################################Estimate of parameters
mu11=mean(mu1)
mu11
sigm11=mean(sigm1)
sigm11
S11=mean(S1)
S11
S22=mean(S2)
S22
S33=mean(S3)
S33
#compute /bias
Bm=mu11-mu  #or Bm=sum(mu1-mu)/nBoot
Bm
Bs=sigm11-sigm
Bs
S01= 1-1/(1+exp(-pi*(log(t1)-mu)/(sigm*sqrt(3))))
S02=1-1/(1+exp(-pi*(log(t2)-mu)/(sigm*sqrt(3))))
S03=1-1/(1+exp(-pi*(log(t3)-mu)/(sigm*sqrt(3))))
BS1=S11-S01
BS1
BS2=S22-S02
BS2
BS3=S33-S03
BS3
########################(MSE)
MSEm=sum((mu1-mu)^2)/nBoot
MSEm
MSEs=sum((sigm1-sigm)^2)/nBoot
MSEs
MSES1=sum((S1-S01)^2)/nBoot
MSES1
MSES2=sum((S2-S02)^2)/nBoot
MSES2
```

```
MSES3=sum((S3-S03)^2)/nBoot
MSES3
#approximate C.I for Survival function
#C.I for s1,t=0.5
CIS1=c(mean(LowS1),mean(UppS1))
CIS1
#approximate C.I for Survival function
#C.I for s2,t=1

CIS2=c(mean(LowS2),mean(UppS2))
CIS2
#approximate C.I for Survival function
#C.I for s3,t=2

CIS3=c(mean(LowS3),mean(UppS3))
CIS3
######################
#average length #the shorter average length is the better performance of the confidence
interval
ALS1=sum(ExactLS1)/nBoot
ALS1
ALS2=sum(ExactLS2)/nBoot
ALS2
ALS3=sum(ExactLS3)/nBoot
ALS3
########################################################
#covarge prob.for S1
for(i in 1:nBoot){

  if(LowS1[i]<=S01 & UppS1[i]>=S01)
    countS1=countS1+1

}
countS1
propS1=sum(countS1)/nBoot
propS1
################coverage prob. for S2
for(i in 1:nBoot){

  if(LowS2[i]<=S02 & UppS2[i]>=S02)
    countS2=countS2+1

}
countS2
propS2=sum(countS2)/nBoot
```

```
propS2

################coverage prob. for S3
for(i in 1:nBoot){

  if(LowS3[i]<=S03 & UppS3[i]>=S03)
    countS3=countS3+1

}
countS3
propS3=sum(countS3)/nBoot
propS3

# A Monte Carlo Simulation Study for n= 50, m=30 & T=1 based on MCMC method by
using non-informative priors.
#case 1
nBoot=2000
nsim=11000
muhat=c()
sigmhat=c()
mmuhat=rep(0,nBoot)
ssigmhat=rep(0,nBoot)
Low1=rep(0,nBoot)
Upper1=rep(0,nBoot)
Exact1=rep(0,nBoot)
Low2=rep(0,nBoot)
Upper2=rep(0,nBoot)
Exact2=rep(0,nBoot)
S11=rep(0,nBoot)
S22=rep(0,nBoot)
S33=rep(0,nBoot)
LowS1=rep(0,nBoot)
UpperS1=rep(0,nBoot)
LowS2=rep(0,nBoot)
UpperS2=rep(0,nBoot)
LowS3=rep(0,nBoot)
UpperS3=rep(0,nBoot)
ExactS1=rep(0,nBoot)
ExactS2=rep(0,nBoot)
ExactS3=rep(0,nBoot)
count1=0;count2=0;countS1=0;countS2=0;countS3=0
for(ii in 1:nBoot){
  print(ii)
  T=1
  n=50
```

```
m=30
mu=0
sigm=1
pi=3.14
t1=0.5
t2=1
t3=2
w=c()
w=runif(m)      #step 1
w
r=rep(0,m)              # empty vector
 #progressive censored scheme
p=rep(0,m)
v=rep(0,m)
x=c()
u=rep(0,m)
y=rep(0,m)
r=c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,20)
length(r)
p=r[m]
p
for(i in 1:m)
{
  v[i]=w[i]^(1/(i+p))
  p=p+r[m-i]
}
x=v[m]                  #step2
x

for(i in 1:m){        #step 3
  u[i]=1-x
  x=x*v[m-i]
}

u      # progressive type II censored sample from the uniform.

for(i in 1:m){
y[i]= exp(mu)*(1/u[i]-1)^(-sigm*sqrt(3)/pi)
}
y     # the requried sample from the quantile function of log-logestic distribution
d=c()

for(i in 1:m ){

  if(y[i]<T){
```

```
    d[i]=y[i]

  } else {d[i]==0 }
 }

 d

 j=length(d)        #Determine the value of j (j is the number of failure observation before
time T)
 j

 ###################(Now generate adaptive progressive Type II censored sample )
 # step B
 if(j+1<m){


   rs=c()

   for(i in 1:j)
   {
    rs[i]=r[i]
   }
   rs #progressive censored scheme for J

   R=sum(rs)
   R
   #using invers transformation method to get sample from truncated distribution.
   g=c()
   U=c()
   a=y[j+1]
  aa=1/(1+a^(-pi/(sigm*sqrt(3)))*exp(mu*pi/(sigm*sqrt(3))))
   bb=1
   U=runif(n-R-j-1,aa,bb)
   U
   A=sort(U)

   for(i in 1:n-R-j-1){

    g[i]=exp(mu)*(1/A[i]-1)^(-sigm*sqrt(3)/pi)
    # quantile function with specific range instead of U(0,1)
}
   # ( g is the generated sample from )
   g
   ###########################################
   f=c()#the first order statistics m-j-1
```

```
for(i in 1:m-j-1)
{
  f[i]=g[i]
}


ds=c()

for(i in 1:j){
  ds[i]=y[i]

}
ds

dss=c()

for(i in 0:j+1){
  dss[i]=y[i]

}
dss

D=c()
D=c(dss,f)  # use function combine to add the two vectors together
D
##############################################
dm=D[m]        # last value in the adaptive progressive type II censored data)
dm

RR=(n-m-sum(rs))     #or    R=sum(rs)
RR
da=D
da

#MCMC(MH):Bayeian method
muhat[1]=mu
sigmhat[1]=sigm
posterior=function(muu,sigmm){
  exp(m*log(pi)-(m*log(sigmm)+sum(log(da))+m*log(sqrt(3)))-
sum(log(da))*pi/(sigmm*sqrt(3))+(m*muu*pi)/(sigmm*sqrt(3))-2*sum(log(1+exp((-
pi*log(da)+pi*muu)/(sigmm*sqrt(3)))))-pi/(sigmm*sqrt(3))*sum(rs*(log(ds)-muu))-
sum(rs*log(1+exp((-pi*log(ds)+pi*muu)/(sigmm*sqrt(3)))))+RR*(-
pi*log(dm)+muu*pi)/(sigmm*sqrt(3))-RR*log(1+exp((-
pi*log(dm)+pi*muu)/(sigmm*sqrt(3))))-log(sigmm))
}
```

```
dcand = function(muu,sigmm) {
  dlogis(muu,mu,1)*dgamma(1/sigmm,sigm^2+2,rate=sigm*(sigm^2+1))/sigmm^2
}

for (i in 2:nsim) {
  cand = c(rlogis(1,muhat[i-1],1),1/rgamma(1,sigmhat[i-1]^2+2,rate=(sigmhat[i-1]^2+1))) #generate samlpe from candidate or proposal (not prior) ,need to install package of inverse gamma

  accep = min(posterior(cand[1], cand[2])*dcand(muhat[i-1],sigmhat[i-1])/(posterior(muhat[i-1], sigmhat[i-1])*dcand(cand[1],cand[2])), 1)

  dad=runif(1)

  rho = (dad < accep)
  muhat[i] = cand[1] * rho + muhat[i - 1] * (1 - rho)
  sigmhat[i] = cand[2] * rho + sigmhat[i - 1] * (1 - rho)


}

n=1000
muhat=muhat[-1:-n]
mmuhat[ii]=mean(muhat)
sigmhat=sigmhat[-1:-n]
sigmhat
ssigmhat[ii]=mean(sigmhat)
#credible interval for mu
Sr1=sort(muhat)
Low1[ii]=Sr1[0.025*(nsim-n)]
Upper1[ii]=Sr1[0.975*(nsim-n)]
Exact1[ii]=Upper1[ii]-Low1[ii]
#credible interval for sigm
Sr2=sort(sigmhat)
Low2[ii]=Sr2[0.025*(nsim-n)]
Upper2[ii]=Sr2[0.975*(nsim-n)]
#EL for mu
Exact2[ii]=Upper2[ii]-Low2[ii]
#################################
t1=0.5
S1=1-1/(1+exp(-pi*(log(t1)-muhat)/(sigmhat*sqrt(3))))
S11[ii]=mean(S1)
t2=1
S2=1-1/(1+exp(-pi*(log(t2)-muhat)/(sigmhat*sqrt(3))))
```

```
S22[ii]=mean(S2)
t3=2
S3=1-1/(1+exp(-pi*(log(t3)-muhat)/(sigmhat*sqrt(3))))
S33[ii]=mean(S3)
#################################
#Credible interval for S1
Srs1=sort(S1)
LowS1[ii]=Srs1[0.025*(nsim-n)]
UpperS1[ii]=Srs1[0.975*(nsim-n)]
#EL for S1
ExactS1[ii]=UpperS1[ii]-LowS1[ii]
#Credible interval for S2
Srs2=sort(S2)
LowS2[ii]=Srs2[0.025*(nsim-n)]
LowS2
UpperS2[ii]=Srs2[0.975*(nsim-n)]
UpperS2
#EL for S2
ExactS2[ii]=UpperS2[ii]-LowS2[ii]
#Credible interval for S3# 95% CI
Srs3=sort(S3)
LowS3[ii]=Srs3[0.025*(nsim-n)]
LowS3
UpperS3[ii]=Srs3[0.975*(nsim-n)]
UpperS3
#EL for S3
ExactS3[ii]=UpperS3[ii]-LowS3[ii]
  }else{if((j==m)||(j+1==m)){
da=y
dm=y[m]        # last value in the adaptive progressive type II censored data)
dm
rs=c()

for(i in 1:j)
{
  rs[i]=r[i]
}
rs #progressive censored scheme for J

R=sum(rs)
R
RR=(n-m-sum(rs))     #or    R=sum(rs)
RR
ds=c()
```

```
for(i in 1:j){
  ds[i]=y[i]

}
ds

#MCMC(MH): Bayesian method

muhat[1]=mu
sigmhat[1]=sigm
posterior=function(muu,sigmm){
  exp(m*log(pi)-(m*log(sigmm)+sum(log(da))+m*log(sqrt(3)))-
sum(log(da))*pi/(sigmm*sqrt(3))+(m*muu*pi)/(sigmm*sqrt(3))-2*sum(log(1+exp((-
pi*log(da)+pi*muu)/(sigmm*sqrt(3)))))-pi/(sigmm*sqrt(3))*sum(rs*(log(ds)-muu))-
sum(rs*log(1+exp((-pi*log(ds)+pi*muu)/(sigmm*sqrt(3)))))+RR*(-
pi*log(dm)+muu*pi)/(sigmm*sqrt(3))-RR*log(1+exp((-
pi*log(dm)+pi*muu)/(sigmm*sqrt(3))))-log(sigmm))
}

dcand = function(muu,sigmm) {
  dlogis(muu,mu,1)*dgamma(1/sigmm,sigm^2+2,rate=sigm*(sigm^2+1))/sigmm^2

}

for (i in 2:nsim) {
  cand = c(rlogis(1,muhat[i-1],1),1/rgamma(1,sigmhat[i-1]^2+2,rate=(sigmhat[i-
1]^2+1))) #generate samlpe from candidate or proposal (not prior) ,need to install package
of inverse gamma

  accep = min(posterior(cand[1], cand[2])*dcand(muhat[i-1],sigmhat[i-
1])/(posterior(muhat[i-1], sigmhat[i-1])*dcand(cand[1],cand[2])), 1)

  dad=runif(1)

  rho = (dad < accep)
  muhat[i] = cand[1] * rho + muhat[i - 1] * (1 - rho)
  sigmhat[i] = cand[2] * rho + sigmhat[i - 1] * (1 - rho)

}

n=1000
muhat=muhat[-1:-n]
muhat
mmuhat[ii]=mean(muhat)
sigmhat=sigmhat[-1:-n]
```

```
sigmhat
ssigmhat[ii]=mean(sigmhat)
#credible interval for mu
Sr1=sort(muhat)
Low1[ii]=Sr1[0.025*(nsim-n)]
Upper1[ii]=Sr1[0.975*(nsim-n)]
Exact1[ii]=Upper1[ii]-Low1[ii]
#credible interval for sigm
Sr2=sort(sigmhat)
Low2[ii]=Sr2[0.025*(nsim-n)]
Upper2[ii]=Sr2[0.975*(nsim-n)]
#EL for mu
Exact2[ii]=Upper2[ii]-Low2[ii]
##################################
 t1=0.5
S1=1-1/(1+exp(-pi*(log(t1)-muhat)/(sigmhat*sqrt(3))))
S11[ii]=mean(S1)
t2=1
S2=1-1/(1+exp(-pi*(log(t2)-muhat)/(sigmhat*sqrt(3))))
S22[ii]=mean(S2)
t3=2
S3=1-1/(1+exp(-pi*(log(t3)-muhat)/(sigmhat*sqrt(3))))
S33[ii]=mean(S3)
#################################
#Credible interval for S1
Srs1=sort(S1)
LowS1[ii]=Srs1[0.025*(nsim-n)]
LowS1
UpperS1[ii]=Srs1[0.975*(nsim-n)]
UpperS1
#EL for S1
ExactS1[ii]=UpperS1[ii]-LowS1[ii]
#Credible interval for S2
Srs2=sort(S2)
LowS2[ii]=Srs2[0.025*(nsim-n)]
LowS2
UpperS2[ii]=Srs2[0.975*(nsim-n)]
UpperS2
#EL for S2
ExactS2[ii]=UpperS2[ii]-LowS2[ii]
#Credible interval for S3# 95% CI
Srs3=sort(S3)
LowS3[ii]=Srs3[0.025*(nsim-n)]
LowS3
UpperS3[ii]=Srs3[0.975*(nsim-n)]
```

```
    UpperS3
    #EL for S3
    ExactS3[ii]=UpperS3[ii]-LowS3[ii]


  }
  }
}

#end of loop
##################################
mb0=mean(mmuhat) #Bayesian estimate for mu
mb0
sb0=mean(ssigmhat)# Bayesian estimate for sigma
sb0
###################### Bias for mu & sigma
Bm=mb0-mu
Bm
Bs=sb0-sigm
Bs
#############################################
# an approximate Bayesian estimate of S(t) based on SEL function is the posterior mean
SS11=mean(S11)#estimate for S1
SS11
SS22=mean(S22)#estimate for S2
SS22
SS33=mean(S33)#estimate for S3
SS33
######################################
#need intial value for S1,S2,S3  by sub. mu &  sigm
S01=1-1/(1+exp(-pi*(log(t1)-mu)/(sigm*sqrt(3))))
S02=1-1/(1+exp(-pi*(log(t2)-mu)/(sigm*sqrt(3))))
S03=1-1/(1+exp(-pi*(log(t3)-mu)/(sigm*sqrt(3))))
###########################bais survival function
S1b=SS11-S01#bais for S1
S1b
S2b=SS22-S02#bais for S2
S2b
S3b=SS33-S03#bais for S3
S3b

########################MSE for mu & sigma
MSEm=sum((mmuhat-mu)^2)/nBoot
MSEm
MSEs=sum((ssigmhat-sigm)^2)/nBoot
```

```
MSEs
######################MSE for S1,S2,S3
MSEs1=sum((S11-S01)^2)/nBoot
MSEs1
MSEs2=sum((S22-S02)^2)/nBoot
MSEs2
MSEs2=sum((S33-S03)^2)/nBoot
MSEs2
######################Expected Length. for mu ,sigma,S1,S2 ,S3
AL1=sum(Exact1)/nBoot
AL1
AL2=sum(Exact2)/nBoot
AL2
ALS1=sum(ExactS1)/nBoot
ALS1
ALS2=sum(ExactS2)/nBoot
ALS2
ALS3=sum(ExactS3)/nBoot
ALS3
###################################
#CII for mu
CI1=c(mean(Low1),mean(Upper1))
CI1
#CII for sigma
CI2=c(mean(Low2),mean(Upper2))
CI2
#CII for S1
CIS1=c(mean(LowS1),mean(UpperS1))
CIS1
#CII for S2
CIS2=c(mean(LowS2),mean(UpperS2))
CIS2
#CII for S3
CIS3=c(mean(LowS3),mean(UpperS3))
CIS3
###################coverage prob. for mu ,sigma,S1,S2 ,S3########
# cv.prob.for mu
for(i in 1:nBoot){

  if(Low1[i]<=mu & Upper1[i]>=mu)  # check with equal or not
    count1=count1+1

}
count1
prop1=sum(count1)/nBoot
```

```
prop1
#####################
# cv.prob.for sigm
for(i in 1:nBoot){

  if(Low2[i]<=sigm & Upper2[i]>=sigm)
    count2=count2+1

}
count2
prop2=sum(count2)/nBoot
prop2
########################
# cv.prob.for S1
for(i in 1:nBoot){

  if(LowS1[i]<=S01 & UpperS1[i]>=S01)
    countS1=countS1+1

}
countS1
prop3=sum(countS1)/nBoot
prop3
##########################
# cv.prob.for S2
for(i in 1:nBoot){

  if(LowS2[i]<=S02 & UpperS2[i]>=S02)
    countS2=countS2+1

}
countS2
prop4=sum(countS2)/nBoot
prop4
##########################
# cv.prob.for S3
for(i in 1:nBoot){

  if(LowS3[i]<=S03 & UpperS3[i]>=S03)
    countS3=countS3+1

}
countS3
prop5=sum(countS3)/nBoot
prop5
```

A Monte Carlo Simulation Study for n= 50, m=30 & T=1 based on MCMC method by using an informative prior.

```
#case 1
nBoot=2000
nsim=11000
muhat=c()
sigmhat=c()
mmuhat=rep(0,nBoot)
ssigmhat=rep(0,nBoot)
Low1=rep(0,nBoot)
Upper1=rep(0,nBoot)
Exact1=rep(0,nBoot)
Low2=rep(0,nBoot)
Upper2=rep(0,nBoot)
Exact2=rep(0,nBoot)
S11=rep(0,nBoot)
S22=rep(0,nBoot)
S33=rep(0,nBoot)
LowS1=rep(0,nBoot)
UpperS1=rep(0,nBoot)
LowS2=rep(0,nBoot)
UpperS2=rep(0,nBoot)
LowS3=rep(0,nBoot)
UpperS3=rep(0,nBoot)
ExactS1=rep(0,nBoot)
ExactS2=rep(0,nBoot)
ExactS3=rep(0,nBoot)
count1=0;count2=0;countS1=0;countS2=0;countS3=0
for(ii in 1:nBoot){
  print(ii)
  T=1
  n=50
  m=30
  mu=0
  sigm=1
  pi=3.14
  t1=0.5
  t2=1
  t3=2
  w=c()
  w=runif(m)      #step 1
  w
  r=rep(0,m)              # empty vector
  #progressive censored scheme
```

```
p=rep(0,m)
v=rep(0,m)
x=c()
u=rep(0,m)
y=rep(0,m)
r=c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,20)
length(r)
p=r[m]
p
for(i in 1:m)
{
  v[i]=w[i]^(1/(i+p))
  p=p+r[m-i]
}
x=v[m]                  #step2
x

for(i in 1:m){        #step 3
  u[i]=1-x
  x=x*v[m-i]
}

u      # progressive type II censored sample from the uniform.

for(i in 1:m){
y[i]= exp(mu)*(1/u[i]-1)^(-sigm*sqrt(3)/pi)
}
y     # the required sample from the quantile function of log-logestic distribution
d=c()

for(i in 1:m ){

  if(y[i]<T){
    d[i]=y[i]

  } else {d[i]==0 }
}

d

j=length(d)          #Determine the value of j (j is the number of failure observation before
time T)
j

if(j+1<m){
```

```
rs=c()

for(i in 1:j)
{
  rs[i]=r[i]
}
rs #progressive censored scheme for J

R=sum(rs)
R
#using invers transformation method to get sample from truncated distribution.
g=c()
U=c()
a=y[j+1]
aa=1/(1+a^(-pi/(sigm*sqrt(3)))*exp(mu*pi/(sigm*sqrt(3)))) #CDF for Log logestic dist
.
bb=1
U=runif(n-R-j-1,aa,bb)
U
A=sort(U)

for(i in 1:n-R-j-1){

  g[i]=exp(mu)*(1/A[i]-1)^(-sigm*sqrt(3)/pi)
  # quantile function with specific range instead of U(0,1)
}
g

f=c()#the first order statistics m-j-1
for(i in 1:m-j-1)
{
  f[i]=g[i]
}

ds=c()

for(i in 1:j){
  ds[i]=y[i]

}
ds

dss=c()

for(i in 0:j+1){
```

```
  dss[i]=y[i]

}
dss

D=c()
D=c(dss,f)  # use function combine to add the two vectors together
D          #The adaptive progressive type II censored data according to both algorithms

dm=D[m]        # last value in the adaptive progressive type II censored data)
dm

RR=(n-m-sum(rs))     #or    R=sum(rs)
RR

da=D
da

#MCMC(MH):Bayeian method
muhat[1]=mu
sigmhat[1]=sigm

posterior=function(muu,sigmm){
  exp(m*log(pi)-(m*log(sigmm)+sum(log(da))+m*log(sqrt(3)))-
sum(log(da))*pi/(sigmm*sqrt(3))+(m*muu*pi)/(sigmm*sqrt(3))-2*sum(log(1+exp((-
pi*log(da)+pi*muu)/(sigmm*sqrt(3)))))-pi/(sigmm*sqrt(3))*sum(rs*(log(ds)-muu))-
sum(rs*log(1+exp((-pi*log(ds)+pi*muu)/(sigmm*sqrt(3)))))+RR*(-
pi*log(dm)+muu*pi)/(sigmm*sqrt(3))-RR*log(1+exp((-
pi*log(dm)+pi*muu)/(sigmm*sqrt(3))))-(sigmm)-(muu)-2*log(1+exp(-muu)))
}

dcand = function(muu,sigmm) {
  dlogis(muu,mu,1)*dgamma(1/sigmm,sigm^2+2,rate=sigm*(sigm^2+1))/sigmm^2
}

for (i in 2:nsim) {
  cand = c(rlogis(1,muhat[i-1],1),1/rgamma(1,sigmhat[i-1]^2+2,rate=(sigmhat[i-
1]^2+1))) #generate samlpe from candidate or proposal (not prior)  ,need to install package
of inverse gamma

  accep = min(posterior(cand[1], cand[2])*dcand(muhat[i-1],sigmhat[i-
1])/(posterior(muhat[i-1], sigmhat[i-1])*dcand(cand[1],cand[2])), 1)

  dad=runif(1)
```

```
  rho = (dad < accep)
  muhat[i] = cand[1] * rho + muhat[i - 1] * (1 - rho)
  sigmhat[i] = cand[2] * rho + sigmhat[i - 1] * (1 - rho)


}

n=1000
muhat=muhat[-1:-n]
mmuhat[ii]=mean(muhat)
sigmhat=sigmhat[-1:-n]
sigmhat
ssigmhat[ii]=mean(sigmhat)
#credible interval for mu
Sr1=sort(muhat)
Low1[ii]=Sr1[0.025*(nsim-n)]
Upper1[ii]=Sr1[0.975*(nsim-n)]
Exact1[ii]=Upper1[ii]-Low1[ii]
#credible interval for sigm
Sr2=sort(sigmhat)
Low2[ii]=Sr2[0.025*(nsim-n)]
Upper2[ii]=Sr2[0.975*(nsim-n)]
#EL for mu
Exact2[ii]=Upper2[ii]-Low2[ii]
##################################
t1=0.5
S1=1-1/(1+exp(-pi*(log(t1)-muhat)/(sigmhat*sqrt(3))))
S11[ii]=mean(S1)
t2=1
S2=1-1/(1+exp(-pi*(log(t2)-muhat)/(sigmhat*sqrt(3))))
S22[ii]=mean(S2)
t3=2
S3=1-1/(1+exp(-pi*(log(t3)-muhat)/(sigmhat*sqrt(3))))
S33[ii]=mean(S3)
#################################
#Credible interval for S1
Srs1=sort(S1)
LowS1[ii]=Srs1[0.025*(nsim-n)]
UpperS1[ii]=Srs1[0.975*(nsim-n)]
#EL for S1
ExactS1[ii]=UpperS1[ii]-LowS1[ii]
#Credible interval for S2
Srs2=sort(S2)
LowS2[ii]=Srs2[0.025*(nsim-n)]
LowS2
```

```
UpperS2[ii]=Srs2[0.975*(nsim-n)]
UpperS2
#EL for S2
ExactS2[ii]=UpperS2[ii]-LowS2[ii]
#Credible interval for S3# 95% CI
Srs3=sort(S3)
LowS3[ii]=Srs3[0.025*(nsim-n)]
LowS3
UpperS3[ii]=Srs3[0.975*(nsim-n)]
UpperS3
#EL for S3
ExactS3[ii]=UpperS3[ii]-LowS3[ii]
  }else{if((j==m)||(j+1==m)){
da=y
dm=y[m]       # last value in the adaptive progressive type II censored data)
dm
rs=c()

for(i in 1:j)
{
 rs[i]=r[i]
}
rs #progressive censored scheme for J

R=sum(rs)
R
RR=(n-m-sum(rs))     #or    R=sum(rs)
RR
ds=c()

for(i in 1:j){
 ds[i]=y[i]

}
ds

#MCMC(MH): Bayesian method

muhat[1]=mu
sigmhat[1]=sigm
 posterior=function(muu,sigmm){
 exp(m*log(pi)-(m*log(sigmm)+sum(log(da))+m*log(sqrt(3)))-
sum(log(da))*pi/(sigmm*sqrt(3))+(m*muu*pi)/(sigmm*sqrt(3))-2*sum(log(1+exp((-
pi*log(da)+pi*muu)/(sigmm*sqrt(3)))))-pi/(sigmm*sqrt(3))*sum(rs*(log(ds)-muu))-
sum(rs*log(1+exp((-pi*log(ds)+pi*muu)/(sigmm*sqrt(3)))))+RR*(-
```

```
pi*log(dm)+muu*pi)/(sigmm*sqrt(3))-RR*log(1+exp((-
pi*log(dm)+pi*muu)/(sigmm*sqrt(3))))-(sigmm)-(muu)-2*log(1+exp(-muu)))
  }
  dcand = function(muu,sigmm) {
    dlogis(muu,mu,1)*dgamma(1/sigmm,sigm^2+2,rate=sigm*(sigm^2+1))/sigmm^2

  }
  for (i in 2:nsim) {
    cand = c(rlogis(1,muhat[i-1],1),1/rgamma(1,sigmhat[i-1]^2+2,rate=(sigmhat[i-
1]^2+1)))

    accep = min(posterior(cand[1], cand[2])*dcand(muhat[i-1],sigmhat[i-
1])/(posterior(muhat[i-1], sigmhat[i-1])*dcand(cand[1],cand[2])), 1)

    dad=runif(1)

    rho = (dad < accep)
    muhat[i] = cand[1] * rho + muhat[i - 1] * (1 - rho)
    sigmhat[i] = cand[2] * rho + sigmhat[i - 1] * (1 - rho)

  }

  n=1000
  muhat=muhat[-1:-n]
  muhat

  mmuhat[ii]=mean(muhat)
  sigmhat=sigmhat[-1:-n]
  sigmhat
  ssigmhat[ii]=mean(sigmhat)
  #credible interval for mu
  Sr1=sort(muhat)
  Low1[ii]=Sr1[0.025*(nsim-n)]
  Upper1[ii]=Sr1[0.975*(nsim-n)]
  Exact1[ii]=Upper1[ii]-Low1[ii]
  #credible interval for sigm
  Sr2=sort(sigmhat)
  Low2[ii]=Sr2[0.025*(nsim-n)]
  Upper2[ii]=Sr2[0.975*(nsim-n)]
  #EL for mu
  Exact2[ii]=Upper2[ii]-Low2[ii]
  #################################

  t1=0.5
  S1=1-1/(1+exp(-pi*(log(t1)-muhat)/(sigmhat*sqrt(3))))
```

```
    S11[ii]=mean(S1)
    t2=1
    S2=1-1/(1+exp(-pi*(log(t2)-muhat)/(sigmhat*sqrt(3))))
    S22[ii]=mean(S2)
    t3=2
    S3=1-1/(1+exp(-pi*(log(t3)-muhat)/(sigmhat*sqrt(3))))
    S33[ii]=mean(S3)
    ################################
    #Credible interval for S1
    Srs1=sort(S1)
    LowS1[ii]=Srs1[0.025*(nsim-n)]
    LowS1
    UpperS1[ii]=Srs1[0.975*(nsim-n)]
    UpperS1
    #EL for S1
    ExactS1[ii]=UpperS1[ii]-LowS1[ii]
    #Credible interval for S2
    Srs2=sort(S2)
    LowS2[ii]=Srs2[0.025*(nsim-n)]
    LowS2
    UpperS2[ii]=Srs2[0.975*(nsim-n)]
    UpperS2
    #EL for S2
    ExactS2[ii]=UpperS2[ii]-LowS2[ii]
    #Credible interval for S3# 95% CI
    Srs3=sort(S3)
    LowS3[ii]=Srs3[0.025*(nsim-n)]
    LowS3
    UpperS3[ii]=Srs3[0.975*(nsim-n)]
    UpperS3
    #EL for S3
    ExactS3[ii]=UpperS3[ii]-LowS3[ii]

    }
  }
}
#end of loop
#################################
mb0=mean(mmuhat) #Bayesian estimate for mu
mb0
sb0=mean(ssigmhat)# Bayesian estimate for sigm
sb0
#################### Bias for mu & sigma
Bm=mb0-mu
Bm
```

```
Bs=sb0-sigm
Bs
############################################
# an approximate Bayesian estimate of S(t) based on SEL function is the posterior mean
SS11=mean(S11)#estimate for S1
SS11
SS22=mean(S22)#estimate for S2
SS22
SS33=mean(S33)#estimate for S3
SS33
######################################
#need intial value for S1,S2,S3  by sub. mu &  sigm
S01=1-1/(1+exp(-pi*(log(t1)-mu)/(sigm*sqrt(3))))
S02=1-1/(1+exp(-pi*(log(t2)-mu)/(sigm*sqrt(3))))
S03=1-1/(1+exp(-pi*(log(t3)-mu)/(sigm*sqrt(3))))

###########################bais survival function
S1b=SS11-S01#bais for S1
S1b
S2b=SS22-S02#bais for S2
S2b
S3b=SS33-S03#bais for S3
S3b

#######################MSE for mu & sigma
MSEm=sum((mmuhat-mu)^2)/nBoot
MSEm
MSEs=sum((ssigmhat-sigm)^2)/nBoot
MSEs
#######################MSE for S1,S2,S3
MSEs1=sum((S11-S01)^2)/nBoot
MSEs1
MSEs2=sum((S22-S02)^2)/nBoot
MSEs2
MSEs2=sum((S33-S03)^2)/nBoot
MSEs2
######################Expected Length. for mu ,sigma,S1,S2 ,S3
AL1=sum(Exact1)/nBoot
AL1
AL2=sum(Exact2)/nBoot
AL2
ALS1=sum(ExactS1)/nBoot
ALS1
ALS2=sum(ExactS2)/nBoot
ALS2
```

```
ALS3=sum(ExactS3)/nBoot
ALS3
##################################
#CII for mu
CI1=c(mean(Low1),mean(Upper1))
CI1
#CII for ssigm
CI2=c(mean(Low2),mean(Upper2))
CI2
#CII for S1
CIS1=c(mean(LowS1),mean(UpperS1))
CIS1
#CII for S2
CIS2=c(mean(LowS2),mean(UpperS2))
CIS2
#CII for S3
CIS3=c(mean(LowS3),mean(UpperS3))
CIS3
###################covarage prob. for mu ,sigma,S1,S2 ,S3
# cv.prob.for mu
for(i in 1:nBoot){

  if(Low1[i]<=mu & Upper1[i]>=mu)  # check with equal or not
    count1=count1+1

}
count1
prop1=sum(count1)/nBoot
prop1
#####################
# cv.prob.for sigm
for(i in 1:nBoot){

  if(Low2[i]<=sigm & Upper2[i]>=sigm)
    count2=count2+1

}
count2
prop2=sum(count2)/nBoot
prop2
########################
# cv.prob.for S1
for(i in 1:nBoot){

  if(LowS1[i]<=S01 & UpperS1[i]>=S01)
```

```
    countS1=countS1+1

}
countS1
prop3=sum(countS1)/nBoot
prop3
##########################
# cv. prob. for S2
for(i in 1:nBoot){

  if(LowS2[i]<=S02 & UpperS2[i]>=S02)
    countS2=countS2+1

}
countS2
prop4=sum(countS2)/nBoot
prop4
##########################
# cv.prob.for S3
for(i in 1:nBoot){

  if(LowS3[i]<=S03 & UpperS3[i]>=S03)
    countS3=countS3+1

}
countS3
prop5=sum(countS3)/nBoot
prop5
####################################################
```

APPENDIX C: R CODE FOR REAL LIFE EXAMPLE

```
#The MLE of complete data set n=m(no censoring)

y=c(0.39,0.81,0.85,0.98,1.08,1.12,1.17,1.18,1.22,1.25,1.36,1.41,1.47,1.57,1.57,1.59,1.59,
1.61,1.61,1.69,1.69,1.71,1.73,1.80,1.84,1.84,1.87,1.89,1.92,2.00,2.03,2.03,2.05,2.12,2.17,
2.17,2.17,2.35,2.38,2.41,2.43,2.48,2.48,2.50,2.53,2.55,2.55,2.56,2.59,2.67,2.73,2.74,2.76,
2.77,2.79,2.81,2.81,2.82,2.83,2.85,2.87,2.88,2.93,2.95,2.96,2.97,2.97,3.09,3.11,3.11,3.15,
3.15,3.19,3.19,3.22,3.22,3.27,3.28,3.31,3.31,3.33,3.39,3.39,3.51,3.56,3.60,3.65,3.68,3.68,
3.68,3.70,3.75,4.20,4.38,4.42,4.70,4.90,4.91,5.08,5.56)
hist(y,freq = FALSE)
length(y)
mu=0;sigm=1
S0=1-1/(1+exp(-pi*(log(t)-mu)/(sigm*sqrt(3))))
S0
m=length(y)
like=function(b)
{-(m*log(pi)-(m*log(b[2])+sum(log(y))+m*log(sqrt(3)))-
sum(log(y))*pi/(b[2]*sqrt(3))+(m*b[1]*pi)/(b[2]*sqrt(3))-2*sum(log(1+exp((-
pi*log(y)+pi*b[1])/(b[2]*sqrt(3))))))}

nn=nlm(like,c(mu,sigm),hessian=TRUE)
nn
mu1=nn$estimate[1]
mu1
sigm1=nn$estimate[2]
sigm1
t=0.5
S=1-1/(1+exp(-pi*(log(t)-nn$estimate[1])/(nn$estimate[2]*sqrt(3))))
S
xx=(pi/sqrt(3))*(log(y)-mu1)/sigm1
xx
## (KS.TEST FOR ONE SAMPLE)
ks.test(jitter(xx),"plogis",0,1)#standard logistic distribution
######################################################
###Real data set (MLE): censoring scheme
#case1:T=3.66
x1=sample(1:99, 30, replace=F)
x1
mu=0
sigm=1
n=100
pi=3.14
m=60
```

```
R=c(30,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10)

j=60 #All number of observed failures before time T
da=ds=c(0.39,0.85,1.08,1.17,1.22,1.25,1.36,1.57,1.59,1.59,1.61,1.71,1.73,1.80,1.84,1.84,
1.87,1.89,1.92,2.03,2.03,2.05,2.12,2.17,2.17,2.38,2.41,2.43,2.48,2.53,2.55,2.55,2.67,2.73,
2.74,2.76,2.77,2.79,2.81,2.81,2.82,2.83,2.87,2.95,2.96,2.97,3.09,3.11,3.11,3.15,3.19,3.19,
3.22,3.27,3.28,3.31,3.39,3.51,3.60,3.65)
length(da)
RR=0
rs=R
rs
dm=3.65
like=function(b)
{-(m*log(pi)-(m*log(b[2])+sum(log(da))+m*log(sqrt(3)))-
sum(log(da))*pi/(b[2]*sqrt(3))+(m*b[1]*pi)/(b[2]*sqrt(3))-2*sum(log(1+exp((-
pi*log(da)+pi*b[1])/(b[2]*sqrt(3)))))-pi/(b[2]*sqrt(3))*sum(rs*(log(ds)-b[1]))-
sum(rs*log(1+exp((-pi*log(ds)+pi*b[1])/(b[2]*sqrt(3)))))+RR*(-
pi*log(dm)+b[1]*pi)/(b[2]*sqrt(3))-RR*log(1+exp((-
pi*log(dm)+pi*b[1])/(b[2]*sqrt(3)))))}

nn=nlm(like,c(mu,sigm),hessian=TRUE)

mu1=nn$estimate[1]
mu1
sigm1=nn$estimate[2]
sigm1
#####
#Estimate of survival functions
t=0.5
S=1-1/(1+exp(-pi*(log(t)-mu1)/(sigm1*sqrt(3))))
S
#95% (alpha=0.05), 100(1-alpha)% two sided approx.CI for the parameters mu & sigma
or qnorm(alpha/2)=qnorm(0.025)

inv=solve(nn$hessian)
var1=inv[1]# mu
var1
var2=inv[4]#sigma
var2
Low1=mu1-qnorm(0.975)*sqrt(var1)
Upp1=mu1+qnorm(0.975)*sqrt(var1)
CImu1=c(Low1,Upp1)
CImu1
Low2=sigm1-qnorm(0.975)*sqrt(var2)
```

```
Upp2=sigm1+qnorm(0.975)*sqrt(var2)
CIsigm1=c(Low2,Upp2)
CIsigm1
###95% CI for survival function S1(0.5)
G1=c(((pi/(sigm1*sqrt(3)))*exp(-pi*(log(t)-mu1)/(sigm1*sqrt(3))))/(1+exp(-pi*(log(t)-
mu1)/(sigm1*sqrt(3))))^2,((pi*(log(t)-mu1)/((sigm1)^2*sqrt(3)))*exp(-pi*(log(t)-
mu1)/(sigm1*sqrt(3))))/(1+exp(-pi*(log(t)-mu1)/(sigm1*sqrt(3))))^2)
G11=rbind(G1)
varS1=G11%*%inv
varS1e=varS1%*%t(G11)
LowS1=S-qnorm(0.975)*sqrt(varS1e)
UppS1=S+qnorm(0.975)*sqrt(varS1e)
CIS1=c(LowS1,UppS1)
CIS1
#### The length of CI for location & scale parameters
ExactL1=Upp1-Low1
ExactL1
ExactL2=Upp2-Low2
ExactL2
#### The length of CI for survival function at t=0.5
ExactLS1=UppS1-LowS1
ExactLS1
#,3.68)3.68,3.68,3.70,3.75,4.38,4.42,4.70,4.90,4.91)
#########################
#case2:T=1.60
x1=sample(1:99, 30, replace=F)
x1
mu=0
sigm=1
n=100
pi=3.14
m=60
R=c(30,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10)
j=13 #number of observed failures before time T
ds=c(0.39,0.81,0.85,0.98,1.12,1.17,1.22,1.25,1.36,1.41,1.57,1.59,1.59)
length(ds)
da=c(0.39,0.81,0.85,0.98,1.12,1.17,1.22,1.25,1.36,1.41,1.57,1.59,1.59,1.61,1.69,1.69,1.7
1,1.73,1.80,1.84,1.84,1.89,1.92,2.00,2.03,2.05,2.12,2.17,2.17,2.35,2.41,2.43,2.48,2.48,2.5
0,2.55,2.59,2.76,2.79,2.81,2.81,2.81,2.83,2.85,2.87,2.93,2.95,2.96,2.97,3.09,3.11,3.15,3.1
9,3.19,3.22,3.31,3.31,3.33,3.39,3.51)
length(da) #,3.56,3.65,3.68,3.68,3.70,3.75,4.20,4.70,4.91,5.56)
rs=c(30,0,0,0,0,0,0,0,0,0,0,0,0)
RR=10
dm=3.51
```

```
like=function(b)
{-(m*log(pi)-(m*log(b[2])+sum(log(da))+m*log(sqrt(3)))-
sum(log(da))*pi/(b[2]*sqrt(3))+(m*b[1]*pi)/(b[2]*sqrt(3))-2*sum(log(1+exp((-
pi*log(da)+pi*b[1])/(b[2]*sqrt(3)))))-pi/(b[2]*sqrt(3))*sum(rs*(log(ds)-b[1]))-
sum(rs*log(1+exp((-pi*log(ds)+pi*b[1])/(b[2]*sqrt(3)))))+RR*(-
pi*log(dm)+b[1]*pi)/(b[2]*sqrt(3))-RR*log(1+exp((-
pi*log(dm)+pi*b[1])/(b[2]*sqrt(3))))))}

nn=nlm(like,c(mu,sigm),hessian=TRUE)

mu1=nn$estimate[1]
mu1
sigm1=nn$estimate[2]
sigm1
#####
#Estimate of survival functions
t=0.5
S=1-1/(1+exp(-pi*(log(t)-mu1)/(sigm1*sqrt(3))))
S

#95% (alpha=0.05)CI (100(1-alpha)% two sided approx.CI for the parameters mu &
sigma or qnorm(alpha/2)=qnorm(0.025)

inv=solve(nn$hessian)
var1=inv[1]# mu
var1
var2=inv[4]#sigma
var2
Low1=mu1-qnorm(0.975)*sqrt(var1)
Upp1=mu1+qnorm(0.975)*sqrt(var1)
CImu1=c(Low1,Upp1)
CImu1
Low2=sigm1-qnorm(0.975)*sqrt(var2)
Upp2=sigm1+qnorm(0.975)*sqrt(var2)
CIsigm1=c(Low2,Upp2)
CIsigm1
###95% CI for survival function S(0.6)
G1=c(((pi/(sigm1*sqrt(3)))*exp(-pi*(log(t)-mu1)/(sigm1*sqrt(3))))/(1+exp(-pi*(log(t)-
mu1)/(sigm1*sqrt(3))))^2,((pi*(log(t)-mu1)/((sigm1)^2*sqrt(3)))*exp(-pi*(log(t)-
mu1)/(sigm1*sqrt(3))))/(1+exp(-pi*(log(t)-mu1)/(sigm1*sqrt(3))))^2)
G11=rbind(G1)
varS1=G11%*%inv
varS1e=varS1%*%t(G11)
LowS1=S-qnorm(0.975)*sqrt(varS1e)
UppS1=S+qnorm(0.975)*sqrt(varS1e)
```

```
CIS1=c(LowS1,UppS1)
CIS1
############Length of CI. for location & scale parameters
ExactL1=Upp1-Low1
ExactL1
ExactL2=Upp2-Low2
ExactL2
##################Length of CI. for survival function at t=0.5
ExactLS1=UppS1-LowS1
ExactLS1
##########################################################################
#case1:T=3.66 (MCMC Method: censoring scheme)
nsim=11000
mu=0
sigm=1
n=100
pi=3.14
m=60
R=c(30,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10)
j=60 #All number of observed failures before time T
da=ds=c(0.39,0.85,1.08,1.17,1.22,1.25,1.36,1.57,1.59,1.59,1.61,1.71,1.73,1.80,1.84,1.84,
1.87,1.89,1.92,2.03,2.03,2.05,2.12,2.17,2.17,2.38,2.41,2.43,2.48,2.53,2.55,2.55,2.67,2.73,
2.74,2.76,2.77,2.79,2.81,2.81,2.82,2.83,2.87,2.95,2.96,2.97,3.09,3.11,3.11,3.15,3.19,3.19,
3.22,3.27,3.28,3.31,3.39,3.51,3.60,3.65)
length(da)
RR=0
rs=R
rs
dm=3.65
muhat=c()
sigmhat=c()
muhat[1]=mu
sigmhat[1]=sigm
posterior=function(muu,sigmm){
  exp(m*log(pi)-(m*log(sigmm)+sum(log(da))+m*log(sqrt(3)))-
sum(log(da))*pi/(sigmm*sqrt(3))+(m*muu*pi)/(sigmm*sqrt(3))-2*sum(log(1+exp((-
pi*log(da)+pi*muu)/(sigmm*sqrt(3)))))-pi/(sigmm*sqrt(3))*sum(rs*(log(ds)-muu))-
sum(rs*log(1+exp((-pi*log(ds)+pi*muu)/(sigmm*sqrt(3)))))+RR*(-
pi*log(dm)+muu*pi)/(sigmm*sqrt(3))-RR*log(1+exp((-
pi*log(dm)+pi*muu)/(sigmm*sqrt(3)))))-log(sigmm))
}

dcand = function(muu,sigmm) {
  dlogis(muu,mu,1)*dgamma(1/sigmm,sigm^2+2,rate=sigm*(sigm^2+1))/sigmm^2
```

```
}
for (i in 2:nsim) {
  cand = c(rlogis(1,muhat[i-1],1),1/rgamma(1,sigmhat[i-1]^2+2,rate=(sigmhat[i-1]^2+1)))
  accep = min(posterior(cand[1], cand[2])/posterior(muhat[i-1], sigmhat[i-
1])*dcand(muhat[i-1],sigmhat[i-1])/dcand(cand[1],cand[2]), 1)
  rho = (runif(1) < accep)
  muhat[i] = cand[1] * rho + muhat[i - 1] * (1 - rho)
  sigmhat[i] = cand[2] * rho + sigmhat[i - 1] * (1 - rho)
 }
muhat
sigmhat
cm=as.mcmc(muhat)
summary(cm)
plot(cm,main="µ")
cs=as.mcmc(sigmhat)
plot(cs,main="sigma")
###################
autocorr.plot(cm,main="µ")
autocorr.plot(cs,main="sigma")
autocorr(cm)# the autocorrelation decrease at lag 50
autocorr(cs)# the autocorrelation decrease at lag 50
####################
n=1000
muhat=muhat[-1:-n]
muhat
sigmhat=sigmhat[-1:-n]
sigmhat
############################## draw marginal density function again
i=seq(1,length(muhat),by=50)
out.sub=density(muhat[i])
out=density(muhat,bw=out.sub$bw)
plot(out,main="Âµ")#marginal density after burn in
hist(muhat,main="Âµ",freq = FALSE)#not draw
hist(sigmhat,main="sigma",freq = FALSE)#not draw
hist(muhat,freq =FALSE)

##########
ii=seq(1,length(sigmhat),by=50)
out.sub=density(sigmhat[ii])
out=density(sigmhat,bw=out.sub$bw)
plot(out,main="sigma")#marginal density after burn in
hist(sigmhat,freq =FALSE)
############################Statistical inference
mmuhat=mean(muhat)
mmuhat
```

```
ssigmhat=mean(sigmhat)
ssigmhat
#Estimate of survival functions
t=0.5
S1=1-1/(1+exp(-pi*(log(t)-muhat)/(sigmhat*sqrt(3))))
S11=mean(S1)
S11
#credible interval for mu
Sr1=sort(muhat)
Low1=Sr1[0.025*(nsim-n)]
Upper1=Sr1[0.975*(nsim-n)]
CImu=c(Low1,Upper1)
CImu
#credible interval for sigm
Sr2=sort(sigmhat)
Low2=Sr2[0.025*(nsim-n)]
Upper2=Sr2[0.975*(nsim-n)]
CIsigm=c(Low2,Upper2)
CIsigm
################################### 95% Credible interval
#Credible interval for S1
Srs1=sort(S1)
LowS1=Srs1[0.025*(nsim-n)]
UpperS1=Srs1[0.975*(nsim-n)]
CIS1=c(LowS1,UpperS1)
CIS1

####################
#EL for mu
Exact1=Upper1-Low1
Exact1
#EL for sigm
Exact2=Upper2-Low2
Exact2
#################
#EL for S1
ExactS1=UpperS1-LowS1
ExactS1
#case2:T=1.60
nsim=11000
mu=0
sigm=1
n=100
pi=3.14
m=60
```

```
R=c(30,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10)
j=13 #number of observed failures before time T
ds=c(0.39,0.81,0.85,0.98,1.12,1.17,1.22,1.25,1.36,1.41,1.57,1.59,1.59)
length(ds)
da=c(0.39,0.81,0.85,0.98,1.12,1.17,1.22,1.25,1.36,1.41,1.57,1.59,1.59,1.61,1.69,1.69,1.7
1,1.73,1.80,1.84,1.84,1.89,1.92,2.00,2.03,2.05,2.12,2.17,2.17,2.35,2.41,2.43,2.48,2.48,2.5
0,2.55,2.59,2.76,2.79,2.81,2.81,2.81,2.83,2.85,2.87,2.93,2.95,2.96,2.97,3.09,3.11,3.15,3.1
9,3.19,3.22,3.31,3.31,3.33,3.39,3.51)
length(da)
rs=c(30,0,0,0,0,0,0,0,0,0,0,0,0)
RR=10
dm=3.51
muhat=c()
sigmhat=c()
muhat[1]=mu
sigmhat[1]=sigm
posterior=function(muu,sigmm){
  exp(m*log(pi)-(m*log(sigmm)+sum(log(da))+m*log(sqrt(3)))-
sum(log(da))*pi/(sigmm*sqrt(3))+(m*muu*pi)/(sigmm*sqrt(3))-2*sum(log(1+exp((-
pi*log(da)+pi*muu)/(sigmm*sqrt(3)))))-pi/(sigmm*sqrt(3))*sum(rs*(log(ds)-muu))-
sum(rs*log(1+exp((-pi*log(ds)+pi*muu)/(sigmm*sqrt(3)))))+RR*(-
pi*log(dm)+muu*pi)/(sigmm*sqrt(3))-RR*log(1+exp((-
pi*log(dm)+pi*muu)/(sigmm*sqrt(3))))-log(sigmm))
}
dcand = function(muu,sigmm) {
  dlogis(muu,mu,1)*dgamma(1/sigmm,sigm^2+2,rate=sigm*(sigm^2+1))/sigmm^2
}
for (i in 2:nsim) {
  cand = c(rlogis(1,muhat[i-1],1),1/rgamma(1,sigmhat[i-1]^2+2,rate=(sigmhat[i-1]^2+1)))
#generate samlpe from candidate or proposal (not prior)  ,need to install package of
inverse gamma
  accep = min(posterior(cand[1], cand[2])/posterior(muhat[i-1], sigmhat[i-
1])*dcand(muhat[i-1],sigmhat[i-1])/dcand(cand[1],cand[2]), 1)
  rho = (runif(1) < accep)
  muhat[i] = cand[1] * rho + muhat[i - 1] * (1 - rho)
  sigmhat[i] = cand[2] * rho + sigmhat[i - 1] * (1 - rho)
}
muhat
sigmhat
n=1000
muhat=muhat[-1:-n]
muhat
mmuhat=mean(muhat)
mmuhat
```

```
sigmhat=sigmhat[-1:-n]
sigmhat
mean(muhat)
mean(sigmhat)
##################################Estimate of survival functions
t=0.5
S1=1-1/(1+exp(-pi*(log(t)-muhat)/(sigmhat*sqrt(3))))
S11=mean(S1)
S11
###################################
#credible interval for mu
Sr1=sort(muhat)
Low1=Sr1[0.025*(nsim-n)]
Upper1=Sr1[0.975*(nsim-n)]
CImu=c(Low1,Upper1)
CImu
#credible interval for sigm
Sr2=sort(sigmhat)
Low2=Sr2[0.025*(nsim-n)]
Upper2=Sr2[0.975*(nsim-n)]
CIsigm=c(Low2,Upper2)
CIsigm
################################## 95% Credible interval
#Credible interval for S1
Srs1=sort(S1)
LowS1=Srs1[0.025*(nsim-n)]
UpperS1=Srs1[0.975*(nsim-n)]
CIS1=c(LowS1,UpperS1)
CIS1
####################
#EL for mu
Exact1=Upper1-Low1
Exact1
#EL for sigm
Exact2=Upper2-Low2
Exact2
#################
#EL for S1
ExactS1=UpperS1-LowS1
ExactS1
```