QATAR UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

LIKLIHOOD INFERENCE FOR STEP STRESS PARTIALLY ACCELERATED LIFE TEST MODEL WITH TYPE I PROGRESSIVELY HYBRID CENSORED DATA FROM GENERALIZED EXPOENTIAL DISTRIBUTION

BY

EMAN ABDULMALIK MOHAMMEDSEMAN

A Thesis Submitted to

the College of Arts and Sciences

in Partial Fulfillment of the Requirements for the Degree of

Masters of Science in Applied Statistics

January 2021

COMMITTEE PAGE

The members of the Committee approve the Thesis of
Eman Mohammedseman defended on 17/11/2020.

Prof. Ayman Bakleezi
Thesis/Dissertation Supervisor

Prof. Mohammed Salehi
Committee Member

Dr. Mohammed Chouch
Committee Member

Approved:

Ibrahim AlKaabi, Dean, College of Arts and Sciences

ABSTRACT

MOHAMMEDSEMAN,EMAN,A., Masters: January : 2021, Applied Statistics

Title: Likelihood Inference for Step Stress Partially Accelerated Life Test Model with Type I Progressively Hybrid Censored Data from Generalized Exponential

Supervisor of Thesis: Prof. Ayman Bakleezi.

This thesis considers the statistical inference on the generalized exponential distribution parameters in presence of progressive Type-I censoring under partially accelerated life test. The maximum likelihood method is used to estimate the unknown parameters in the case of step-stress partially accelerated life tests. The performance of the estimators is investigated using simulation for certain simulation designs and sample sizes. The biases and mean square errors of the maximum-likelihood estimators are computed to assess the performance of the point estimators whereas coverage probability and expected length is used to assess the performance of the interval estimators. These interval estimators are derived using three classical approaches namely; asymptotic, Bootstrap percentile interval, and Bootstrap-t confidence intervals. Comparison between these three methods is also conducted. Further, reliability functions are derived for various time-t and the point and interval estimators are investigated. To illustrate the above, a data analysis is conducted. Finally, conclusions and ideas for possible future research are discussed in this thesis.

DEDICATION

This thesis is dedicated to my parents.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisor Professor Ayman Bakleezi for his constant support, patience, constructive feedbacks, insightful and informative discussions. Without his guidance and continuous help this dissertation would not have been possible. Besides my advisor, I would like to thank the rest of my committee members, Professor Mohammad Salehi and Dr. Mohamed Chaouch.

Additionally, I would like to pass my appreciation to all my teachers who contributed to my education throughout my years of study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

KEYWORDS

G.E	Generalized Exponential
MLE	Maximum likelihood estimator
ALT	Accelerated life test
PALT	Partially accelerated life test
SSALT	Stress accelerated life test
SSPALT	Step stress partially accelerated life test
EL	Expected length
CP	Coverage probability
C.I	Confidence interval
RF	Reliability function
PCI	Bootstrap percentile confidence interval
Boot-t	Bootstrap-t

CHAPTER 1: INTRODUCTION

Due to the rapid advancement in technology and increasing global competition, manufacturers need to produce high-quality products and need to test the reliability of these products. Hence it is important to understand and analyze the life and durability of these products using reliability study. The statistical analysis of what are variously referred to as lifetime, survival time, or failure time data is an important topic in many areas, including the biomedical, engineering, and social sciences. (Lawless, 1982). This thesis considers the likelihood inference for data from generalized exponential distribution under step stress accelerated life test with progressive hybrid type I censoring scheme.

During a reliability study, the life of a product might be very long and hence failure time is very costly. In efforts to reduce these costs; different accelerated life testing is introduced to discover faults and potential modes of failure in a short amount of time. The items are assumed to come from a generalized exponential distribution, and we want to derive and study the performance of likelihood-based inference procedures for the unknown parameters and the reliability of the model when the items are under SSPALT with type I progressive hybrid censoring scheme.

1.1 Life Testing Experiments and Step Stress Accelerated Life Test

Life test is the process of testing the life of a product/subject to identify when the product/subject fails. During a reliability study, the life of a product might be very long, and this makes the life test procedure very costly. In efforts to reduce these costs accelerated life testing (ALT) is introduced. In case of life test for industrial purpose, the process of testing a product is subjected to conditions such as stress, strain, temperatures, voltage, vibration rate, pressure etc. that are excess of its normal service parameters. The aim for ALT is to discover faults and potential modes of failure in a

short amount of time. If the experimenter includes all test units under such stresses, the test is called accelerated life test (ALT), but if he/she includes some of them then the test is called partially accelerated life test (PALT). In ALT test units are run only at accelerated conditions, while in PALT they are run at both normal and accelerated use conditions. The information obtained from the test performed in accelerated conditions is used to predict the actual product performance in the normal conditions.

There are two types of accelerated life test; these are usage rate accelerated test and over stress accelerated test. In usage rate accelerated test, the product usage is prolonged i.e it is kept in operation continuously under its normal working condition until it fails, whereas in over stress accelerated life test, an additional stress is added to its normal condition and the product is observed until it fails.

Over stress accelerated test is also known as step- stress accelerated life test. In a step-stress accelerated life test (SSALT), all test unites are first subjected to a pre-determined low stress level and then at a pre-specified time, the stress level is increased and then the test is continued until a next pre-determined time. In this test, if the stress is changed only once it is known as a Simple SSALT and if the stress is changed multiple times, then it is called a multiple SSALT. One of the major issues in dealing with data from SSALT is the explanation of the effect of the stress change on the remaining lifetime of the test units. There are four basic models that are proposal by different researchers to handle this problem. The first model is the tampered random variable model (TRVM) which was introduced by DeGroot and Goel (1979) for partially accelerated life test, this test is similar to Simple SSALT, except that the low stress level is actually the normal working condition of the item/machine. The second model is the cumulative exposure model (CEM) which was

proposed by Nelson (1980), in this model the distribution of the life of testing units at each stress level is known and the remaining lifetime of the test units is assumed to depend only on the cumulative exposure it has observed. The third model is known as the tampered failure rate model (TFRM) which was developed by Bhattacharrya and Soejoeti (1989). In this model the failure rate at the first stress is known and the effect of changing stress is assumed to multiply a factor to the current failure rate. The fourth model is the linear cumulative exposure model (LCEM) which was suggested by Tang el al. (1996) and Tang (2003). In this model the exposure at each stress can be accumulated linearly and the exposure of a unit at each stress is the ratio of the actual operating time at this stress to its lifetime at this stress and the unit fails when the cumulative exposure of the test units reaches 1. (Xu and Fei, 2012)

1.2 Censoring in Life Test Experiments

Censored data occurs when the time of the event for the data have not been completely observed and this is only partially known. Censoring occurs for many data in several fields, such as engineering, biomedical, social sciences etc. and each of these have their own unique reasons for occurring. Assuming that death is the event of interest and the patients are the subjects, if a patient leaves the study before its completion, the time of death of this patient is unknown and hence it is said to be censored. Such instances make censoring very common in survival analysis, reliability analysis etc.

There are various types of censoring and some of these are listed below with brief description of each.

- Left censoring: occurs when only upper bound on lifetime is available.
- Right censoring: occurs when only lower bound on lifetime is available.

- Interval censoring: when data is collected based on an interval between two values only.
- Type I Censoring: data is collected until experiment reaches a per-determined time.
- Type II Censoring: data is collected until specific number of failures is observed.

The most common censoring for lifetime data is right censoring and this thesis focuses on progressively type-I hybrid censoring which means that the experiment ends when it reaches a pre-determined time, or a pre-determined number of items fail. The term progressive implies that certain number of surviving items are removed from the experiment before the pre-determined time or pre-determined number of items is reached. This allows the experimenter to save more time and cost associated with testing. Additionally, these removals sometimes may be suitable when some of the unfailed units are required for some other tests or when a compromise between reduced time of the experiment and the observation of some extreme lifetimes is required. (Ismail, 2012).

1.3 The Generalized Exponential Distribution

The lifetime in general is a positive random variable which is denoted by X in this thesis and is assumed to be continuous with probability density function (pdf) $f(x)$. Some examples of common lifetime distributions are the exponential distribution, Weibull distribution, gamma distribution, log normal distribution, log logistic distribution, and generalized exponential distribution.

In 1997, Gupta and Kundu introduced the generalized exponential distribution (G.E) which is relatively a new distribution in lifetime distributions. (Gupta and Kundu, 1997). This distribution has 3 parameters, shape, scale and location. When the

value of the shape parameter for G.E, Weibull and gamma distribution is equal to 1, the 3 distributions coincide with a 2-parameter exponential distribution. Thus, all three distributions are somehow extensions or generalizations of the exponential distribution in distinctive ways.

Moreover, the main property of G.E is that it shares many similar properties with gamma distribution and also has distribution function like that of Weibull distribution which enables easy computation. Further, G.E also has the likelihood ratio ordering property on the shape parameter (a property that it shares with gamma distribution); hence when the scale and location parameters are known, it is possible to construct a uniformly most powerful test for testing a one-sided hypothesis on the shape. Hence, G.E distribution can be used as an alternative to the Weibull and gamma distribution for analyzing lifetime data. (Gupta and Kundu, 1999). This study will only consider two parameter G.E where the shape and scale parameters are denoted by θ and λ respectively. Therefore, the density and cumulative functions are given as below:

$$f(x;\theta,\lambda) = \frac{\theta}{\lambda} e^{-\frac{x}{\lambda}} \left(1 - e^{-\frac{x}{\lambda}}\right)^{\theta-1}, \quad x, \theta \,\&\, \lambda > 0 \qquad [1]$$

$$\mathrm{F}(x;\theta,\lambda) = \left(1 - e^{-x/\lambda}\right)^{\theta}, \quad x, \theta \,\&\, \lambda > 0 \qquad [2]$$

For illustration, the probability density function is plotted for various parameter values and this is show in Figure 1 and 2 as follows; $(\theta_1, \lambda_1) = (1,1.4)$, $(\theta_2, \lambda_2) = (2.2,1.4)$, $(\theta_3, \lambda_3) = (6,1.4)$, $(\theta_4, \lambda_4) = (12,1.4)$, $(\theta_5, \lambda_5) = (2.2,0.5)$, $(\theta_6, \lambda_6) = (2.2,0.9)$, $(\theta_7, \lambda_7) = (2.2, 1.5)$ and $(\theta_8, \lambda_8) = (2.2, 5)$.

From Figure 1, it is observed that when the value of shape parameter (θ) is set to 1, the curve becomes a decreasing function. It is also noticed that the density of G.E is unimodal and more importantly, for a fixed scale parameter (λ), as the θ increases it becomes more and more symmetric.

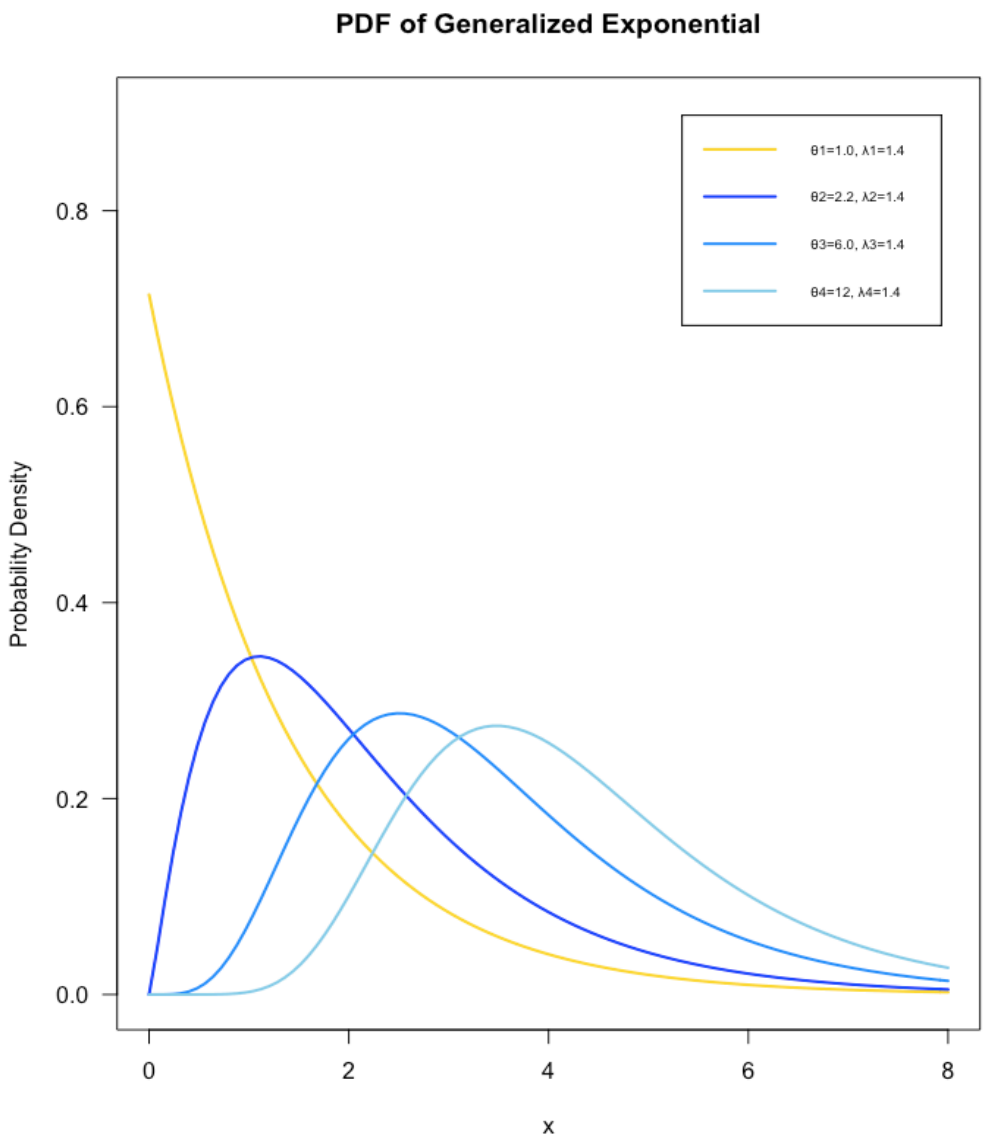


Figure 1. PDF plot for generalized exponential distribution for fixed value of scale parameter.

From Figure 2, it is noticed that for a fixed shape parameter (θ), increasing scale parameter (λ) makes the curve flatter and stretches out.

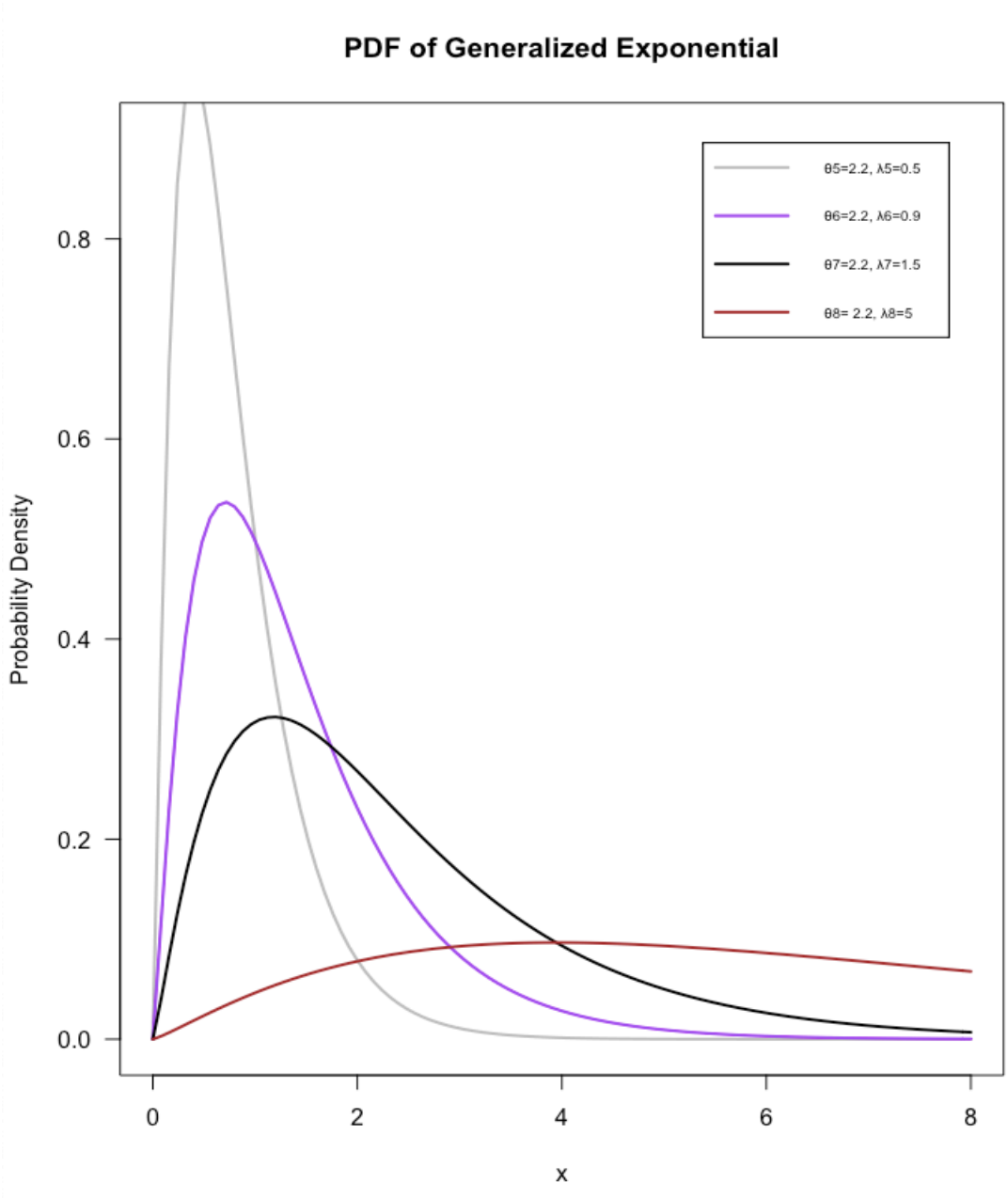


Figure 2. PDF plot for generalized exponential distribution for fixed value of shape parameter.

1.4 Literature Review

The literature review aims to understand and to be aware of the various researches done in the area of the study. Over the years, many kinds of research have been conducted regarding life test models for various distributions. Exponential, Weibull, gamma and lognormal are among the most popular lifetime distributions used to model the failure time and reliability data. The main goal of these studies is to derive the estimators and evaluate these estimators for the life test models using different techniques and several censoring schemes.

There are several studies based on the accelerated life test, Ismail (2004) discusses the inferential procedures (the point and interval maximum-likelihood estimations) for step-stress partially accelerated life test model for the Weibull distribution using tampered random variable model for the effect of the acceleration. The estimation is performed assuming Type-I progressively hybrid censored data and the performance is assessed based on MSE and Bias via simulation. Another study by Balakrishnan et al. (2009) examined the maximum likelihood estimators for the unknown parameters under the simple step stress model for the log-normal distribution with type I censoring scheme. They used the cumulative exposure model to account for the step stress acceleration. They derived the confidence intervals for the unknown parameters based on the asymptotic normality of the MLEs and parametric Bootstrap resampling technique. Additionally, they evaluate the performance of these inferential methods by simulations. Furthermore, Step stress accelerated test with tampered random variable model, tampered failure rate model, cumulative exposure model was conducted by Zarrin et al. (2010), Wang et al (2012) Sun and Shi (2016) respectively. Zarrin et al. (2010) assumed the life-time distribution to be 3 parameter exponentiated Weibull distribution with no censoring and they derived the estimators. Wang et al. (2012) used the geometric distribution with

progressive type-II censored data where they derived and evaluated the MLEs. Sun and Shi (2016) considered the Birnbaum-Saunders distribution with a type II censoring scheme. They derived estimators using maximum likelihood estimation and Bayesian estimation and these were evaluated and compared.

There are several studies focused on the inferential procedures for the exponential distribution. Childs et al. (2002) derived the MLE for exponential distribution with Type-I and type-II hybrid censored samples. In the same year, Wu (2002) derived the MLE of two-parameter exponential distribution on step stress accelerated test model with a cumulative exposure model with a type II censoring scheme. The MLE in this study were not evaluated nor compared to another estimator. However, Balakrishnan et al. (2007) derived the maximum likelihood estimators (MLEs) of the parameters assuming a cumulative exposure model for the exponential distribution. The exact distributions of the MLEs of parameters are obtained using conditional moment-generating functions. Furthermore, they derived confidence intervals for the unknown parameters using these exact distributions, asymptotic distributions, and the parametric Bootstrap method and then evaluated their performance through a Monte Carlo simulation study. Nine years later, Shafay (2016) studied the same model but with a generalized type I hybrid censoring scheme and conducted inferential procedures using maximum likelihood and Bayesian estimation. He also derived confidence intervals for the parameters using exact distributions, asymptotic distributions of the MLEs, Bayesian, and the parametric Bootstrap methods. Further, he evaluated the performance and compared the estimators and the intervals.

Likewise, several inferential procedures have been conducted for the generalized exponential distribution with and without the consideration of accelerated life test model using many estimation methods. Gupta and Kundu (2001), Raqab and Madi (2005), Mitra and Kundu (2008), Raqab and Madi (2009), Kundu and Pradhan (2009), Yameen and Aslam (2013) and El-Din et al. (2017) derived the MLE for the different censoring scheme but did not study the step stress accelerated model. Most of these studies derived the Bayesian estimators using Gibbs sampling technique and compared it to the MLEs. On the other hand, Ismail (2012), Jaheen et al. (2014) and Zheng and Shi (2015) studied the accelerated life test for the generalized exponential distribution and derived estimators for the same using various censoring techniques. Ismail (2012) worked on the step-stress partially accelerated life tests under progressive type-II censoring scheme with binomial removal of items at each failure time. He also investigated MSE and variance for the MLEs. Using these models, he derived the MLEs for point estimation and interval estimation. Jaheen et al. (2014) and Zheng and Shi (2015) worked on the constant stress accelerated life test model for the generalized exponential distribution with progressive type-II censoring scheme and progressive Type-I hybrid censoring schemes respectively. Jaheen et al. (2014) derived the MLEs and Bayesian estimators, whereas Zheng and Shi (2015) derived the MLE along with the confidence intervals using asymptotic normality of MLEs and percentile Bootstrap method. Finally, these methods were assessed and compared using Monte-Carlo method simulation method.

As seen from the literature review, maximum likelihood estimation under the progressive type I censoring scheme for the generalized exponential distribution with tampered random variable model as not been addressed so far.

1.5 Objective of the Study

The main objectives of this research are to derive and evaluate the maximum likelihood estimators of the scale and shape parameters and reliability function for the generalized exponential distribution under step stress partially accelerated model with progressive type 1 hybrid censoring scheme. After estimating the parameters, to evaluate the performances of these estimators. Additionally, to derive the confidence intervals for these unknown parameters and to evaluate them. Further, to apply and analyze these inferential procedures on real data or simulated data.

The generalized exponential distribution is proposed by Gupta and Kundu in 1999 which makes it relatively new distribution as compared to other distributions and hence it is not fully explored yet. As seen from the literature review, SSPALT has been studied for many lifetime distributions based on various censoring schemes. Unfortunately, inference in an accelerated life test model for the generalized exponential distribution is not very popular in these works of literature. Most of the literature is regarding other distributions that have been studied under SSPALT or CALT with a censoring scheme or the studies related to generalized exponential distribution but without any accelerated life test. This thesis proposes to fill this gap by deriving maximum likelihood estimators for the unknown parameters of the generalized exponential distribution with SSALT plan with a progressive hybrid-I censoring scheme. These estimators will be derived, and their performance will be evaluated based on Bias and MSE. Further confidence intervals will be derived using asymptomatic method and Bootstrap methods and compared based on the coverage probability and expected length.

As Soliman (2002) indicates, estimation of the reliability function of some equipment is one of the main problems of reliability theory. Finding the reliability function and finding the estimators will solve this problem. In most practical

applications and life-test experiments, the distributions with positive domain are appropriate for modeling. As generalized exponential distribution has a positive domain, it would be a good fit for a reliability study.

Alizadeh et al. (2015) studied the different estimators for the parameter of generalized exponential distribution like uniformly minimum variance unbiased estimator, maximum likelihood estimator, percentile estimator, least squares estimator, weighted least squares estimator, and moments estimator and compared these estimators based on the bias and the mean squared error. They found MLE to be the better estimator in terms of bias and MSE. Hence, we are considering only MLE when studying the parameter estimates under the SSPALT model. We chose SSPALT with hybrid progressive 1 censoring as it guarantees that there are r pre-defined failures, or the experiment terminates after a per-defined time η. More importantly, this study has not been addressed so far for the generalized exponential distribution.

CHAPTER 2: INFERENCE FOR STEP STRESS ACCELERATED LIFE TEST IN THE GENERALIZED EXPONENTIAL DISTRIBUTION

Assume that the random variable x represents the lifetime of a product which are identical and independent and has generalized exponential distribution with the scale and shape parameters as λ and θ, respectively.

The probability density function (pdf) of x is

$$f(x) = \frac{\theta}{\lambda} e^{-\frac{x}{\lambda}} \left(1 - e^{-\frac{x}{\lambda}}\right)^{\theta-1}, \qquad x > 0, \qquad \theta \,\&\, \lambda > \qquad [3]$$

In order to account for the effect of stress change on the remaining lifetime of the testing units of the step stress accelerated model, we have applied the tampered random variable model (TRVM) which was introduced by DeGroot and Goel (1979). In this model, the low stress is actually the normal use condition of the units. (Xu and Fei, 2012)

$$X = \begin{cases} T, & if \quad T \leq \tau \\ \tau + \beta^{-1}(T - \tau), & if \quad T > \tau \end{cases} \qquad [4]$$

where T is the lifetime of the unit under use condition, τ is the stress change time and β is the acceleration factor; β > 1

The pdf of x (generalized exponential distribution) under SSPALT is shown as below:

$$f(x) = \begin{cases} 0, & x \leq 0 \\ f_1(x), & 0 < x \leq \tau \\ f_2(x), & x > \tau \end{cases} \qquad [5]$$

where

$$f_1(x) = \frac{\theta}{\lambda} e^{-\frac{x}{\lambda}} \left(1 - e^{-\frac{x}{\lambda}}\right)^{\theta-1}$$

$$f_2(x) = \beta \frac{\theta}{\lambda} e^{\frac{-1}{\lambda}[\tau + \beta(x-\tau)]} \left(1 - e^{\frac{-1}{\lambda}[\tau + \beta(x-\tau)]}\right)^{\theta-1}$$

2.1 Timeline of Experiment Under SSPALT

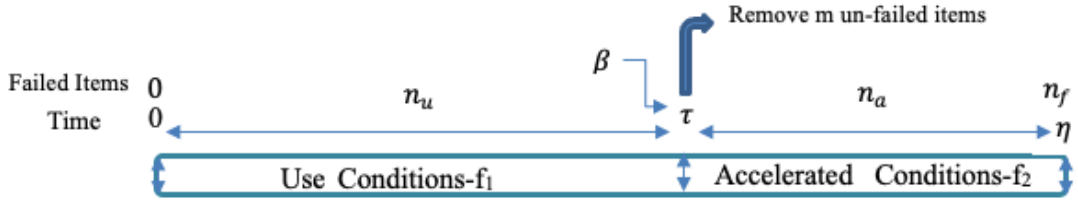


Figure 3. Timeline of experiment under SSPALT with type I progressively hybrid censoring scheme.
Created by author.

τ – Stress change time & β– acceleration factor.

m- un-failed units that are randomly removed at time τ (m & τ are prefixed)

η - time of experiment termination

n- no. of units under experiment

n_u -no. of units that fail before τ

n_a- no. of units that fail before η at accelerated condition

n_f- no. of units that fail before the experiment is terminated

n_f-can be of different value depending on the failure time of items as follow

$$n_f = \begin{cases} n_u + n_a = r, & x_{r:n} \le \eta \\ n_u + n_a < r, & x_{r:n} > \eta \end{cases}$$

Under the progressively Type-I hybrid censoring scheme, there are two scenarios as below:

$$Set1: x1{:}n < \cdots < xn_{\mathrm{u}}{:}n \le \tau < xn_{\mathrm{u}}+1{:}n < \cdots < xr{:}n \le \eta, \quad \text{if } xr{:}n \le \eta \qquad [6]$$

$$Set\ 2: x1{:}n < \cdots < xn_{\mathrm{u}}{:}n \le \tau < xn_{\mathrm{u}}+1{:}n < \cdots < xn_{\mathrm{u}}+n_{\mathrm{a}}{:}n \le \eta, \quad \text{if } xr{:}n > \eta \qquad [6]$$

2.1 Likelihood Construction

This thesis is considering the progressive hybrid censoring scheme, there are two scenarios and hence constructing the likelihood function for each scenario. The construction involves the pdf and the survival functions as follows:

2.1.1 Likelihood Function for Scenario 1

$$Set1: x_{1:n} < \cdots < x_{n_{\mathrm{u}}:n} \le \tau < x_{n_{\mathrm{u}}+1:n} < \cdots < x_{r:n} \le \eta, \quad \text{if } x_{r:n} \le \eta$$

$$L(\lambda, \theta, \beta) \propto \prod_{i=1}^{n_u} f_1(x_i)\,[S_1(\tau)]^m \prod_{i=n_u+1}^{r} f_2(x_i)\,[S_2(x_{r:n})]^{n-r-m}$$

Where $S_1(y) = 1 - \left(1 - e^{-x/\lambda}\right)^{\theta}$ (Survival function at use conditions)

$S_2(y) = 1 - \left(1 - e^{\frac{-1}{\lambda}[\tau + \beta(x-\tau)]}\right)^{\theta}$ (Survival function at accelerated conditions)

Survival function of generalized exponential:

$$S(t) = P(T > t)$$

$$= 1 - \left(1 - e^{-t/\lambda}\right)^{\theta}, \quad t > 0$$

$$L(\lambda,\theta,\beta) \propto \prod_{i=1}^{n_u} f_1(x_i)\,[S_1(\tau)]^m \prod_{i=n_u+1}^{r} f_2(x_i)\,[S_2(x_{r:n})]^{n-r-m}$$

$$L(\lambda,\theta,\beta) \propto \prod_{i=1}^{n_u}\left[\frac{\theta}{\lambda}e^{-\frac{x_i}{\lambda}}\left(1-\ e^{-\frac{x_i}{\lambda}}\right)^{\theta-1}\right].\ \left(1-\left(1-e^{-\tau/\lambda}\right)^{\theta}\right)^{m}$$

$$\prod_{i=n_u+1}^{r}\left[\beta\frac{\theta}{\lambda}e^{\frac{-1}{\lambda}[\tau+\beta(x_i-\tau)]}\left(1-e^{\frac{-1}{\lambda}[\tau+\beta(x_i-\tau)]}\right)^{\theta-1}\right]\left(1-\left(1-\right.\right.$$

$$\left.\left.e^{\frac{-1}{\lambda}[\tau+\beta(x_{r:n}-\tau)]}\right)^{\theta}\right)^{n-r-m}$$

$$\ln L(\lambda,\theta,\beta)\ = n_u\ln\theta - n_u\ln\lambda - \frac{\sum_{i=1}^{n_u}x_i}{\lambda} + (\theta-1)\sum_{i=1}^{n_u}\ln\left(1-e^{-\frac{x_i}{\lambda}}\right) +$$

$$m\ln\left(1-\left(1-e^{-\frac{\tau}{\lambda}}\right)^{\theta}\right) + (r-n_u)\ln\beta + (r-n_u)\ln\theta - (r-n_u)\ln\lambda -$$

$$\frac{1}{\lambda}\sum_{i=n_u+1}^{r}[\tau+\ \beta(x_i-\tau)] + (\theta-1)\sum_{i=n_u+1}^{r}\ln\left(1-e^{\frac{-1}{\lambda}[\tau+\beta(x_i-\tau)]}\right) + (n-$$

$$r-m)\ln\left(1-\left(1-e^{\frac{-1}{\lambda}(\tau+\beta(x_{r:n}-\tau))}\right)^{\theta}\right) \qquad [7]$$

2.1.2 Likelihood Function for Scenario 2

Set 2: $x_{1:n} < \cdots < x_{n_u:n} \le \tau < x_{n_u+1:n} < \cdots < x_{n_u+n_a:n} \le \eta,$ if $x_{r:n} > \eta$

$$L(\lambda,\theta,\beta) \propto \prod_{i=1}^{n_u} f_1(x_i)\,[S_1(\tau)]^m \prod_{i=n_u+1}^{n_u+n_a} f_2(x_i)\,[S_2(\eta)]^{n-(n_u+n_a)-m}$$

$$L(\lambda,\theta,\beta) \propto \prod_{i=1}^{n_u}\left[\frac{\theta}{\lambda}e^{-\frac{x_i}{\lambda}}\left(1-e^{-\frac{x_i}{\lambda}}\right)^{\theta-1}\right].\left(1-\left(1-\right.\right.$$

$$\left.\left.e^{-\tau/\lambda}\right)^{\theta}\right)^{m}\prod_{i=n_u+1}^{n_u+n_a}\left[\beta\frac{\theta}{\lambda}e^{\frac{-1}{\lambda}[\tau+\beta(x_i-\tau)]}\left(1-e^{\frac{-1}{\lambda}[\tau+\beta(x_i-\tau)]}\right)\theta-1\right].\left(1-\left(1-\right.\right.$$

$$\left.\left.e^{\frac{-1}{\lambda}[\tau+\beta(\eta-\tau)]}\right)^{\theta}\right)^{n-(n_u+n_a)-m}$$

$$\ln L(\lambda,\theta,\beta) = n_u\ln\theta - n_u\ln\lambda - \frac{\sum_{i=1}^{n_u}x_i}{\lambda} + (\theta-1)\sum_{i=1}^{n_u} ln\left(1-e^{-\frac{x_i}{\lambda}}\right) +$$

$$m\ln\left(1-\left(1-e^{-\frac{\tau}{\lambda}}\right)^{\theta}\right) + n_a\ln\beta + n_a\ln\theta - n_a\ln\lambda - \frac{1}{\lambda}\sum_{i=n_u+1}^{n_u+n_a}[\tau +$$

$$\beta(x_i-\tau)] + \sum_{i=n_u+1}^{n_u+n_a}\ln\left(1-e^{\frac{-1}{\lambda}[\tau+\beta(x_i-\tau)]}\right)(\theta-1) + (n-(n_u+n_a)-$$

$$m)\ln\left(1-\left(1-e^{\frac{-1}{\lambda}[\tau+\beta(\eta-\tau)]}\right)^{\theta}\right) \quad [8]$$

2.3 Likelihood Inference of the Distribution Parameters

Maximum likelihood (ML) methods provide estimates and confidence limits for model parameters, a product life distribution at a design stress, and other quantities of interest. It also provides check on the validity of the model and data.

The ML is a method that uses likelihood function which is known as joint probability function of random variables $X_1, X_2 \ldots X_n$. Likelihood function is denoted by $L(x,\emptyset) = f(x_1, x_2 \ldots x_n)$ where $x_1, x_2 \ldots x_n$are the values of the random sample from a population with unknown parameter $\emptyset$. The maximum likelihood estimator is found by maximizing the likelihood function with respect to $\emptyset$ and the value that maximizes this likelihood function is known as the maximum likelihood estimator (MLE) of $\emptyset$ and is denoted by $\widehat{\emptyset}$. The first partial derivative of the log likelihood function is usually required to find the MLE and this is known as the score. And the variance of the score is known as Fisher information matrix which is defined as the negative of the second derivative of the log-likelihood function. It is therefore also

known as the expected value of the observed information. Fisher information matrix is denoted by

$$I(\emptyset) = E\left[-\frac{\partial^2}{\partial \emptyset^2} ln\, L(\underline{x}; \emptyset)\right] \qquad \text{where } 0 \leq I(\emptyset) < \infty$$

In this thesis, the MLE for the G.E distribution parameters $\lambda, \theta\ and\ \beta$ are found by simultaneously solving the non-linear Equations [7] and [8] as these can't be found analytically in closed forms. Hence Optim function in R program is applied to find the MLEs and Fisher information matrix, this function uses the Nelder–Mead method which is a commonly applied numerical method used to find the minimum or maximum of an objective function in a multidimensional space.

Additionally, the variance of these MLEs can be found using the inverse of Fisher Information matrix and these can be used to approximate confidence intervals of the unknown parameters $\phi = (\lambda, \theta, \beta)$ which are derived based on the asymptotic distribution of the MLEs $(\hat{\lambda}, \hat{\theta}, \hat{\beta})$.

It is known that the asymptotic distribution of the MLE is given as below:

$$\left((\hat{\lambda} - \lambda), (\hat{\theta} - \theta), (\hat{\beta} - \beta)\right) \rightarrow N\left(0, I^{-1}(\lambda, \theta, \beta)\right)$$

where $I^{-1}(\lambda, \theta, \beta)$ is the variance-covariance matrix of the unknown parameters $\phi = (\lambda, \theta, \beta)$. The elements of this matrix $I_{ij}(\lambda, \theta, \beta),\ i, j = 1,2,3$ can be estimated approximately by $I^{-1}{}_{ij}(\hat{\lambda}, \hat{\theta}, \hat{\beta})$ where

$$I_{ij}(\hat{\phi}) = -\frac{\partial^2 \ln L(\phi)}{\partial \phi_i\, \partial \phi_j}\bigg|_{\phi = \hat{\phi}}$$

Thus, the approximate $100(1-\alpha)\%$ two-sided confidence intervals for ϕ are given by,

$$\hat{\lambda} \pm Z_{\alpha/2}\sqrt{{I_{11}}^{-1}\left(\hat{\lambda},\hat{\theta},\hat{\beta}\right)}\ ,$$

$$\hat{\theta} \pm Z_{\alpha/2}\sqrt{{I_{22}}^{-1}\left(\hat{\lambda},\hat{\theta},\hat{\beta}\right)}\ , \qquad [9]$$

$$\hat{\beta} \pm Z_{\alpha/2}\sqrt{{I_{33}}^{-1}\left(\hat{\lambda},\hat{\theta},\hat{\beta}\right)}\ ,$$

where $Z_{\alpha/2}$ is the upper $\left(\alpha/2\right)^{th}$ percentile of the standard normal distribution.

Further, MLE has an important and very useful property which is known was invariance property of MLE. (Casella & Berger, 2002). Simply given as if the MLE of ϕ is $\hat{\phi}$, then for any function $\gamma(\phi)$, the MLE of $\gamma(\phi)$ is $\gamma\left(\hat{\phi}\right)$.

Additionally, asymptotic normality is defined as below, it is said that $\hat{\phi}$ is asymptotically normal if

$$\sqrt{n}\left(\hat{\phi}-\phi_0\right) \xrightarrow{D} N\left(0,\sigma_{\phi_0}^2\right)$$

where

$\sigma_{\phi_0}^2$: is known to be the asymptotic variance of the estimator $\hat{\phi}$.

ϕ_0 : is known to be the true parameter value of ϕ.

Suppose that $\hat{\phi}_{MLE}$ converges in probability to ϕ_0, then it is said that $\hat{\phi}_{MLE}$ is consistent and consistency is a desirable property of an estimator.

(Lehmann, E., 1998) Generally, let a statistical model $\{f(x\,,\phi)\colon \phi \in \Theta\}$ of probability density function (pdf) or probability mass function (pmf) on $X \subseteq R^d$ is satisfied along with the consistency assumption. Then if we have:

1- The true value of $\phi_0 \in \Theta$.

2- There exists $U \subseteq \Theta$, such that function $\theta \to f(x\,,\phi)$, $\forall\, x \in X$ which is twice continuously differentiable with respect to $\phi \in$ U.

3- A $p \times p$ non-singular Fisher information matrix $I(\phi_0)$ and $E_{\phi_0}\left[\left|\left|\nabla_\phi log f(x,\phi_0)\right|\right|\right] < \infty$ is satisfied.

4- A compact ball $W \subseteq U$ of a non-empty interior exists, which is centered at ϕ_0, such that $E_{\phi_0} sup_{\phi \in W}\left[\left|\left|\nabla_\theta^2 log f(x,\phi)\right|\right|\right] < \infty$,

$$\int_x sup_{\phi \in W}\left|\nabla_\phi log f(x,\phi_0)\right| d\phi < \infty ,$$

$$\int_x sup_{\phi \in W}\left|\nabla_\phi^2 log f(x,\phi)\right| d\phi < \infty ,$$

Related to the assumptions above with their properties, then asymptotic normality of the MLE must hold.

(Asymptotic normality). Under the above assumptions for any choice of ϕ_n satisfying $\phi_n \in \text{Argmax} l_n(\phi)$ the sequence $\{ n(\hat{\phi}_n - \phi)\}_{n \in Z_+}$ converges in **P** -distribution to a centered Gaussian random vector with covariance matrix $\Sigma^{-1}{}_\phi$. (Falconnet et al., 2014). Hence the Asymptotic normality of the MLE can be given as below:

Let $X_1, X_2, X_3, ...$, be identically independent distributed (iid) for $f(x|\phi)$ and $\hat{\phi}$ be the MLE of ϕ . Then,

$$\sqrt{n}\left(\hat{\phi} - \phi_0\right) \to N\left(0,\ I^{-1}(\phi_0)\right)$$

This theorem is used to construct the asymptotic confidence intervals for the unknown model parameters as well as the reliability functions.

Further, assume that we have a sequence of random variables H_n such that

$$\sqrt{n}(H_n - \phi) \xrightarrow{D} N(0, \sigma^2) \text{ as } n \to \infty$$

where ϕ and σ^2 are finite valued constants. Let $g(\phi)$ be a continuous function such that $\acute{g}(\phi) \neq 0$ and it exists then

$$\sqrt{n}\left(g(H_n) - g(\phi)\right) \xrightarrow{D} N\left(0, \sigma^2[\acute{g}(H_n)]^2\right)$$

The above result is known as the univariate delta method and similarly multivariate delta method can be applied for multivariate data which is given in Section 2.5. (Gugushvili, 2014)

(Multivariate Delta method) (Gugushvili, 2014) Suppose we have a multiparameter vector of differentiated parameters $\tau = g(\phi_1, \dots, \phi_k)$ and it can be differentiated as

$$\nabla g = \begin{pmatrix} \frac{\partial g}{\partial \phi_1} \\ \vdots \\ \frac{\partial g}{\partial \phi_k} \end{pmatrix},$$

and let $\hat{\tau} = g(\hat{\phi})$. Then $\hat{\tau} - \tau \to N\left(0, \left(\hat{\nabla} g\right)^T \widehat{I^{-1}}_n \left(\hat{\nabla} g\right)\right)$,

where $I_n^{-1}(\phi)$ be the inverse of the Fisher information matrix $I_n(\theta)$, with $\widehat{I^{-1}}_n = I_n^{-1}(\hat{\phi})$ and $\hat{\nabla} g = \nabla g(\hat{\phi})$.

2.4 Distribution Parameters Estimate Evaluation

In the previous Section 2.3, the MLE of the distribution parameters were derived using numerical methods and asymptotic confidence interval is constructed using the Fisher information matrix and the delta method. In this Section, the performance of these point and interval estimators will be evaluated using the below evaluation methods.

2.4.1 Point estimator evaluation

The performance of the MLE will be assessed based on the bias and MSE as follows:

- Bias

Bias is the expected value of the difference between the parameter and its estimate. An estimator is considered unbiased estimator of the parameter when its bias value is equal to zero and unbiases is a desirable quality of an estimator. In general, bias is denoted as below:

$$Bias(\widehat{\omega}) = E(\widehat{\omega} - \omega)$$

Bias for the MLE of the distribution parameters is computed as below:

$$Bias(\hat{\lambda}) = E(\hat{\lambda} - \lambda) \quad [10]$$

$$Bias(\hat{\theta}) = E(\hat{\theta} - \theta) \quad [11]$$

$$Bias(\hat{\beta}) = E(\hat{\beta} - \beta) \quad [12]$$

- MSE

MSE which is the mean square error; is the expected value of the difference square between parameter and its estimate. It is a non-negative value and the estimator is considered better as the value of MSE is close to zero. MSE is also known as risk factor and is a measure of quality and it is denoted as below:

$$MSE(\widehat{\omega}) = E(\widehat{\omega} - \omega)^2$$

MSE for the MLE of the parameters is computed as below:

$$MSE(\hat{\lambda}) = E(\hat{\lambda} - \lambda)^2 \quad [13]$$

$$MSE(\hat{\theta}) = E(\hat{\theta} - \theta)^2 \quad [14]$$

$$MSE(\hat{\beta}) = E(\hat{\beta} - \beta)^2 \qquad [15]$$

2.4.2 Interval Estimator Evaluation

- Coverage probability

Coverage probability is the number of the times the true parameter value lies within the confidence intervals which are computed using the Monte Carlo simulation method. The value of coverage probability understandably lies between 0 and 1 and the interval estimator is considered better when the coverage probability is close to 1. It is computed as below:

$$\boldsymbol{CP = \frac{\#\ of\ interval\ that\ covers\ parameter}{Total\ number\ of\ Intervals}}$$

- Expected length

Expected length is another measure used to evaluate the performance of the estimated interval. An estimated interval is desired to have a shorter expected length. It is computed as the expected value of the length of the interval which is the difference between the upper limit and lower limit of the interval, and this is calculated as below.

$$Expected\ Length = \frac{\sum_{i=1}^{K}(U_i - L_i)}{K}\ ,\quad where\ K: Total\ no\ of\ intervals$$

2.5 Inference about the Reliability Function

Reliability function at time t is defined as the probability of surviving until time t. The reliability function; $R(t)$ for G.E under SSPLAT with progressive hybrid type I censoring scheme is given by the below two models.

Model 1: $t < \tau$

$$R_1(t) = 1 - \left(1 - e^{\frac{-t}{\lambda}}\right)^{\theta},\quad t < \tau, \lambda\ \&\ \theta > 0$$

which is estimated by using the invariance property of the MLE ($\hat{\lambda}\ and\ \hat{\theta}$) (Casella

and Berger, 2002)

$$\hat{R}_1(t) = 1 - \left(1 - e^{\frac{-t}{\hat{\lambda}}}\right)^{\hat{\theta}} \qquad [16]$$

Model 2: $\tau < t < \eta$

$$R_2(t) = 1 - \left(1 - e^{\frac{-1}{\lambda}(\tau + \beta(t-\tau))}\right)^{\theta}, \qquad \tau < t < \eta\,, \lambda, \theta\ \&\ \beta > 0$$

which is estimated by invariance property of the MLE $\left(\hat{\lambda}, \hat{\theta}\text{ and }\hat{\beta}\right)$ (Casella and Berger, 2002)

$$\hat{R}_2(t) = 1 - \left(1 - e^{\frac{-1}{\hat{\lambda}}\left(\tau + \hat{\beta}\,(t-\tau)\right)}\right)^{\hat{\theta}} \qquad [17]$$

The confidence interval estimation for the reliability function is calculated using the asymptotic distribution as below:

$$\sqrt{n}\left(\hat{R} - R\right) \xrightarrow{D} N\left(0, Var(R)\right)$$

The above relation is based on the theorem of asymptotic efficiency of MLEs (Casella and Berger, 2002) which is stated in Section 2.3.

The variance for the estimated reliability function is derived using multivariate delta method (Gugushvili, 2014) which is stated in Section 2.3 is applied as follows:

Model 1: When $t < \tau$

$$Var\left(\hat{R}_1(t)\right) = \hat{\nabla}{R_1}^T \begin{bmatrix} Var(\hat{\lambda}) & Cov(\hat{\theta}, \hat{\lambda}) \\ Cov(\hat{\lambda}, \hat{\theta}) & Var(\hat{\theta}) \end{bmatrix} \hat{\nabla} R_1,$$

$$where\ \hat{\nabla} R_1 = \begin{bmatrix} \frac{\partial \hat{R}_1(t)}{\partial \hat{\lambda}} \\ \frac{\partial \hat{R}_1(t)}{\partial \hat{\theta}} \end{bmatrix} \qquad [18]$$

$$\frac{d\hat{R}_1}{d\hat{\lambda}} = \frac{\hat{\theta}t\left(1 - e^{\frac{-t}{\hat{\lambda}}}\right)^{\hat{\theta}}}{\hat{\lambda}^2\left(e^{\frac{t}{\hat{\lambda}}} - 1\right)} \quad [19]$$

$$\frac{d\hat{R}_1}{d\hat{\theta}} = -\left(1 - e^{\frac{-t}{\hat{\lambda}}}\right)^{\hat{\theta}} ln\left(1 - e^{\frac{-t}{\hat{\lambda}}}\right) \quad [20]$$

The asymptotic variance for MLE of $\lambda \, and \, \theta$ are derived from the inverse Fisher information matrix I as shown in Section 2.3.

Model 2: When $\tau < t < \eta$

$$Var\left(\hat{R}_2(t)\right) = \hat{\nabla}{R_2}^T \begin{bmatrix} Var(\hat{\lambda}) & Cov(\hat{\theta},\hat{\lambda}) & Cov(\hat{\beta},\hat{\lambda}) \\ Cov(\hat{\lambda},\hat{\theta}) & Var(\hat{\theta}) & Cov(\hat{\beta},\hat{\theta},) \\ Cov(\hat{\lambda},\hat{\beta}) & Cov(\hat{\theta},\hat{\beta}) & Var(\hat{\beta}) \end{bmatrix} \hat{\nabla}R_2,$$

$$where \, \hat{\nabla}R_2 = \begin{bmatrix} \frac{\partial \hat{R}_2(t)}{\partial \hat{\lambda}} \\ \frac{\partial \hat{R}_2(t)}{\partial \hat{\theta}} \\ \frac{\partial \hat{R}_2(t)}{\partial \hat{\beta}} \end{bmatrix} \quad [21]$$

$$\frac{d\hat{R}_2}{d\hat{\lambda}} = \frac{\hat{\theta}\left(\hat{\beta}\,(t-\tau)+\tau\right)\left(1 - e^{\frac{-1}{\hat{\lambda}}\left(\hat{\beta}\,(t-\tau)+\tau\right)}\right)^{\hat{\theta}-1} e^{\frac{-1}{\hat{\lambda}}\left(\hat{\beta}(t-\tau)+\tau\right)}}{\hat{\lambda}^2} \quad [22]$$

$$\frac{d\hat{R}_2}{d\hat{\theta}} = -\left(1 - e^{\frac{-1}{\hat{\lambda}}(\hat{\beta}\,(t-\tau)+\tau)}\right)^{\hat{\theta}} \qquad [23]$$

$$\frac{d\hat{R}_2}{d\hat{\beta}} = \frac{-\left(\hat{\theta}(t-\tau)\right)\left(1 - e^{\frac{-1}{\hat{\lambda}}(\hat{\beta}\,(t-\tau)+\tau)}\right)^{\hat{\theta}}}{\hat{\lambda}\left(e^{\frac{1}{\hat{\lambda}}(\hat{\beta}\,(t-\tau)+\tau)} - 1\right)} \qquad [24]$$

The asymptotic variance for MLE of $\lambda, \theta \; and \; \beta$ are derived from the inverse Fisher information matrix I as shown in Section 2.3.

2.5.1 Transformation of Reliability Function

Reliability function is a function that gives the probability of survival after a certain point in time. This makes the range of reliability function lie between 0 and 1 and consequently the estimated reliability function $\hat{R}(t)$ lies between 0 and 1. Therefore, this function is expected to converge slower to Normal distribution when constructing the asymptotic confidence interval for the reliability function. However, a log transformation of the estimated reliability function denoted by $\hat{R}$*(t) is considered, and this is expected to converge faster to normal distribution as its range is from -infinity to 0. Thus, the asymptotic confidence interval for $\hat{R}$*(t) is calculated and then it is transformed back to $\hat{R}(t)$, this new transformed reliability function is denoted by $\hat{R}^{**}$(t) which is computed by exponentiating the lower and upper bound of the asymptotic confidence interval for $\hat{R}$*(t).

$$\hat{R}^{*}(\mathrm{t}) = \log\left(\frac{\hat{R}(t)}{1-\hat{R}(t)}\right), \qquad \mathbf{0} < \hat{R}^{*}(\mathrm{t}) < -\infty \qquad [25]$$

Asymptotic confidence interval for $\hat{R}^*(t)$ is given by

$$\hat{R}^*(t) \pm z_{\alpha/2}\sqrt{Var\left(\hat{R}^*(t)\right)} \qquad [26]$$

The variance of $\hat{R}^*(t)$ is derived using the delta method as below:

$$\sqrt{n}\left(\log\left(\frac{\hat{R}(t)}{1-\hat{R}(t)}\right) - \log\left(\frac{R(t)}{1-R(t)}\right)\right) \rightarrow N\left(0, Var\left(\hat{R}^*(t)\right)\right) \qquad [27]$$

$$\begin{aligned} Var\left(\hat{R}^*(t)\right) &= Var\left(\hat{R}(t)\right)\left(\frac{d\hat{R}^*(t)}{d\hat{R}(t)}\right)^2 \downarrow_{\hat{R}(t)} \qquad where\ \frac{d\hat{R}^*(t)}{d\hat{R}(t)} \\ &= \frac{-1}{\left(\hat{R}(t)-1\right)\hat{R}(t)} \end{aligned} \qquad [28]$$

The asymptotic variance of estimated reliability function $Var\left(\hat{R}^*(t)\right)$ is calculated using the estimated MLEs as shown in Section 2.5.

Asymptotic confidence interval for $\hat{R}^{**}(t)$ is calculated as below:

$$\left(e^{Lower\,\hat{R}^*(t)}, e^{Upper\,\hat{R}^*(t)}\right)$$

$$\left(e^{\hat{R}^*(t)-z_{\alpha/2}\sqrt{Var\left(\hat{R}^*(t)\right)}}, e^{\hat{R}^*(t)+z_{\alpha/2}\sqrt{Var\left(\hat{R}^*(t)\right)}}\right) \qquad [29]$$

2.5.2 Reliability Function Estimate Evaluation

In the previous Section, the reliability function is estimated using the invariance property of the MLEs of the distribution parameters and asymptotic confidence interval is constructed. In this Section, the performance of these point and interval estimators will be evaluated using the below evaluation methods.

2.5.2.1 Point Estimator Evaluation

The performance of the reliability function estimator $\hat{R}(t)$ and $\hat{R}^{**}(\mathrm{t})$ will be evaluated based on the bias and MSE which is similar to Section 2.4.1.

- ***Bias***

Bias for reliability function estimator:

$$Bias\left(\hat{R}(\mathrm{t})\right) = E\left(\hat{R}(t) - R(t)\right) \quad [30]$$

Bias for reliability function estimator after transformation:

$$Bias\left(\hat{R}^{**}(\mathrm{t})\right) = E\left(\hat{R}^{**}(\mathrm{t}) - R(t)\right) \quad [31]$$

- MSE

MSE for reliability function estimator:

$$MSE\left(\hat{R}(\mathrm{t})\right) = E\left(\hat{R}(t) - R(t)\right)^2 \quad [32]$$

MSE for reliability function estimator after transformation:

$$MSE\left(\hat{R}^{**}(\mathrm{t})\right) = E\left(\hat{R}^{**}(\mathrm{t}) - R(t)\right)^2 \quad [33]$$

2.5.2.2 Interval Estimator Evaluation

Similar to point estimator evaluation, interval estimator evaluation is used to evaluate the performance of the asymptotic confidence intervals for the reliability function. This study considers the below evaluation methods for the assessment.

- Coverage probability

Coverage probabilities for $R(t)\ and\ \hat{R}^{**}(\mathrm{t})$ is computed in a similar method described in Section 2.4.2 using Monte Carlo simulation.

- Expected length

Expected length for $R(t)\ and\ \hat{R}^{**}(\mathrm{t})$ is computed in a similar method described in Section 2.4.2 using Monte Carlo simulation.

2.6 Bootstrap Interval Estimation

The following confidence intervals are based on the Bootstrap approach introduced by Efron and Tibshirani in 1986 (Efron and Tibshirani, 1986). These methods are computer intensive which are based on resampling from the original data and then using these estimated parameters to generate simulated Bootstrap samples to construct the required confidence interval. There are non-parametric and parametric Bootstrap methods. Non-parametric Bootstrap is used when the distribution of the data is unknown. On the other hand, when the parametric form of a distribution from which the data is generated is known with some unknown parameters, parametric Bootstrap methods are used. The unknown parameters of the distribution are estimated, and Bootstrap samples are generated using these estimated values and then Bootstrap intervals are constructed. There are various types of Bootstrap intervals that are discussed in the literature (Efron and Tibshirani, 1986). The most popular ones are, percentile interval, Bootstrap-t interval, Bias corrected and accelerated (BC_a) interval and these vary based on a series of increasingly less restrictive assumptions. This thesis focuses on parametric Bootstrap method, as the distribution is considered known which is the generalized exponential distribution.

2.6.1 Bootstrap Percentile Interval

In Bootstrap percentile interval method, the MLEs are calculated from the Bootstrap sample $(\hat{\lambda}, \hat{\theta}, \hat{\beta})$. Then Bootstrap distribution of $(\hat{\lambda}, \hat{\theta}, \hat{\beta})$ is simulated by resampling repeated B number of times from the parametric model of the original data

and $\hat{\lambda}^*, \hat{\theta}^*, \hat{\beta}^*$ are calculated for each Bootstrap sample.

Let $\widehat{K}$ be the cumulative distribution function of $\hat{\lambda}^*, \hat{\theta}^*, \hat{\beta}^*$, then the $(1-\alpha)$ interval is given by

$$\left(\widehat{K}^{-1}\left(\frac{\alpha}{2}\right), \widehat{K}^{-1}\left(1-\frac{\alpha}{2}\right)\right) \quad [34]$$

2.6.2 Bootstrap-t Interval

Let $\hat{\phi}$ be the maximum likelihood estimator of ϕ and let $\hat{\phi}^*$ be the estimator from the Bootstrap samples. $\hat{\phi}^*$ is the MLE obtained from the bootstrap sample.

Let ${z_\alpha}^*$ be the α quantile of the Bootstrap distribution of

$$Z^* = \left(\hat{\phi}^* - \hat{\phi}\right) / \sqrt{\hat{V}\left(\hat{\phi}^*\right)} \quad [35]$$

where $\hat{V}\left(\hat{\phi}^*\right)$ is the estimated variance of $\hat{\phi}$ from the Bootstrap sample. The Bootstrap interval for ϕ is given by

$$\hat{\phi} - {z^*}_{1-\alpha/2}\sqrt{\hat{V}\left(\hat{\phi}\right)}, \hat{\phi} - {z^*}_{\alpha/2}\sqrt{\hat{V}\left(\hat{\phi}\right)} \quad [36]$$

where ${z_\alpha}^*$ is obtained by simulation.

The above equations [35] and [36] are applied for the 3 unknown parameters λ, θ and β and these are used to construct the Bootstrap-t intervals.

CHAPTER 3: SIMULATION AND RESULTS

3.1 Simulation Design

A- The following simulation algorithm steps explain the way of generating a progressively censored type I data from generalized exponential distribution and computing the MLEs.

1- Specify the values of n, m, r, τ and η.

2- Specify the values of the parameters λ, θ and β.

3- Generate n independent random observations from uniform distribution; $n \sim Uniform\ (0,1)$ and sort the data.

4- Generate generalized exponential random variable using transformation as follows:

If U represents the uniform random variable from [0, 1]. Then U_1 and U_2 are selected such that U_1 consists of the uniform random numbers that are <= $F_1(\tau)$ and U_2 consists of the uniform random numbers that are >$F_1(\tau)$ and are <= $F(\eta)$. Then using inverse transformation $X_1 = \lambda \log\left(1 - U_1^{\frac{1}{\theta}}\right)$, X_1 has G.E with pdf given in Equation [3] if $x < \tau$. But if $x > \tau$ then using inverse transformation $X_2 = \frac{\tau\beta - \lambda \log\left(1 - U_2^{\frac{1}{\theta}}\right) - \tau}{\beta}$, X_2 has G.E with pdf given by Equation [5]. Further this is summarized in Table 1.

5- Then use the model in Equation [5] to generate progressively Type-I hybrid censored data for a given n, m, r, τ, η such that (η > τ), α, λ and β.

6- Two data sets can be considered as below

*Set*1: $x_{1:n} < \cdots < x_{n_u:n} \leq \tau < x_{n_u+1:n} < \cdots < x_{r:n} \leq \eta, \quad$ if $x_{r:n} \leq \eta$

Set 2: $x_{1:n} < \cdots < x_{n_u:n} \leq \tau < x_{n_u+1:n} < \cdots < x_{n_u+n_a:n} \leq \eta, \quad$ if $x_{r:n} > \eta$

7- Create function in R for the two likelihood functions; Equations [7] and [8].

8- Use the Optim function; which is found in a package in R to obtain the maximum likelihood estimators for unknown parameters (λ, θ, β) and put hessian=True to obtain the Fisher information matrix. The Optim function minimizes by nature, hence, to maximize the result obtained minus is added in likelihood function of Step 3.

9- Obtain Variance-Covariance matrix by computing the inverse of the Fisher information matrix.

10- Use the Variance-Covariance matrix to construct asymptotic confidence interval for the 3 unknown parameters; λ, θ, β using Equation [9].

Table 1. PDF, CDF and transformation

Probability Distribution Function	Cumulative Distribution Function	After Transformation
$\boldsymbol{f_1(x,\lambda,\theta) = \frac{\theta}{\lambda} e^{-\frac{x}{\lambda}}\left(1 - e^{-\frac{x}{\lambda}}\right)^{\theta-1}}$	$F_1(x,\lambda,\theta) = \left(1 - e^{-x/\lambda}\right)^{\theta}$	$x_1 = -\lambda \log\left(1 - U_1^{\frac{1}{\theta}}\right)$
$\boldsymbol{f_2(x,\lambda,\theta,\beta,\tau) = \beta \frac{\theta}{\lambda} e^{\frac{-1}{\lambda}[\tau+\beta(x-\tau)]}\left(1 - e^{\frac{-1}{\lambda}[\tau+\beta(x-\tau)]}\right)^{\theta-1}}$	$\boldsymbol{F_2(x,\lambda,\theta,\beta,\tau) = \left(1 - e^{-(\tau+\beta(x-\tau))/\lambda}\right)^{\theta}}$	$\boldsymbol{x_2 = \frac{\tau\beta - \lambda \log\left(1 - U_2^{\frac{1}{\theta}}\right) - \tau}{\beta}}$

B- The following is the simulation algorithm steps of to find Bootstrap percentile interval:

1- Use the MLEs $\left(\hat{\lambda}, \hat{\theta}, \hat{\beta}\right)$ obtained in the Section A to generate B Bootstrap samples.

2- Calculate the MLE for each of Bootstrap sample; $\left(\hat{\lambda}^*, \hat{\theta}^*, \hat{\beta}^*\right)$

3- Sort the values of $\hat{\lambda}^*, \hat{\theta}^*, \hat{\beta}^*$ separately.

4- Find the α^{th} quantile and $(1-\alpha)^{th}$ quantile of Bootstrap values in step 3 and call it L and U.

5- From the sorted data, find the value that corresponds to the values of L and U. These values are the lower bound and the upper bound of the confidence interval as in Equation [34].

C- The following is the simulation algorithm steps of to find Bootstrap-t:

1- Use the MLEs $(\hat{\lambda}, \hat{\theta}, \hat{\beta})$ obtained in the Section A to generate B Bootstrap samples.

2- Similar to Section B, calculate the MLE for each of Bootstrap sample; $(\hat{\lambda}^*, \hat{\theta}^*, \hat{\beta}^*)$

3- Find the $Z^*$for each of the Bootstrap estimators $\hat{\lambda}^*, \hat{\theta}^*, \hat{\beta}^*$ by calculating it as given in Equation [35].

4- Sort the values of Z^* for each of the $\hat{\lambda}^*, \hat{\theta}^*, \hat{\beta}^*$

5- Find the α^{th} quantile and $(1-\alpha)^{th}$quantile of sorted Z^*, these are values of ${z^*}_{1-\alpha/2}$ and ${z^*}_{\alpha/2}$.

6- Use the values obtained in Step 5 to calculate Equation [36].

Similarly, Bootstrap percentile interval and Bootstrap-t intervals for reliability function are obtained.

The below simulation designs are considered to obtain results and make conclusions about the MLEs.

Table 2. Simulation design

n	r	τ	η	t
	r=0.70n r=0.50n	R(τ)=0.70	R(η)=0.45 R(η)=0.10	R(t)=0.90,0.75,0.65,0.55
30	6, 15			0.1587378
50	10, 25			0.3783076
70	14, 35	0.4567534	0.6961452 1.466511	0.4980954
100	20, 50			0.5892076

Table 3. Parameter schemes for simulation

Scheme	Parameter $(r/n, \tau, \eta)$
1	0.50, 0.45, 0.69
2	0.70, 0.45, 0.69
3	0.70, 0.45, 1.44

3.2 Results

The performance of the estimators (MLEs) and estimated reliability functions for various values of t were examined by replicating the process 2000 times. For each generated sample, 95% confidence interval was constructed for each parameter and reliability function, the initial true value was checked if it falls inside this interval and the expected length of the interval was calculated. The coverage probability was measured as the count of intervals that contained the initial true value divided by 2000, while the expected lengths of the confidence were evaluated as mentioned in Section 2.4.2 where K is 2000.

The bias, MSE of λ, θ, β and R(t) parameters for each design are tabulated in Table 4 and Table 8. From the simulation study, it is observed that the MLEs for these

parameters are approximately unbiased and MSE decreases when increasing sample size n. For fixed $r/n, \tau \; and \; \eta$, bias decreases for λ, θ, β as n increases. Overall, the MSE and bias values are satisfactory for λ and θ, however for β, it is satisfactory when value of n increases. Thus, a minimum sample size of 50 is recommended for achieving satisfactory performances based on bias and MSE for these estimators.

In addition to estimating the unknown parameters, reliability function was evaluated for 4 different t values, two values were selected such that $t > \tau$ and the other two such that $t < \tau$. The bias and MSE observed are different for these four values of t in Table 8. For values of t such that t<τ; a slight positive bias is observed whereas for those values of t such that t>τ; a slight negative bias is observed. In both cases, the bias reduced as n is increased. Further, it is observed that bias and MSE increases as t increases but decreases as n increases.

Interval estimation for the MLEs and reliability function was derived by asymptotic approach, Bootstrap percentile interval and Bootstrap-t and the performance of each method is evaluated by coverage probability and expected length. The comparison of the three confidence intervals are between those intervals who attain the nominal coverage probabilities which is 0.95. These are found in Table 5, 6, 7, 9, 10 and 11, where it is observed that when n is increased, expected length derived from asymptotic method, Bootstrap percentile interval and Bootstrap-t tend to be equal, however this value is slightly higher for Bootstrap-t. Different observations are noticed for the MLEs of the parameter, for θ it is noticed that the asymptotic method gives the shortest EL especially when the sample size n is small. Furthermore, expected length shows the same pattern when fixing and changing the values of r/n and η.

It is also observed that as sample size increases the expected length for

reliability function for all values of t decreases and no major changes are observed in simulation schemes 1, 2 and 3. The width of the confidence interval for reliability function for values of t such that $t < \tau$, are shorter and approximately equal for asymptotic interval and Bootstrap percentile interval. And for those values of t such that $t > \tau$, Bootstrap percentile interval and Bootstrap-t are shorter and approximately equal.

Overall, when comparing the confidence intervals that attain or are very close to 0.95 in coverage probability, it is observed that asymptotic interval shows good results for θ , Bootstrap percentile interval displays good results for λ, θ and β and Bootstrap-t intervals don't provide satisfactory results as these intervals don't attain the nominal coverage probability value.

For reliability function, comparing the 4 different values of t for reliability function, it is observed that for asymptotic interval shows satisfactory results for t2 and moderate results for t1, Bootstrap percentile interval shows good results for t2 and t3 and Bootstrap-t shows satisfactory results for almost all values of t. Comparing the three methods for those intervals that attained the nominal coverage probability, bootstrap-t seems to give the best results for all the 4 values of time.

In addition, as explained in Section 2.5.1 a transformation was used to examine if transformation would converge faster to normal distribution and shorten the expected length and improve the coverage probability of the confidence intervals from asymptotic method. In Table 12, it was observed that the transformed reliability function denoted by R*(t) and reliability function R(t) show similar pattern for expected length when increasing n. The expected length for R(t) and R*(t) is very close for values of t such that R(t)< 0.7 i.e t_1 and t_2 and for t such that t>τ i.e t_3 and t_4 the expected length differs. From Table 13, it was also observed that the coverage

probabilities for R(t) and R*(t) are different for t_1 and t_2 and are the same for t_3 and t_4. However, when sample size n is increased, the coverage probability for R(t) and R*(t) tend to be approximately equal for t_1 and t_2. Overall, using the transformation shown in Section 2.5.1 yields better results in terms of expected length and coverage probability for reliability function for only t_1 and t_2.

Table 4. Bias and MSE results for the unknown parameters for the various simulation designs.

N=2000 and α=0.05							
n	Scheme	Bias			MSE		
		λ	θ	β	λ	θ	β
30	1	0.1126053	0.3464858	0.4465773	0.6612528	0.7693031	2.3755210
	2	0.1717799	0.2966042	0.2427189	0.7961205	0.7061647	1.9133193
	3	0.1032952	0.3173267	0.2635402	0.5906662	0.6957241	1.9010514
50	1	0.0756817	0.1738323	0.2835895	0.3526725	0.2864391	1.4092229
	2	0.1045121	0.1698394	0.1030035	0.4499678	0.2771209	1.1806660
	3	0.0648259	0.1854113	0.1967213	0.3277541	0.3042042	1.1655357
70	1	0.0654395	0.1127929	0.2154143	0.2416644	0.1671699	1.0017277
	2	0.0792746	0.1150067	0.1144503	0.2994288	0.1762348	0.8937722
	3	0.0670749	0.1115332	0.1713943	0.2380763	0.1672700	0.8135526
100	1	0.0355795	0.0805516	0.1517311	0.1433724	0.0982802	0.5933364

N=2000 and α=0.05							
n	Scheme	Bias			MSE		
		λ	θ	β	λ	θ	β
	2	0.0607651	0.0705514	0.0679098	0.1774840	0.0988394	0.5551100
	3	0.0477141	0.0721333	0.1232473	0.1559792	0.0861174	0.5566964

Table 5. Expected length and coverage probability results for unknown parameters based on asymptotic C.I

N=2000 and α=0.05							
n	Scheme	EL			CP		
		λ	θ	β	λ	θ	β
30	1	3.185142	2.671151	6.505859	0.8285	0.9700	0.9060
	2	3.432520	2.586394	5.733880	0.8415	0.9565	0.8745
	3	3.064304	2.569809	5.599392	0.8295	0.9690	0.8810
50	1	2.242863	1.735126	4.601824	0.8665	0.9645	0.9110
	2	2.386592	1.735190	4.118122	0.8490	0.9635	0.8725
	3	2.174643	1.736514	4.099120	0.8655	0.9625	0.9050
70	1	1.824896	1.371839	3.722868	0.8900	0.9620	0.9270
	2	1.879815	1.381141	3.424578	0.8885	0.9565	0.9030
	3	1.797999	1.355477	3.373849	0.8845	0.9580	0.9150
100	1	1.429466	1.104569	2.965000	0.8975	0.9610	0.9370
	2	1.494112	1.100360	2.764367	0.8995	0.9520	0.9140

N=2000 and α=0.05							
Scheme		EL			CP		
		λ	θ	β	λ	θ	β
	3	1.447970	1.086761	2.730350	0.8990	0.9630	0.9230

Table 6. Expected length and coverage probability results for unknown parameters based on Bootstrap percentile interval.

N=2000 & α=0.05							
n	Scheme	EL			CP		
		λ	θ	β	λ	θ	β
30	1	2.6210080	3.0553590	4.8730963	0.936	0.887	0.987
	2	2.8115603	3.1753263	4.3603144	0.941	0.903	0.957
	3	2.5163297	3.1756946	4.3392082	0.933	0.895	0.971
50	1	2.2383670	2.2417402	4.2929696	0.946	0.915	0.971
	2	2.3367847	2.2328083	3.7352688	0.936	0.921	0.939
	3	2.1259117	2.2366345	3.7627767	0.941	0.906	0.960
70	1	1.9404910	1.6860004	3.7568832	0.948	0.928	0.965
	2	1.9941431	1.6805977	3.3474680	0.944	0.932	0.945
	3	1.8768163	1.6674279	3.3653389	0.942	0.932	0.955
100	1	1.5873377	1.2793554	3.1748536	0.942	0.933	0.960
	2	1.6585902	1.2663017	2.8646332	0.941	0.932	0.938

N=2000 and α=0.05							
	Scheme	EL			CP		
		λ	θ	β	λ	θ	β
	3	1.5495102	1.2571000	2.8382958	0.948	0.935	0.946

Table 7. Expected length and coverage probability results for unknown parameters based on Bootstrap-t.

N=2000 and α=0.05							
n	Scheme	EL			CP		
		λ	θ	β	λ	θ	β
30	1	5.8658910	3.0553590	10.1866820	0.803	0.767	0.758
	2	6.4498720	3.1654529	9.7707379	0.815	0.792	0.783
	3	5.5923531	3.1707858	9.6046664	0.814	0.791	0.800
50	1	3.1861200	2.1034740	6.1928840	0.863	0.821	0.840
	2	3.4998366	2.1622985	5.7885149	0.865	0.824	0.846
	3	3.1001633	2.1665594	5.7192066	0.870	0.824	0.858
70	1	2.3252496	1.6272873	4.6153026	0.881	0.869	0.862
	2	2.3965523	1.6480395	4.2944606	0.888	0.866	0.877
	3	2.3020550	1.6367977	4.2740113	0.896	0.869	0.873
100	1	1.6741147	1.2587569	3.4388645	0.899	0.881	0.882

N=2000 and α=0.05						
Scheme	EL			CP		
	λ	θ	β	λ	θ	β
2	1.7497019	1.2558288	3.2118985	0.897	0.886	0.898
3	1.7077701	1.2498781	3.2033436	0.917	0.892	0.904

Table 8. Bias and MSE results for the reliability function for 4 different values of t.

N=2000 and α=0.05						
	n	Scheme	time			
			t_1=0.1587378	t_2=0.3783076	t_3=0.4980954	t_4=0.5892076
Bias	30	1	0.00963678	0.00593609	-0.00770479	-0.01988904
		2	0.00600627	0.00304072	-0.00227451	-0.00348137
		3	0.00626944	-0.00015882	-0.00791981	-0.01162270
	50	1	0.00432810	0.00137681	-0.00627578	-0.01337404
		2	0.00435129	0.00085979	-0.00163332	-0.00015445
		3	0.00459719	0.00122325	-0.00411604	-0.00773600
	70	1	0.00271893	0.00047200	-0.00487294	-0.00972188
		2	0.00294729	0.00112084	-0.00114414	-0.00137957
		3	0.00287533	0.00110607	-0.00247759	-0.00525864
	100	1	0.00214814	0.00007532	-0.00406769	-0.00817180
		2	0.00175561	0.00078443	-0.00022264	0.00034988
		3	0.00241832	0.00119803	-0.00120978	-0.00314300
MSE	30	1	0.00188774	0.00462490	0.00706499	0.01115392
		2	0.00197574	0.00500545	0.00617705	0.00690437
		3	0.00195801	0.00527073	0.00654901	0.00658110
	50	1	0.00123581	0.00296123	0.00398190	0.00560050
		2	0.00120498	0.00321731	0.00396386	0.00420968
		3	0.00132828	0.00326627	0.00389575	0.00387587
	70	1	0.00087896	0.00214284	0.00290838	0.00398082

N=2000 and α=0.05						
		Scheme	time			
			t_1=0.1587378	t_2=0.3783076	t_3=0.4980954	t_4=0.5892076
	100	2	0.00089381	0.00228636	0.00273188	0.00285963
		3	0.00091372	0.00232181	0.00278185	0.00273501
		1	0.00061468	0.00156435	0.00201584	0.00252201
		2	0.00065004	0.00159314	0.00191372	0.00202177
		3	0.00059558	0.00163265	0.00194248	0.00180617

Table 9. Expected length and coverage probability results for reliability function for 4 different values of t based on asymptotic C.I

N=2000 and α=0.05						
	n	Scheme	time			
			t_1=0.1587378	t_2=0.3783076	t_3=0.4980954	t_4=0.5892076
EL	30	1	0.1632563	0.2753633	3.7973229	3.5068506
		2	0.1661919	0.2762342	3.3337053	3.2099051
		3	0.1648510	0.2770716	3.3599970	3.2294510
	50	1	0.1320642	0.2166958	2.6342247	2.5026806
		2	0.1322582	0.2164643	2.3959728	2.3273966
		3	0.1307392	0.2161487	2.4900169	2.4027128
	70	1	0.1133680	0.1840668	2.1245281	2.0349213
		2	0.1130889	0.1836276	2.0114391	1.9549398
		3	0.1123607	0.1836857	2.0380350	1.9698301
	100	1	0.0955541	0.1543771	1.7238546	1.6675863
		2	0.0956625	0.1542394	1.6371600	1.5976408
		3	0.0948297	0.1540453	1.6660227	1.6170208
CP	30	1	0.8750	0.9400	1	0.9995
		2	0.8795	0.9290	1	0.9995
		3	0.8810	0.9280	1	1

N=2000 and α=0.05					
	Scheme	time			
		t_1=0.1587378	t_2=0.3783076	t_3=0.4980954	t_4=0.5892076
50	1	0.8950	0.9460	1	1
	2	0.9070	0.9340	1	1
	3	0.8860	0.9315	1	1
70	1	0.9145	0.9485	1	1
	2	0.9170	0.9425	1	1
	3	0.9115	0.9355	1	1
100	1	0.9315	0.9460	1	1
	2	0.9220	0.9355	1	1
	3	0.9315	0.9380	1	1

Table 10. Expected length and coverage probability results for reliability function for 4 different values of t based on bootstrap percentile interval.

N=2000 and α=0.05						
	n	Scheme	time			
			t_1=0.1587378	t_2=0.3783076	t_3=0.4980954	t_4=0.5892076
EL	30	1	0.1470809	0.2449988	0.3276528	0.4136359
		2	0.1584575	0.2727874	0.3104718	0.3309276
		3	0.1569062	0.2741428	0.3115610	0.3211346
	50	1	0.1270482	0.2058324	0.2555818	0.3153941
		2	0.1297334	0.2156067	0.2406526	0.2507377
		3	0.1284285	0.2157179	0.2399237	0.2411143
	70	1	0.1115042	0.1795383	0.2137008	0.2558771
		2	0.1121673	0.1834123	0.2032826	0.2104144
		3	0.1116223	0.1835012	0.2025403	0.2013550
	100	1	0.0950020	0.1534487	0.1770410	0.2051670
		2	0.0950962	0.1541612	0.1702578	0.1751969

N=2000 and α=0.05						
		Scheme	time			
			t_1=0.1587378	t_2=0.3783076	t_3=0.4980954	t_4=0.5892076
CP	30	3	0.0945216	0.1539914	0.1692529	0.1667562
		1	0.8810	0.9520	0.9355	0.9025
		2	0.8890	0.9395	0.9450	0.9385
		3	0.8870	0.9340	0.9360	0.9265
	50	1	0.8975	0.9510	0.9445	0.9125
		2	0.9105	0.9425	0.9380	0.9435
		3	0.8855	0.9370	0.9430	0.9350
	70	1	0.9175	0.9540	0.9455	0.9230
		2	0.9195	0.9405	0.9435	0.9485
		3	0.9130	0.9395	0.9435	0.9355
	100	1	0.9325	0.9495	0.9390	0.9265
		2	0.9255	0.9425	0.9460	0.9435
		3	0.9350	0.9380	0.9410	0.9380

Table 11. Expected length and coverage probability results for reliability function for 4 different values of t based on Bootstrap-t.

N=2000 and α=0.05						
	n	Scheme	time			
			t_1=0.1587378	t_2=0.3783076	t_3=0.4980954	t_4=0.5892076
EL	30	1	0.2589080	0.2869767	0.3518019	0.4282943
		2	0.2581866	0.3093627	0.4144693	0.4188926
		3	0.2552045	0.3122906	0.3377899	0.3166551
	50	1	0.1682810	0.2225767	0.2618972	0.3002069
		2	0.1681745	0.2307062	0.2885081	0.2900461
		3	0.1676209	0.2317799	0.2514454	0.2371723
	70	1	0.1328202	0.1885195	0.2167745	0.2406170
		2	0.1320698	0.1918778	0.2280280	0.2301147

N=2000 and α=0.05						
		Scheme	time			
			t_1=0.1587378	t_2=0.3783076	t_3=0.4980954	t_4=0.5892076
CP	100	3	0.1322407	0.1925327	0.2087426	0.1982333
		1	0.1058006	0.1581923	0.1778321	0.1933899
		2	0.1052958	0.1587719	0.1835651	0.1856874
		3	0.1051546	0.1588640	0.1725504	0.1649995
	30	1	0.9010	0.8670	0.8365	0.9465
		2	0.9395	0.9700	0.9495	0.9675
		3	0.9425	0.9650	0.9280	0.9330
	50	1	0.9425	0.9140	0.8850	0.9805
		2	0.9645	0.9680	0.9440	0.9670
		3	0.9530	0.9685	0.9265	0.9305
	70	1	0.9630	0.9335	0.9115	0.9705
		2	0.9690	0.9580	0.9470	0.9620
		3	0.9645	0.9600	0.9290	0.9300
	100	1	0.9670	0.9480	0.9285	0.9575
		2	0.9600	0.9515	0.9415	0.9560
		3	0.9700	0.9545	0.9295	0.9405

Table 12. Expected length results for R(t) and R*(t) for 4 different values of t based on Asymptotic Confidence Interval.

N=2000 and α=0.05						
	n	Scheme	time			
			t_1=0.1587378	t_2=0.3783076	t_3=0.4980954	t_4=0.5892076
R(t)	30	1	0.1632563	0.2753633	3.7973229	3.5068506
		2	0.1661919	0.2762342	3.3337053	3.2099051
		3	0.1648510	0.2770716	3.3599970	3.2294510
	50	1	0.1320642	0.2166958	2.6342247	2.5026806
		2	0.1322582	0.2164643	2.3959728	2.3273966
		3	0.1307392	0.2161487	2.4900169	2.4027128

			N=2000 and α=0.05			
		Scheme	time			
			t_1=0.1587378	t_2=0.3783076	t_3=0.4980954	t_4=0.5892076
R*(t)	70	1	0.1133680	0.1840668	2.1245281	2.0349213
		2	0.1130889	0.1836276	2.0114391	1.9549398
		3	0.1123607	0.1836857	2.0380350	1.9698301
	100	1	0.0955541	0.1543771	1.7238546	1.6675863
		2	0.0956625	0.1542394	1.6371600	1.5976408
		3	0.0948297	0.1540453	1.6660227	1.6170208
	30	1	0.1822397	0.2733138	0.9759413	0.9775335
		2	0.1844822	0.2740300	0.9733097	0.9745598
		3	0.1828486	0.2746579	0.9869290	0.9859487
	50	1	0.1394286	0.2154278	0.9638355	0.9624906
		2	0.1396005	0.2152149	0.9590539	0.9573457
		3	0.1379826	0.2149098	0.9744137	0.9702545
	70	1	0.1175455	0.1832395	0.9503871	0.9452814
		2	0.1172999	0.1828266	0.9457665	0.9402117
		3	0.1164552	0.1828832	0.9565318	0.9485673
	100	1	0.0979228	0.1538718	0.9238171	0.9140362
		2	0.0980238	0.1537431	0.9149610	0.9049882
		3	0.0971444	0.1535573	0.9255189	0.9130748

Table 13. Coverage probability results for R(t) and R*(t) for 4 different values of t based on Asymptotic Confidence Interval

			N=2000 and α=0.05			
	n	Scheme	time			
			t_1=0.1587378	t_2=0.3783076	t_3=0.4980954	t_4=0.5892076
R(t)	30	1	0.8750	0.9400	1	0.9995
		2	0.8795	0.9290	1	0.9995
		3	0.8810	0.9280	1	1
	50	1	0.8950	0.9460	1	1

N=2000 and α=0.05						
		Scheme	time			
			t_1=0.1587378	t_2=0.3783076	t_3=0.4980954	t_4=0.5892076
R*(t)	70	2	0.9070	0.9340	1	1
		3	0.8860	0.9315	1	1
		1	0.9145	0.9485	1	1
	100	2	0.9170	0.9425	1	1
		3	0.9115	0.9355	1	1
	30	1	0.9315	0.9460	1	1
		2	0.9220	0.9355	1	1
		3	0.9315	0.9380	1	1
	50	1	0.9875	0.9705	1	0.9995
		2	0.9790	0.9610	1	0.9995
		3	0.9795	0.9535	1	1
	70	1	0.9650	0.9575	1	1
		2	0.9640	0.9575	1	1
		3	0.9510	0.9560	1	1
	100	1	0.9625	0.9585	1	1
		2	0.9590	0.9485	1	1
		3	0.9515	0.9530	1	1
		1	0.9555	0.9535	1	1
		2	0.9445	0.9450	1	1
		3	0.9565	0.9495	1	1

CHAPTER 4: SIMULATED ANALYSIS

To illustrate the estimation derived in this thesis in a practical setting, a real data is required. However, a real data could not be obtained for generalized exponential data under SSPALT with progressive type-I hybrid censoring scheme and hence a simulated data is used. Two data sets of sample size 30 and 50 respectively are generated for this purpose. In the below simulated data sets, maximum likelihood estimation is used to calculate the unknown parameter values and confidence intervals is generated using asymptotic, Bootstrap percentile interval (PCI) and Bootstrap-t methods. Additionally, reliability function is calculated using four different values of t along with asymptotic interval, Bootstrap percentile interval and Bootstrap-t.

Example 1: Lifetime Simulated Data (Sample size 30)

The below data is generated using simulation with the following pre-determined parameters; $r/n = 0.5,\ \tau = 0.4567534\ and\ \eta = 0.6961452$ with initial values (1, 1.2, 2) at $\alpha = 0.05$

Table 14. Sample data of size 30 generated by simulation from generalized exponential distribution.

0.03406849	0.05063735	0.13103505	0.19087983	0.19694517
0.24263653	0.38938057	0.40181677	0.44491406	0.55505705
0.58615414	0.60367090	0.63335949	0.66275814	0.66425155
0.70856446	0.74948326	0.86391230	0.93951292	0.97961327
0.99378408	1.01798581	1.02092040	1.03786668	1.31825925
1.49570221	1.55842289	1.57455878	1.62319714	1.88963843

Even though this data was generated via simulation for the generalized exponential distribution, a statistical test is conducted to confirm the distribution of the sample.

Using R, the Kolmogorov-Smirnov Tests is conducted and below are the results

```
        One-sample Kolmogorov-Smirnov test

data:  Sample
D = 0.20829, p-value = 0.1279
alternative hypothesis: two-sided
```

Figure 4. One-sample Kolmogorov-Smirnov test for Example 1

H_0: Data is from generalized exponential distribution

H_1: Data is not from generalized exponential distribution

As seen from Figure 4, the p-value is 0.1279>0.05, hence we fail to reject H_0. At α=0.05, it is concluded that there is no sufficient evidence to reject H_0, therefore the data is considered to come from generalized exponential distribution. The below simulation is carried out using the same simulation explained in Section 3.1 and below are the results.

The estimated parameter values are $\hat{\lambda} = 0.9858772$, $\hat{\theta} = 1.221241$ and $\hat{\beta} = 1.61501$. These values are close to the initial values of the parameters, especially for $\hat{\lambda}\ and\ \hat{\theta}$. Additionally, Table 15represents the 3 types of confidence intervals for the estimators. It is noticed that the length for each interval type is different from each other. Asymptotic interval has the shortest interval length followed by Bootstrap-t and then Bootstrap percentile interval and they all contain the parameter values.

In Table 16, reliability function is estimated for four different values of t. The estimated values are close to the initial values and as expected the value of t increases

the estimated value of reliability function decreases. Addtionally Table 17 represents the confidence intervals for 4 different reliability functions based on the 4 values of t. All the confidence intervals attain the paramter values. Although, on one hand, it is observed that as t is small, t such that $t < \tau$, the confidence intervals from Asymptotic, Bootstrap percentile interval and Bootstrap-t are approximately close to each other. On the other hand, as the value of t increases, such that $t > \tau$ these values differ and Bootstrap-t and Bootstrap percentile interval show approximately close interval length followed by asymptotic. These results are in line with the simulation analysis conducted in Section 3.2.

Table 15. 95% confidence intervals for the 3 parameters.

C.I	λ	θ	β
A.I	(-0.0974521, 2.069207)	(0.3377488, 2.104733)	(-0.2561468, 3.486178)
PCI	(0.3407004, 4.463754)	(0.6555775, 3.572566)	(0.4084743, 6.000000)
Boot-t	(0.5584242, 3.595433)	(0.8726382, 3.349422)	(0.8767046, 6.122327)

Table 16. Estimated reliability function for 4 different values of T.

t	Value of t	$\hat{R}(t)$
t_1	0.1587378	0.902443
t_2	0.3783076	0.752554
t_3	0.4980954	0.661397
t_4	0.5892060	0.577863

Table 17. 95% confidence intervals for the 4 values of T

C.I Methods	t	$R(t)_L$	$R(t)_u$	C.I width
A.I	t_1	0.8148824	0.9900036	0.1751212
	t_2	0.6100124	0.8950956	0.2850832
	t_3	-0.4143124	1.7371060	2.1514184
	t_4	-0.5180922	1.6738180	2.1919102
PCI	t_1	0.8141996	0.9828775	0.1686779
	t_2	0.6161318	0.8908346	0.2747028
	t_3	0.4867290	0.8123081	0.3255791
	t_4	0.3354071	0.7343671	0.3989600
Boot-t	t_1	0.7100100	0.9695701	0.2595601
	t_2	0.5596900	0.8727651	0.3130751
	t_3	0.5197616	0.8763915	0.3566299
	t_4	0.3643291	0.7405037	0.3761746

Example 2: Lifetime simulated Data (Sample size 50)

The below data is generated using simulation with the following pre-determined parameters; $r/n = 0.5,\ \tau = 0.4567534\ and\ \eta = 0.6961452$ with initial values (1, 1.2, 2) at $\alpha = 0.05$.

Table 18. Sample data of size 50 generated by simulation from generalized exponential distribution.

0.01937873	0.10909720	0.11508837	0.13367552	0.13675059
0.15343321	0.18121483	0.25328849	0.26183436	0.28135437
0.28218964	0.35013744	0.36875291	0.39946431	0.44909080
0.47020824	0.47262729	0.48968695	0.50857753	0.52350004
0.52584269	0.54000764	0.57638812	0.59259399	0.59600734
0.59818988	0.63020078	0.63822068	0.70879673	0.79461262
0.81223634	0.87411961	1.02150266	1.05021016	1.09181785
1.09530114	1.11394134	1.15853776	1.22390723	1.25153354
1.45888091	1.48824558	1.49783096	2.01617612	2.08896864
2.10725467	2.44398666	2.56227775	2.56989766	3.17687637

```
        One-sample Kolmogorov-Smirnov test

data:  Sample
D = 0.15405, p-value = 0.1677
alternative hypothesis: two-sided
```

Figure 5. One-sample Kolmogrov-Smirnov test for Example 2

H_0: Data is from generalized exponential distribution

H_1: Data is not from generalized exponential distribution

As seen from Figure 5, the p-value is 0.1677 > 0.05, hence we fail to reject H_0. At α=0.05, it is concluded that there is no sufficient evidence to reject H_0, therefore the data is considered to come from generalized exponential distribution.

The estimated parameter values are $\hat{\lambda} = 0.8627704$, $\hat{\theta} = 1.347482$, $\hat{\beta} = 2.350746$. These values are close to the initial values of the parameters. Additionally, Table 19 represents the 3 types of confidence intervals for the estimators. It is noticed that the length for each interval type is different from each other. Asymptotic interval has the shortest interval length followed by Bootstrap percentile interval and then Bootstrap-t. However, it is observed that all the confidence intervals obtain the parameter values.

In Table 20, reliability function is estimated for four different values of t. The estimated values are close to the initial values and as expected, when the value of t increases, the estimated value of reliability function decreases. Additionally, Table 21 represents the confidence intervals for 4 different reliability functions based on the 4 values of t. All the confidence intervals contain the parameter value, however on one hand, it is observed that as t is small, such that $t < \tau$, the confidence intervals from Asymptotic, Bootstrap percentile interval and Bootstrap-t are very close to each other. On the other hand, as the value of t increases, such that $t > \tau$ these values differ, and Bootstrap-t has the shortest interval followed by Bootstrap percentile interval and then asymptotic. Similar to example 1, these results are in line with the simulation analysis conducting in Section 3.2 and it is also noticed that the confidence intervals are shorter for the parameters as well as reliability function when the sample size is 50 compared to when its 30.

Table 19. 95% confidence intervals for the 3 parameters.

C.I	λ	θ	β
A.I	(0.1155691, 1.609972)	(0.5261868, 2.168777)	(0.1831402, 4.518353)

C.I	λ	θ	β
PCI	(0.3948141, 2.082358)	(0.8516092, 2.886419)	(0.9141918, 6.000000)
Boot-t	(0.5222755, 2.313122)	(0.9732228, 2.941653)	(1.362982, 6.5581700)

Table 20. Estimated Reliability function for 4 different values of t.

t	Value of t	$\hat{R}(t)$
t_1	0.1587378	0.909572
t_2	0.3783076	0.752307
t_3	0.4980954	0.634530
t_4	0.5892060	0.509437

Table 21. 95% confidence intervals for the 4 values of t.

C.I Methods	t	$R(t)_L$	$R(t)_u$	C.I width
A.I	t_1	0.8435741	0.9755699	0.1319958
	t_2	0.6429788	0.8616352	0.2186564
	t_3	-0.8456245	2.1146840	2.9603085
	t_4	-0.8962348	1.9151090	2.8113438
PCI	t_1	0.8426583	0.9694195	0.1267612
	t_2	0.6430421	0.8586869	0.2156448
	t_3	0.4824778	0.7461056	0.2636278
	t_4	0.2943830	0.6257144	0.3313314
Boot-t	t1	0.8074727	0.9622568	0.1547841
	t_2	0.6204945	0.8501791	0.2296846
	t_3	0.5351075	0.7728837	0.2377762
	t_4	0.3824049	0.6620908	0.2796859

CHAPTER 5: SUMMARY, CONCLUSION AND FUTURE WORK

5.1 Summary and Conclusions

This thesis considers maximum likelihood estimation for the unknown parameters for the generalized exponential distribution under step stress partially accelerated life test model (SSPALT) with progressive type I hybrid censoring scheme. MLEs for unknown parameters are derived using numeric methods such as Nelder–Mead, quasi-Newton and conjugate-gradient algorithms which is used in Optim function in R program. The variances of the MLEs are derived using the Fisher information matrix which is used to construct the asymptotic confidence intervals for the unknown parameters. Further, reliability function is estimated using the invariance property of the MLEs.

The performance of the point estimators is evaluated by bias and MSE and coverage probability and expected length is used to evaluate the performance of the interval estimators. These are further investigated using simulation with various designs and the result of the analysis is provided in detail in Section 3.2. A real data suitable for this thesis was not available and hence two simulated data were used to illustrate the MLEs derived in the thesis.

From the simulation study, it was found that the MSE and Bias values are satisfactory for the scale and shape parameters, however for the acceleration factor it was satisfactory as the value of n was increased. In general, the performance of the MLEs is better as sample size increases and for confidence interval when comparing those intervals that attain the nominal coverage probability, it is observed that Bootstrap percentile interval displays the best results for λ, θ and β and Bootstrap-t intervals has the poorest results, as it doesn't attain the nominal coverage probability value.

Further, reliability function for four different values of t were examined. The estimated values for the reliability functions are close to the initial values and hence satisfactory. Comparing the three methods for obtaining confidence interval and evaluating them based on expected length and coverage probability for those intervals attaining nominal coverage probability, it is concluded that bootstrap-t provides the best results for reliability function all the 4 values of time. And for the transformed reliability functions, it was found that the transformed reliability function shows better results in terms of coverage probability and expected length for only t_1 and t_2 for the asymptotic confidence interval.

5.2 Suggestions for Further Research

For further research, another estimation method such as Bayesian inference in case of SSPLAT under the same censoring scheme can be considered and compared to the results obtained in this thesis. Additionally, it was also observed from this study that the MLEs are close to the initial values when the sample size is large, hence another estimation method can be explored for small sample size. They can also apply the transformed reliability function for bootstrap percentile and bootstrap-t intervals and investigate the results. Furthermore, as a future work the considered methods in this thesis can also be used to estimate other symmetric or non-symmetric distributions.

REFERENCES

Aldrich, J. (1997) R. A. Fisher and the making of maximum likelihood 1912-1922. *Statistical Science*, 12, 162–19.

Alizadeh, M., Rezaei, S., Bagheri, S. and Nadarajah, S. (2015) Efficient estimation for the generalized exponential distribution, 56:1015–1031, DOI 10.1007/s00362-014-0621-7

Balakrishnan, N., Zhang, L. and Xie, Q. (2009) Inference for a Simple Step- Stress Model with Type-I Censoring and Lognormally Distributed Lifetimes, Communications in Statistics—Theory and Methods, 38:10, 1690-1709, DOI: 10.1080/03610920902866966.

Casella, G., & Berger, R. L. (2002). *Statistical Inference*. Duxbury Advanced Series.

Childs, A., Chandrasekar, B., Balakrishnan, N. and Kundu, D. (2002), Exact likelihood inference based on type-I and type-II hybrid censoring samples from the exponential distribution, Vol. 55, No. 2, 319-330 (2003) Ann. Inst. Statist. Math, The Institute of Statistical Mathematics.

Danish, M.Y. and Aslam, M. (2013) Bayesian estimation for randomly censored generalized exponential distribution under asymmetric loss functions, Journal of Applied Statistics, 40:5, 1106-1119, DOI: 10.1080/02664763.2013.780159

Efron, B. and Tibshirani, R. (1986). Bootstrap Methods for Standard Errors, Confidence Intervals, and Other Measures of Statistical Accuracy. Statistical Science, 54-75

Efron, B. and Tibshirani, R. (1993). An Introduction to the Bootstrap, London: Chapman & Hall.

El-Din, M., Amein M. M., Shafay, A.R. and Mohamed, S. (2017) Estimation of generalized exponential distribution based on an adaptive progressively type-II censored sample, Journal of Statistical Computation and Simulation, 87:7, 1292-1304, DOI: 10.1080/00949655.2016.1261863

Falconnet, M., Loukianova, D. and Matias, C. (2014) Asymptotic normality and efficiency of the maximum likelihood estimator for the parameter of a ballistic random walk in a random environment. Mathematical Methods of Statistics, Allerton Press, Springer (link), 23 (1), pp.1-19. 10.3103/S1066530714010013 .hal-00783980v2

Gu, B.-Q and Yue, R,-X (2013) Some Notes on Parameter Estimation for Generalized Exponential Distribution, Communications in Statistics - Theory and Methods, 42:10, 1787-1805, DOI: 10.1080/03610926.2011.597918

Gugushvili, S. (2014, October 20). Maximum Likelihood Estimation: Multidimensional Case. Leiden, Netherlands.

Gupta, R.D. and Kundu, D. (1997). Exponentiated exponential family: an alternative to gamma and Weibull distribution. Technical report. Dept of Math., Stat. & Comp. Sci., University of New Brusnwick, Saint John, NB, Canada.

Gupta, R.D. and Kundu, D. (1999), Theory & Methods: Generalized exponential distributions. Australian & New Zealand Journal of Statistics, 41: 173-188. doi:10.1111/1467-842X.00072

Gupta, R.D. and Kundu, D. (2001) Generalized exponential distribution: different method of estimations, Journal of Statistical Computation and Simulation, 69:4, 315-337, DOI: 10.1080/00949650108812098

Ismail, A.A (2012) Inference in the generalized exponential distribution under partially accelerated tests with progressive Type-II censoring, Elsevier, Theoretical and Applied Fracture Mechanics 59 (2012) 49–56

Ismail, A.A (2014) Likelihood inference for a step-stress partially accelerated life test model with Type-I progressively hybrid censored data from Weibull distribution, Journal of Statistical Computation and Simulation, 84:11, 2486-2494, DOI: 10.1080/00949655.2013.836195

Jaheen, Z. F. , Moustafa, H. M. and Abd El-Monem, G. H. (2014) Bayes Inference in Constant Partially Accelerated Life Tests for the Generalized Exponential Distribution with Progressive Censoring, Communications in Statistics - Theory and Methods, 43:14, 2973-2988, DOI: 10.1080/03610926.2012.687068

Kundu, D. & Pradhan, B. (2009) Estimating the Parameters of the Generalized Exponential Distribution in Presence of Hybrid Censoring, Communications in Statistics—Theory and Methods, 38:12, 2030-2041, DOI: 10.1080/03610920802192505

Lawless, J.F. (1982). Statistical Models and Methods for Lifetime Data. New York: Wiley.

Lehmann, E. (1998). Elements of Large-Sample Theory (Springer Texts in Statistics) Corrected Edition. Springer.

Madi, M.T., and Raqab, M.Z. (2009) Bayesian Inference for the Generalized Exponential Distribution Based on Progressively Censored Data, Communications in Statistics—Theory and Methods, 38:12, 2016-2029, DOI: 10.1080/03610920902855951

Mitra, S. and Kundu, D (2008) Analysis of left censored data from the generalized exponential distribution, Journal of Statistical Computation and Simulation, 78:7, 669-679, DOI: 10.1080/00949650701344158

Raqab, M.Z. and Madi, M.T. (2005) Bayesian inference for the generalized exponential distribution, Journal of Statistical Computation and Simulation, 75:10, 841-852, DOI: 10.1080/00949650412331299166

Shafay, A. R. (2016) Exact Inference for a Simple Step-stress Model with Generalized Type-I Hybrid Censored Data from the Exponential Distribution, Communications in Statistics - Simulation and Computation, 45:1, 181-206, DOI: 10.1080/03610918.2013.858165

Soliman, A.A.(2002) Reliability estimation in a generalized life-model with application to the burr-XII. IEEE Trans Reliab. 2002;51(3):337–343.

Sun, T. and Shi,Y. (2016) Estimation for Birnbaum–Saunders Distribution in Simple Step Stress–accelerated Life Test with Type-II Censoring, Communications in Statistics - Simulation and Computation, 45:3, 880-901, DOI: 10.1080/03610918.2013.

Wang, R. , Xu, X. , Pan, R. and Sha, N. (2012) On Parameter Inference for Step-Stress Accelerated Life Test with Geometric Distribution, Communications in

Statistics - Theory and Methods, 41:10, 1796-1812, DOI: 10.1080/03610926.2010.551455

Wu, S,-J. (2002) Parameter estimation of the two-parameter exponential distribution under step-stress accelerated test with censoring, Journal of Information and Optimization Sciences, 23:2, 355-365, DOI: 10.1080/02522667.2002.10699533

Xu, H.-Y. and Fei, H. (2012) Models Comparison for Step-Stress Accelerated Life Testing, Communications in Statistics - Theory and Methods, 41:21, 3878-3887, DOI: 10.1080/03610926.2012.701699

Zarrin, S. , Shashi, S., and Prof. Arif-ul-lslam (2010) Step-stress Accelerated Life Testing for Exponentiated Weibull Distribution, Safety and Reliability, 30:3, 5-13, DOI: 10.1080/09617353.2010.11690911

Zheng, G. and Shi, Y. (2015) Statistical Analysis in Constant-stress Accelerated Life Tests for Generalized Exponential Distribution with Progressive Type-I Hybrid Censoring, Communications in Statistics - Theory and Methods, 44:23, 4962-4982, DOI: 10.1080/03610926.2013.841920

APPENDIX

APPENDIX A: R CODE FOR SIMULATION STUDY

```
#nr=20;n=100;Bnr=100;r=40; tau=0.3; eta=3; lmd=1; tht=1.2; beta =2;
m=0;alpha=0.05;t=0.7 #Original values
#Simulation Design 1
#1
#set.seed(101)
#nr=2000; Bnr=1000; n=30; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.1587378
#nr=2000; Bnr=1000; n=30; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.3783076
#nr=2000; Bnr=1000; n=30; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.4980954
#nr=2000; Bnr=1000; n=30; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.5892060
#2
#set.seed(102)
#nr=2000; Bnr=1000; n=30; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.1587378
#nr=2000; Bnr=1000; n=30; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.3783076
#nr=2000; Bnr=1000; n=30; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.4980954
#nr=2000; Bnr=1000; n=30; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.5892076
#3
#set.seed(103)
#nr=2000; Bnr=1000; n=30; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.1587378
#nr=2000; Bnr=1000; n=30; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.3783076
#nr=2000; Bnr=1000; n=30; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.4980954
#nr=2000; Bnr=1000; n=30; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.5892076
#4
#set.seed(201)
```

#nr=2000; Bnr=1000; n=50; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.1587378

#nr=2000; Bnr=1000; n=50; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.3783076

#nr=2000; Bnr=1000; n=50; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.4980954

#nr=2000; Bnr=1000; n=50; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.5892076

#5

#set.seed(202)

#nr=2000; Bnr=1000; n=50; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.1587378

#nr=2000; Bnr=1000; n=50; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.3783076

#nr=2000; Bnr=1000; n=50; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.4980954

#nr=2000; Bnr=1000; n=50; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.5892076

#6

#set.seed(203)

#nr=2000; Bnr=1000; n=50; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.1587378

#nr=2000; Bnr=1000; n=50; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.3783076

#nr=2000; Bnr=1000; n=50; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.4980954

#nr=2000; Bnr=1000; n=50; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.5892076

#7

#set.seed(301)

#nr=2000; Bnr=1000; n=70; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.1587378

#nr=2000; Bnr=1000; n=70; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.3783076

#nr=2000; Bnr=1000; n=70; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.4980954

#nr=2000; Bnr=1000; n=70; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.5892076

#8

#set.seed(302)

#nr=2000; Bnr=1000; n=70; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.1587378

#nr=2000; Bnr=1000; n=70; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.3783076

#nr=2000; Bnr=1000; n=70; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.4980954

#nr=2000; Bnr=1000; n=70; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.5892076

#9

#set.seed(303)

#nr=2000; Bnr=1000; n=70; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.1587378

#nr=2000; Bnr=1000; n=70; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.3783076

#nr=2000; Bnr=1000; n=70; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.4980954

#nr=2000; Bnr=1000; n=70; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.5892076

#10

#set.seed(401)

#nr=2000; Bnr=1000; n=100; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.1587378

#nr=2000; Bnr=1000; n=100; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.3783076

#nr=2000; Bnr=1000; n=100; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.4980954

#nr=2000; Bnr=1000; n=100; r=0.5*n; tau=0.4567534; eta=0.6961452; t=0.5892076

#11

#set.seed(402)

#nr=2000; Bnr=1000; n=100; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.1587378

#nr=2000; Bnr=1000; n=100; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.3783076

#nr=2000; Bnr=1000; n=100; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.4980954

#nr=2000; Bnr=1000; n=100; r=0.7*n; tau=0.4567534; eta=0.6961452; t=0.5892076

#12

#set.seed(403)

#nr=2000; Bnr=1000; n=100; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.1587378

#nr=2000; Bnr=1000; n=100; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.3783076

```
#nr=2000; Bnr=1000; n=100; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.4980954
#nr=2000; Bnr=1000; n=100; r=0.7*n; tau=0.4567534; eta=1.466511; t=0.5892076
lmd=1; tht=1.2; beta=2; m=0; alpha=0.05
# Creating Empty Vectors/Matrix
#Parameter
P.All=matrix(NA,nrow=nr,ncol=3)  #to save all parameter estimators before Bootstrap
colnames(P.All)=c("Est.lambda","Est.Theta","Est.Beta")
Var.P.All=matrix(NA,nrow=nr,ncol=3)
colnames(Var.P.All)=c("Var.lambda","Var.Theta","Var.Beta")
C.I.P.All=matrix(NA,ncol=6,nrow=nr)
colnames(C.I.P.All)=c("lower.Lambda","Upper.Lambda","Lower.Theta","Upper.Thet
a","Lower.Beta","Upper.Beta")
#Reliability Function Main loop
RF.All=c()
Var.RF.All=c()
d.l1=c()
d.t1=c()
d.l2=c()
d.t2=c()
d.b2=c()
#Expected length
Ex.Lambda=c()
Ex.Theta=c()
Ex.Beta=c()
Ex.R.F=c()
```

```
Ex.T.R.F=c()
#Coverage probability
C.Prob.L= c()
C.Prob.T=c()
C.Prob.B=c()
C.Prob.RF=c()
C.Prob.T.RF=c()
#Bootstrap
B=matrix(NA,nrow=Bnr,ncol=3)
colnames(B)=c("B.lambda","B.Theta","B.Beta")
Bz=c()
BR.F=c()
Bcov=matrix(NA,nrow=Bnr,ncol=3)
colnames(Bcov)=c("BVar.lambda","BVar.Theta","BVar.Beta")
#Bootstrap-t
Boot.t=matrix(NA,nrow=nr,ncol=8)   # for Boot T for parameters
colnames(Boot.t)=c("lower.BTLambda","Upper.BTLambda","Lower.BTTheta","Upp
er.BTTheta","Lower.BTBeta","Upper.BTBeta","LowerR.F","Upper.R.F")
Zstar=matrix(NA,ncol=3,nrow=Bnr)   #to save values of parameters Z* Bootstrap t
#Bootstrap Reliability
BRF=c()
BVarR.F=matrix(NA,nrow=nr,ncol=1)  #to save values of variance of R.F
Bd.l1=c()
Bd.t1=c()
Bd.l2=c()
```

```
Bd.t2=c()
Bd.b2=c()
Boot.t.RF.L=c()                #R.F lower bound Boot-t
Boot.t.RF.U=c()                #R.F upper bound Boot-t
ZRFstar=c()
#Bootstrap-p
Boot.P=matrix(NA,nrow=nr,ncol=8)
colnames(Boot.P)=c("lower.BLambda","Upper.BLambda","Lower.BTheta","Upper.B
Theta","Lower.BBeta","Upper.BBeta","LowerR.F","Upper.R.F")
#Boothstrap Expected length
P.E.L.L=c()
P.E.L.T=c()
P.E.L.B=c()
P.E.L.R=c()
T.E.L.L=c()
T.E.L.T=c()
T.E.L.B=c()
T.E.L.R=c()
#Start of Main loop
for (jj in 1:nr){
  U=runif(n)
  US=sort(U) #Sort the uniform random numbers
  F1= function(x) (1-exp(-x/lmd))^tht  # function for 1-exp((-tau)/lmd))^tht
  F2=function(x) (1-exp(-(tau+beta*(x-tau))/lmd))^tht
```

```
  U1=US[US<=F1(tau)] #U1 consists of the uniform random numbers that are <= F(tau)
  U2=US[US>F1(tau)] #U2 consists of the uniform random numbers that are  >F(tau) and are <= F(eta)
  x1=c()
  xx1=c()
  x2=c()
  y=c()
  xx1 = -lmd * log(1-U1^(1/tht))
  nnu=length(xx1)
  nu=min(nnu,r)
  x1=xx1[xx1<=xx1[nu]]
  y= tau+(-lmd*log(1-U2^(1/tht))- tau)/beta  #Use U2 to generate observations from f_2(x), find their number (n_a)
  # print(y)
  if (nu<r){x2=y[y<=min(y[r-nu],eta)]} else {print ("Number exceeding or equal to pre-determined failures")}
  x=c(x1,x2)
  nu=length(x1)
  na=length(x2)
  p=c(1,1.2,2)
  if (na>0 & nu>0){
  ## a[1]= Lambda, a[2]=Theta, a[3]=Beta
    L1=function(a){-(
      nu*log(a[2]/a[1])
      - sum(x1)/a[1]
```

```
    +(a[2]-1)*sum(log(1-exp(-x1/a[1])))
    +(r-nu)*log(a[3]*a[2]/a[1])
    -(1/a[1])*sum(tau+a[3]*(x2-tau))
    +(a[2]-1)*sum(log(1-exp((-1/a[1])*(tau+a[3]*(x2-tau)))))
    +(n-r-m)*log(1-(1-exp((-1/a[1])*(tau+a[3]*(x[r]-tau))))^a[2]))}
  L2=function(a){-(
    nu*log(a[2]/a[1])
    - sum(x1)/a[1]
    + (a[2]-1)*sum(log(1-exp(-x1/a[1])))
    +na*log(a[3]*a[2]/a[1])
    -(1/a[1])*sum(tau+a[3]*(x2-tau))
    +(a[2]-1)*sum(log(1-exp((-1/a[1])*(tau+a[3]*(x2-tau)))))
    +(n-(nu+na)-m)*log(1-(1-exp((-1/a[1])*(tau+a[3]*(eta-tau))))^a[2]))}
  try({if (nu+na==r){
    res=optim(p,L1,method="L-BFGS-B",lower=c(0.1,0.1,0.3),upper =
c(5,5,6),hessian = TRUE)
    z=res$par
    CovMatrix=solve(res$hessian)
  } else
  {
    res=optim(p,L2,method="L-BFGS-B",lower=c(0.1,0.1,0.3),upper =
c(5,5,6),hessian = TRUE)
    z=res$par
    CovMatrix=solve(res$hessian)
  }},silent=T)
```

```
} #end of if
P.All[jj,1]=z[1]   #lambda
P.All[jj,2]=z[2]    #theta
P.All[jj,3]=z[3]     #beta
Var.P.All[jj,1]= CovMatrix[1,1]
Var.P.All[jj,2]= CovMatrix[2,2]
Var.P.All[jj,3]= CovMatrix[3,3]
C.I.P.All[jj,1]= z[1] - qnorm (1 - (alpha/2)) *sqrt(CovMatrix[1,1])
C.I.P.All[jj,2]= z[1] + qnorm (1 - (alpha/2)) *sqrt(CovMatrix[1,1])
C.I.P.All[jj,3]= z[2] - qnorm (1 - (alpha/2)) *sqrt(CovMatrix[2,2])
C.I.P.All[jj,4]= z[2] + qnorm (1 - (alpha/2)) *sqrt(CovMatrix[2,2])
C.I.P.All[jj,5]= z[3] - qnorm (1 - (alpha/2)) *sqrt(CovMatrix[3,3])
C.I.P.All[jj,6]= z[3] + qnorm (1 - (alpha/2)) *sqrt(CovMatrix[3,3])
# True parameter R.F
p=c(1,1.2,2)
if (t < tau) {
  P.R.F=1-(1-exp(-t/p[1]))^p[2]
} else if
(t < eta)
{P.R.F= 1-(1-exp((-1/p[1])*(tau+(p[3]*(t-tau)))))^p[2]
} else if
(t>eta)
{ #print("Invalid:t greater than eta")
  P.R.F="Invalid:t greater than eta"
}
```

```
 # Reliability Function (Main loop)
 if (t < tau) {
   RF=round(1-(1-exp(-t/z[1]))^z[2],6)
 } else if
 (t < eta)
 { RF= round(1-(1-exp((-1/z[1])*(tau+(z[3]*(t-tau)))))^z[2],6)
 } else if
 (t > eta)
   RF="Invalid:t is greater than eta"
RF.All[jj]=RF
c4=RF.All-P.R.F
#Variance of R.F (Main loop)
 if (t<tau){
   d.l1= (z[2]*t*(1-exp(-t/z[1]))^z[2])/(((z[1])^2)*(exp(t/z[1]) -1)) # R.F1 Bootstrap derv of lambada
   d.t1=-((1-exp(-t/z[1]))^z[2]) * (log(1-exp(-t/z[1]))) # R.F1 Bootstrap derv of theta
   gradient.T=matrix(c(d.l1,d.t1),ncol=2,byrow=F)
   Var.RF=(gradient.T)%*% CovMatrix[1:2,1:2] %*% t(gradient.T)
 } else if (t<eta){
   d.l2= (1/(z[1])^2)*(z[2]*(z[3]*((t-tau)+tau)))*((1-exp((-1/z[1])*(z[3]*(t-tau)+tau)))^(z[2]-1))*(exp((-1/z[1])*z[3]*(t-tau)+tau)) #derivative of Reliability function 2 from main loop with respect to lambda
   d.t2=-1*((1-exp((-1/z[1])*((z[3]*(t-tau))+tau)))^(z[2]))*log(1-exp((-1/z[1])*((z[3]*(t-tau))+tau))) ##derivative of Reliability function 2 from main loop with respect to theta
```

```
    d.b2=-1/(z[1]*(exp((1/z[1])*((t-tau)*z[3])+tau)-1))*(z[2]*(t-tau))*(1-exp((-
1/z[1])*((z[3]*(t-tau))+tau)))^(z[2]) ##derivative of Reliability 2 function  from main
loop with respect to beta
    gradient.T=matrix(c(d.l2,d.t2,d.b2),ncol=3,byrow=F)
    Var.RF=(gradient.T)%*% CovMatrix %*% t(gradient.T)
  }else if (t>eta){
    c="Invalid:t is greater than eta"
  }
Var.RF.All[jj]=Var.RF
#Start of Bootstrap
  for (i in 1:Bnr) {
    BU=runif(n)
    BUS=sort(BU) #Sort the uniform random numbers
    BF1= function(x) (1-exp(-x/z[1]))^z[2]  # function for 1-exp((-tau)/lmd))^tht
    BF2=function(x) (1-exp(-(tau+z[3]*(x-tau))/z[1]))^z[2]
    BU1=BUS[BUS<=BF1(tau)] #U1 consists of the uniform random numbers that are
<= F(tau)
    BU2=BUS[BUS>BF1(tau)] #U2 consists of the uniform random numbers that are
>F(tau) and are <= F(eta)
    Bx1=c()
    Bxx1=c()
    Bx2=c()
    By=c()
    Bxx1 = -z[1] * log(1-BU1^(1/z[2]))
    Bnnu=length(Bxx1)
```

```
   Bnu=min(Bnnu,r)
   Bx1=Bxx1[Bxx1<=Bxx1[nu]]
   Bx1 = -z[1] * log(1-BU1^(1/z[2]))   #
   Bnu=length(Bx1)
   By= tau+(-z[1]*log(1-BU2^(1/z[2]))- tau)/z[3]  #Use BU2 to generate observations from f_2 (x), find their number (n_a)
   # print(By)
   if (Bnu<r){Bx2=By[By<=min(By[r-Bnu],eta)]} else {print ("Number exceeding or equal to pre-determined failures")}
   #print(Bx1)
   #print(Bx2)
   Bx=c(Bx1,Bx2)
   Bna=length(Bx2)
   By= tau+(-z[1]*log(1-BU2^(1/z[2]))- tau)/z[3] #Use U2 to generate observations from f_2 (x), find their number (n_a)
   # print(By)
   if (Bnu<r){Bx2=By[By<=min(By[r-Bnu],eta)]} else {print ("Number exceeding or equal to pre-determined failures")}
   Bx=c(Bx1,Bx2)
   Bnu=length(Bx1)
   Bna=length(Bx2)
   if (Bna>0 & Bnu>0){
     ## a[1]= Lambda, a[2]=Theta, a[3]=Beta
     BL1=function(a){-(
       Bnu*log(a[2]/a[1])
```

```
  - sum(Bx1)/a[1]
  +(a[2]-1)*sum(log(1-exp(-Bx1/a[1])))
  +(r-Bnu)*log(a[3]*a[2]/a[1])
  -(1/a[1])*sum(tau+a[3]*(Bx2-tau))
  +(a[2]-1)*sum(log(1-exp((-1/a[1])*(tau+a[3]*(Bx2-tau)))))
  +(n-r-m)*log(1-(1-exp((-1/a[1])*(tau+a[3]*(Bx[r]-tau))))^a[2]))}
BL2=function(a){-(
  Bnu*log(a[2]/a[1])
  - sum(Bx1)/a[1]
  + (a[2]-1)*sum(log(1-exp(-Bx1/a[1])))
  +Bna*log(a[3]*a[2]/a[1])
  -(1/a[1])*sum(tau+a[3]*(Bx2-tau))
  +(a[2]-1)*sum(log(1-exp((-1/a[1])*(tau+a[3]*(Bx2-tau)))))
  +(n-(Bnu+Bna)-m)*log(1-(1-exp((-1/a[1])*(tau+a[3]*(eta-tau))))^a[2]))}
try({if (Bnu+Bna==r){
  Bres=optim(p,BL1,method="L-BFGS-B",lower=c(0.1,0.1,0.3),upper =
c(5,5,6),hessian = TRUE)
  Bz=Bres$par
  BCovMatrix=solve(Bres$hessian)
} else
{
  Bres=optim(p,BL2,method="L-BFGS-B",lower=c(0.1,0.1,0.3),upper =
c(5,5,6),hessian = TRUE)
  Bz=Bres$par
  BCovMatrix=solve(Bres$hessian)
```

```
  }},silent=T)
 }
 B[i,1]=Bz[1]
 B[i,2]=Bz[2]
 B[i,3]=Bz[3]
 Bcov[i,1]=BCovMatrix[1,1]
 Bcov[i,2]=BCovMatrix[2,2]
 Bcov[i,3]=BCovMatrix[3,3]
 # Reliability function Bootstrap
 if (t<tau){
  Bd.l1= (Bz[2]*t*(1-exp(-t/Bz[1]))^Bz[2])/(((Bz[1])^2)*(exp(t/Bz[1]) -1))   # R.F1
Bootstrap derv of lambada
  Bd.t1=-((1-exp(-t/Bz[1]))^Bz[2]) * (log(1-exp(-t/Bz[1])))  # R.F1 Bootstrap derv of
theta
  gradient=matrix(c(Bd.l1,Bd.t1),ncol=2,byrow=F)
  BVar.R.F=round((gradient)%*% BCovMatrix[1:2,1:2] %*% t(gradient),6)
 } else if (t<eta){
  Bd.l2= (1/(Bz[1])^2)*(Bz[2]*(Bz[3]*((t-tau)+tau)))*((1-exp((-1/Bz[1])*(Bz[3]*(t-
tau)+tau)))^(Bz[2]-1))*(exp((-1/Bz[1])*Bz[3]*(t-tau)+tau)) #derivative of  Bootstrap
R.F2 with respect to lambda
  Bd.t2=-1*((1-exp((-1/Bz[1])*((Bz[3]*(t-tau))+tau)))^(Bz[2]))*log(1-exp((-
1/Bz[1])*((Bz[3]*(t-tau))+tau))) ##derivative of  Bootstrap R.F2 with respect to theta
  Bd.b2=-1/(Bz[1]*(exp((1/Bz[1])*((t-tau)*Bz[3])+tau)-1))*(Bz[2]*(t-tau))*(1-
exp((-1/Bz[1])*((Bz[3]*(t-tau))+tau)))^(Bz[2]) ##derivative of  Bootstrap R.F2 with
respect to beta
```

```
  gradient=matrix(c(Bd.l2,Bd.t2,Bd.b2),ncol=3,byrow=F)
  BVar.R.F= (gradient)%*% BCovMatrix %*% t(gradient)
 } else if (t>eta){
  BVar.R.F="Invalid:t is greater than eta"
 }
 if (t < tau) {
  BRF=1-(1-exp(-t/Bz[1]))^Bz[2]
 } else if
 (t < eta)
 {  BRF=  1-(1-exp((-1/Bz[1])*(tau+(Bz[3]*(t-tau)))))^Bz[2]  #using  invariance
property of MLE for accelerated condition
 } else if
 (t>eta)
  BRF="Invalid:t is greater than eta"
 BR.F[i]=BRF  #for Bootstrap
 # Boot-t
 if (BCovMatrix[3,3]=="NaN") cat(jj,i,BCovMatrix[3,3],"\n")
 Zstar[i,1]= (Bz[1]-z[1])/sqrt(BCovMatrix[1,1])  # Z star for each Boot
 Zstar[i,2]= (B[2]-z[2])/sqrt(BCovMatrix[2,2])
 Zstar[i,3]= (Bz[3]-z[3])/sqrt(BCovMatrix[3,3])
 #cat("yes",BRF,RF,sqrt(BVar.R.F),"\n")
 ZRFstar[i]=(BRF - RF)/sqrt(as.numeric(BVar.R.F))
} #end of Boot loop
```

```
 # Boot-t
  zRF= quantile(sort(ZRFstar),probs=c(1-(alpha/2),alpha/2),na.rm=F)
 # Boot-t Confidence interval
  Boot.t.RF.L[jj]=RF-(zRF[1]*sqrt(as.numeric(Var.RF)))
  Boot.t.RF.U[jj]=RF-(zRF[2]*sqrt(as.numeric(Var.RF)))
  zL=quantile(sort(Zstar[,1]),probs=c(1-(alpha/2),alpha/2))
  zT=quantile(sort(Zstar[,2]),probs=c(1-(alpha/2),alpha/2))
  zB=quantile(sort(Zstar[,3]),probs=c(1-(alpha/2),alpha/2))
  Boot.t[jj,1]=z[1]-(zL[1]*sqrt(CovMatrix[1,1]))
  Boot.t[jj,2]=z[1]-(zL[2]*sqrt(CovMatrix[1,1]))
  Boot.t[jj,3]=z[2]-(zL[1]*sqrt(CovMatrix[2,2]))
  Boot.t[jj,4]=z[2]-(zL[2]*sqrt(CovMatrix[2,2]))
  Boot.t[jj,5]=z[3]-(zL[1]*sqrt(CovMatrix[3,3]))
  Boot.t[jj,6]=z[3]-(zL[2]*sqrt(CovMatrix[3,3]))
  Boot.t[jj,7]= Boot.t.RF.L[jj]
  Boot.t[jj,8]= Boot.t.RF.U[jj]
  # Boot-p
  low=round((alpha/2)*Bnr,0)
  Up=round((1-(alpha/2))*Bnr,0)
  SortB=apply(B,2,sort)              #Parameter Estimate
  SortBR.F=apply(matrix(BR.F,ncol=1,nrow=Bnr),2,sort)       #Reliability Functions
  Pa.Boot=
matrix(c(SortB[low,1],SortB[low,2],SortB[low,3],SortBR.F[low,1],SortB[Up,1],Sort
B[Up,2],SortB[Up,3],SortBR.F[Up,1]),ncol=2,nrow=4,byrow=FALSE)              #C.I
Bootstrap-Percentile
```

```
  colnames(Pa.Boot)=c("Lower Bound", "Upper Bound")
  row.names(Pa.Boot)=c("Lambda","Theta","Beta","Reliability")
  Boot.P[jj,1]=Pa.Boot[1,1]
  Boot.P[jj,2]=Pa.Boot[1,2]
  Boot.P[jj,3]=Pa.Boot[2,1]
  Boot.P[jj,4]=Pa.Boot[2,2]
  Boot.P[jj,5]=Pa.Boot[3,1]
  Boot.P[jj,6]=Pa.Boot[3,2]
  Boot.P[jj,7]=Pa.Boot[4,1]
  Boot.P[jj,8]=Pa.Boot[4,2]
} #End of  main loop
# Log Transformation of Reliability Function (From main loop)
R.Fstar=log(RF.All/(1-RF.All))
d.R.Fstar= -(1/((RF.All-1)*RF.All))  #derivative for log transformation of R.F
Var.Rstar=(d.R.Fstar^2)*Var.RF.All  #variance for log transformation of R.F
# Asymptotic Confidence Intervals
# Transformed R.F (Log R.F)
l.R.Flog= R.Fstar - qnorm (1 - (alpha/2)) *sqrt(Var.Rstar) #Lower bound
u.R.Flog= R.Fstar + qnorm (1 - (alpha/2)) *sqrt(Var.Rstar) #upper bound
# RF** (Transformed back to R.F)
T.l.R.F=exp(l.R.Flog)/(1+exp(l.R.Flog))  #Lower bound for C.I after transforming back to R.F
T.u.R.F=exp(u.R.Flog)/(1+exp(u.R.Flog))  #upper bound for C.I after transforming back to R.F
```

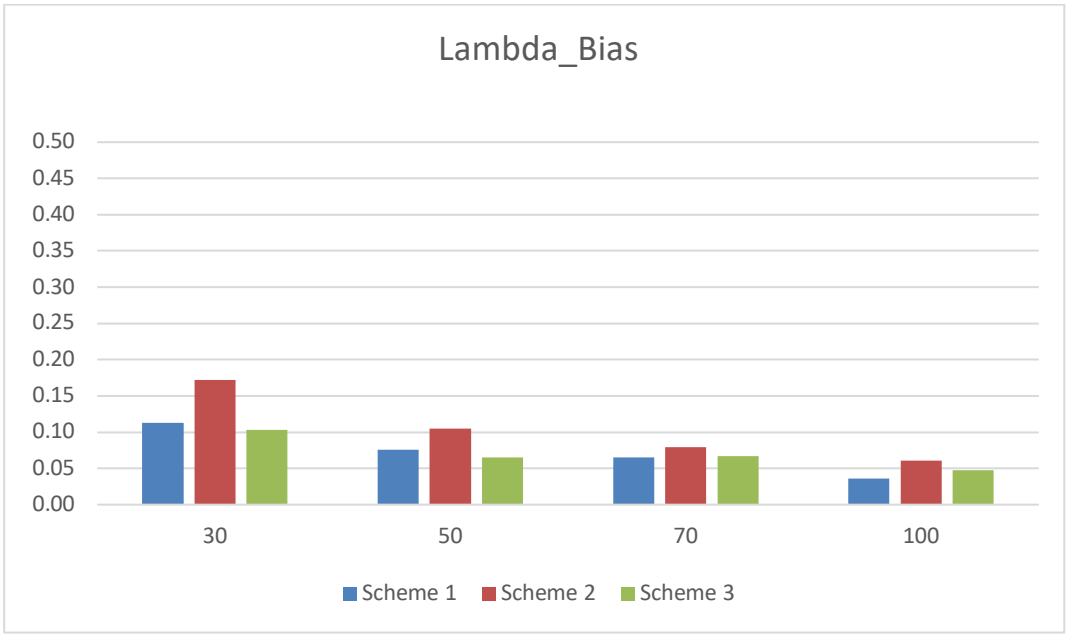
```
# Confidence Width of R.F**
Tciw=T.u.R.F-T.l.R.F
#Confidence Interval of Original R.F (without Transformation)
l.R.F=RF.All - qnorm (1 - (alpha/2)) *sqrt(Var.RF.All)    #lower bound of C.I
u.R.F=RF.All + qnorm (1 - (alpha/2)) *sqrt(Var.RF.All)   # upper bound of C.I
# Confidence Width of R.F
ciw=u.R.F-l.R.F
C.I.RF=matrix(c(l.R.F,u.R.F,T.l.R.F,T.u.R.F),ncol=4,nrow=nr)
colnames(C.I.RF)=c("Lower.RF","Upper.RF","Lower.RF**","Upper.RF**")
 # Bias (Main loop)
 c1=P.All[,1]-p[1]
 c2=P.All[,2]-p[2]
 c3=P.All[,3]-p[3]
 C=matrix(c(c1,c2,c3),ncol=3,byrow=F)
 Bias=(apply(C,2,sum))/nr
 c4=RF.All-P.R.F
 BiasRF=sum(c4)/nr
 MSERF=sum(c4^2)/nr
 #MSE (Main loop)
 C2=matrix(c(c1^2,c2^2,c3^2),ncol=3,byrow=F)
 MSE=(apply(C2,2,sum))/nr
 #Coverage probability
 C.P.L=sum(as.numeric ((C.I.P.All[,2]>p[1]) & (p[1]>C.I.P.All[,1])))/nr
 C.P.T=sum(as.numeric ((C.I.P.All[,4]>p[2]) & (p[2]>C.I.P.All[,3])))/nr
```

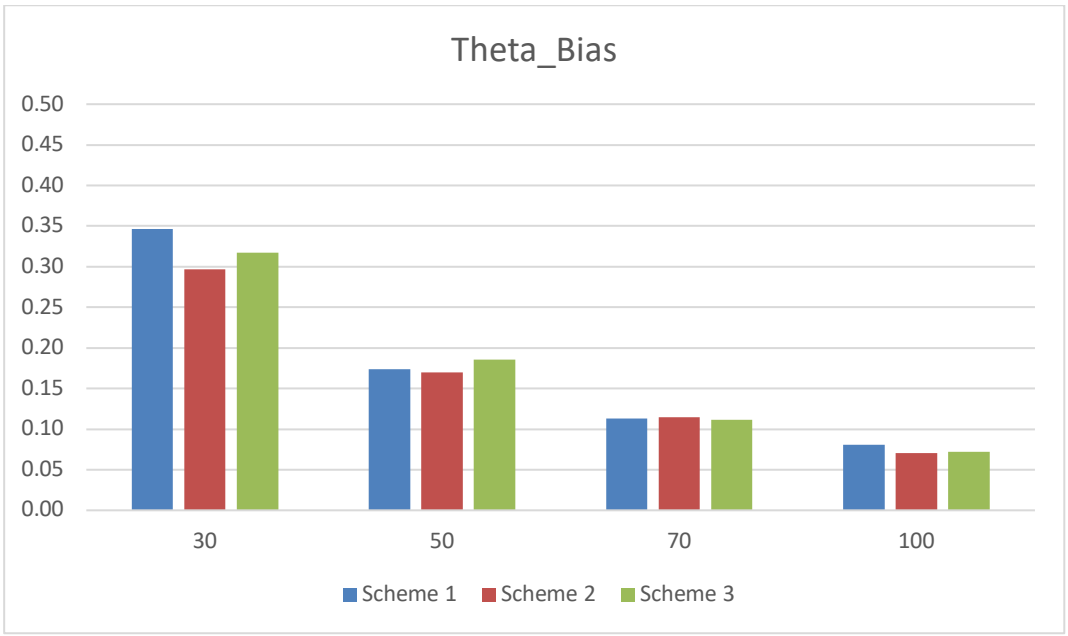
```
C.P.B=sum(as.numeric ((C.I.P.All[,6]>p[3]) & (p[3]>C.I.P.All[,5])))/nr
C.P.RF= sum(as.numeric((l.R.F<P.R.F) & (P.R.F<u.R.F)))/nr
C.P.T.RF= sum(as.numeric ((T.l.R.F<P.R.F) & (P.R.F<T.u.R.F)))/nr  # C.P RF**
C.P.BP.L=sum(as.numeric((Boot.P[,1]<p[1]) & (p[1]<Boot.P[,2])))/nr   #to calculate
the coverage probility for Bootstrap-p
C.P.BP.T=sum(as.numeric((Boot.P[,3]<p[2]) & (p[2]<Boot.P[,4])))/nr
C.P.BP.B=sum(as.numeric((Boot.P[,5]<p[3]) & (p[3]<Boot.P[,6])))/nr
C.P.BP.RF=sum(as.numeric((Boot.P[,7]<P.R.F) & (P.R.F<Boot.P[,8])))/nr
C.P.BT.L=sum(as.numeric((Boot.t[,1]<p[1]) & (p[1]<Boot.t[,2])))/nr    #to calculate
the coverage probility for Bootstrap-t
C.P.BT.T=sum(as.numeric((Boot.t[,3]<p[2]) & (p[2]<Boot.t[,4])))/nr
C.P.BT.B=sum(as.numeric((Boot.t[,5]<p[3]) & (p[3]<Boot.t[,6])))/nr
C.P.BT.RF=sum(as.numeric((Boot.t[,7]<P.R.F) & (P.R.F<Boot.t[,8])))/nr
C.P.All=matrix(c(C.P.L,C.P.T,C.P.B),nrow=3,ncol=1)
rownames(C.P.All)=c("Lambda","Theta","Beta")
colnames(C.P.All)=("Coverage.Probability")
CP.RF=matrix(c(C.P.RF,C.P.T.RF),nrow=2,ncol=1)
rownames(CP.RF)=c("R.F","R.F**")
colnames(CP.RF)=("Coverage Probability")
C.P.Boot.P=matrix(c(C.P.BP.L,C.P.BP.T,C.P.BP.B,C.P.BP.RF),nrow=4,ncol=1)
rownames(C.P.Boot.P)=c("Lambda","Theta","Beta","R.F")
colnames(C.P.Boot.P)=("Coverage.Probability")
C.P.Boot.T=matrix(c(C.P.BT.L,C.P.BT.T,C.P.BT.B,C.P.BT.RF),nrow=4,ncol=1)
rownames(C.P.Boot.T)=c("Lambda","Theta","Beta","R.F")
colnames(C.P.Boot.T)=("Coverage.Probability")
```

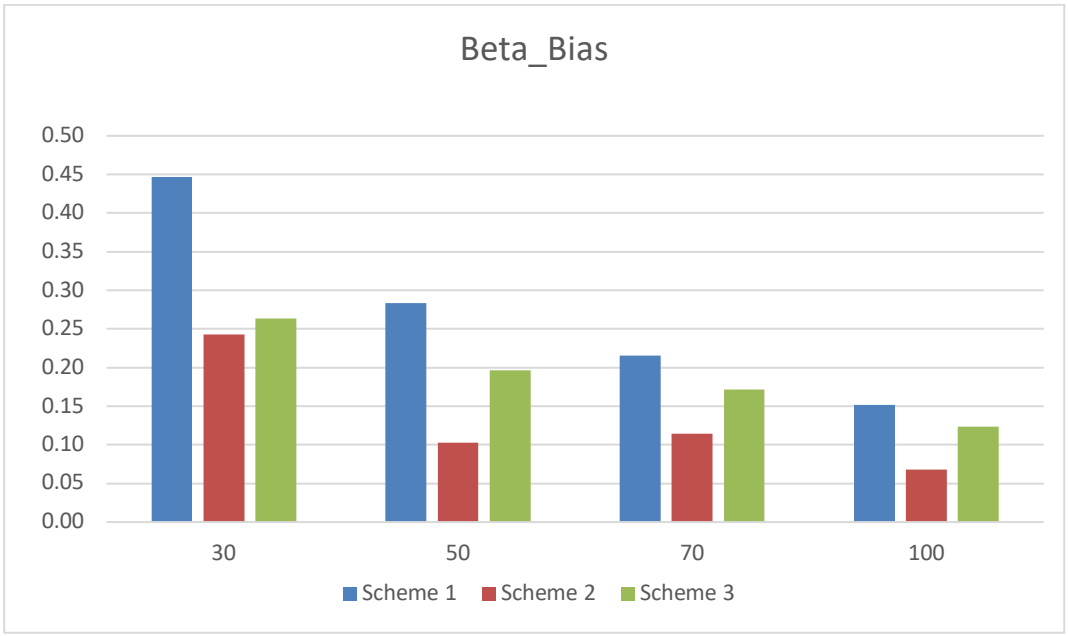
```
#Expect length
Ex.Lambda=sum(C.I.P.All[,2]-C.I.P.All[,1])/nr
Ex.Theta=sum(C.I.P.All[,4]-C.I.P.All[,3])/nr
Ex.Beta=sum(C.I.P.All[,6]-C.I.P.All[,5])/nr
Ex.R.F=sum(ciw)/nr
Ex.T.R.F=sum(Tciw)/nr  #EL RF**
Exp.Len.All=matrix(c(Ex.Lambda,Ex.Theta,Ex.Beta),nrow=3,ncol=1)
rownames(Exp.Len.All)=c("Lambda","Theta","Beta")
colnames(Exp.Len.All)=("Expected.Length")
Exp.Len.RF=matrix(c(Ex.R.F,Ex.T.R.F),nrow=2,ncol=1)
rownames(Exp.Len.RF)=c("R.F","R.F**")
colnames(Exp.Len.RF)=("Expected.Length")
P.E.L.L=sum(Boot.P[,2]-Boot.P[,1])/nr  #to Calculate Expected length Boot-p
P.E.L.T=sum(Boot.P[,4]-Boot.P[,3])/nr
P.E.L.B=sum(Boot.P[,6]-Boot.P[,5])/nr
P.E.L.R=sum(Boot.P[,8]-Boot.P[,7])/nr
T.E.L.L=sum(Boot.t[,2]-Boot.t[,1])/nr  #to Calculate Expected length Boot-t
T.E.L.T=sum(Boot.t[,4]-Boot.t[,3])/nr
T.E.L.B=sum(Boot.t[,6]-Boot.t[,5])/nr
T.E.L.R=sum(Boot.t[,8]-Boot.t[,7])/nr
Exp.Len.Boot.P=matrix(c(P.E.L.L,P.E.L.T,P.E.L.B,P.E.L.R),nrow=4,ncol=1)
rownames(Exp.Len.Boot.P)=c("Boot.P.Lambda","Boot.P.Theta","Boot.P.Beta","Boot
.P.Reliability")
colnames(Exp.Len.Boot.P)=("Expected Length")
```

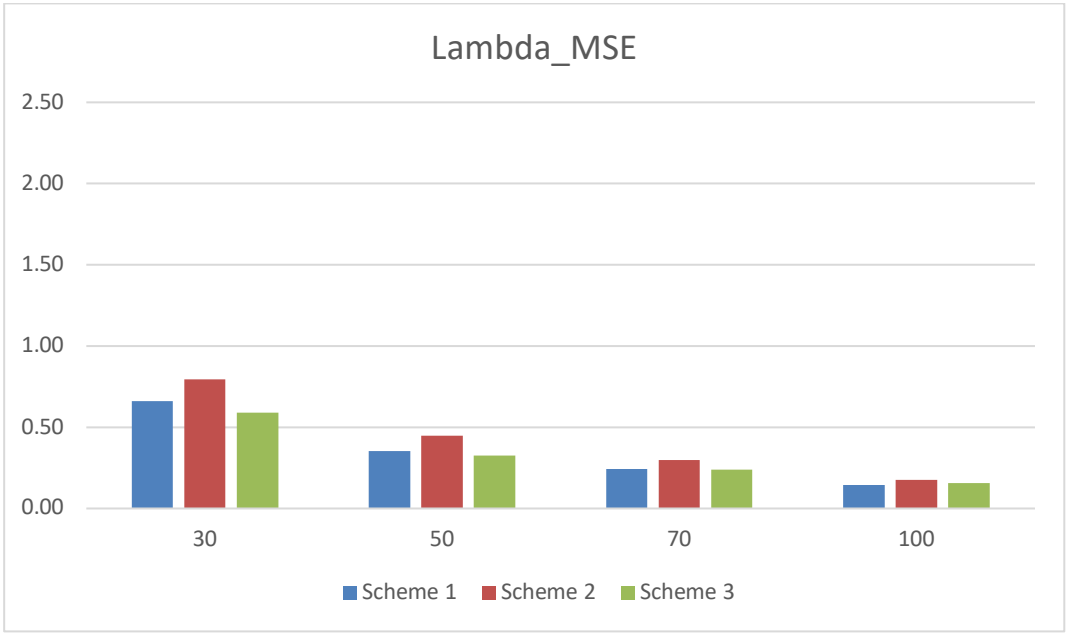
```
  Exp.Len.Boot.t=matrix(c(T.E.L.L,T.E.L.T,T.E.L.B,T.E.L.R),nrow=4,ncol=1)
rownames(Exp.Len.Boot.t)=c("Boot.t.Lambda","Boot.t.Theta","Boot.t.Beta","Boot.t.
Reliability")
  colnames(Exp.Len.Boot.t)=("Expected Length")
  #Printing Results
  # Parameter Estimates, their Variances and Confidence Intervals (MLE)
    #  print(list("All  MLE"=P.All,"Variance  MLE"=Var.P.All,"Asymptotic  C.I
MLE"=C.I.P.All))
  # Reliability Function
    # print(list("RF"=RF.All, "Variance RF"=Var.RF.All))
    # print(list("Ays.CI.RF"=C.I.RF))
  # Bias and MSE (from Main loop)
  print(list("Bias"=Bias,"MSE"=MSE,"BiasRF"=BiasRF,"MSERF"=MSERF))
  #Coverage Probability and  #Expected length
  print(list("MLE EL"=Exp.Len.All,"R.F EL"=Exp.Len.RF))
  print(list("MLE C.P"=C.P.All,"R.F C.P"=CP.RF))
  print(list("Boot-p.EL"=Exp.Len.Boot.P,"Boot-t.EL"=Exp.Len.Boot.t))
  print(list("Boot-p C.P"=C.P.Boot.P,"Boot-t C.P"=C.P.Boot.T))
```

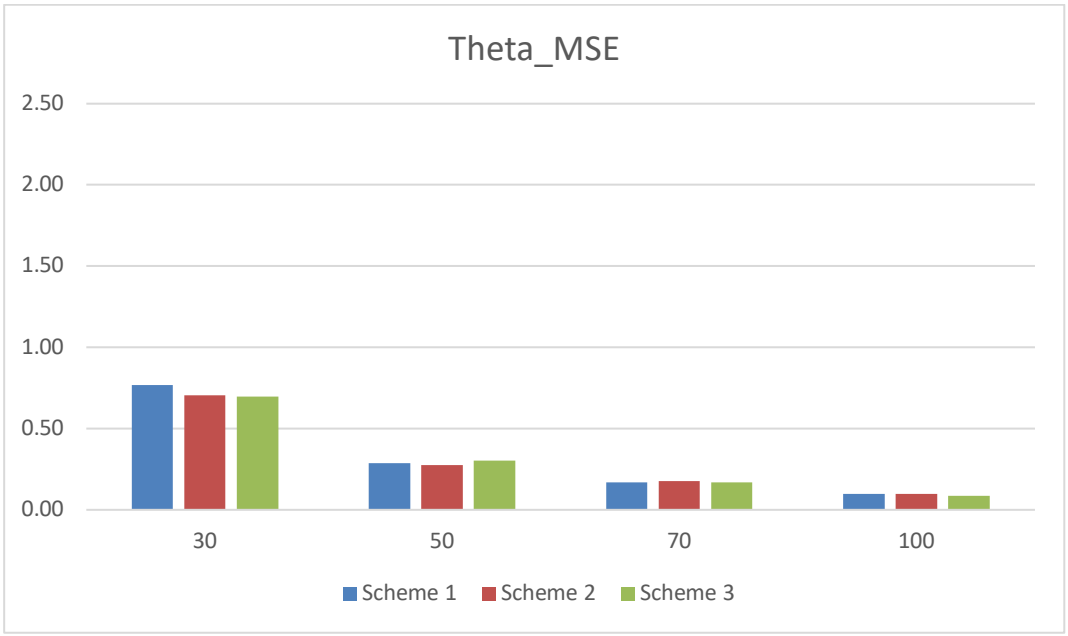
APPENDIX B: BARCHART FOR BIAS AND MSE FOR THE MLEs

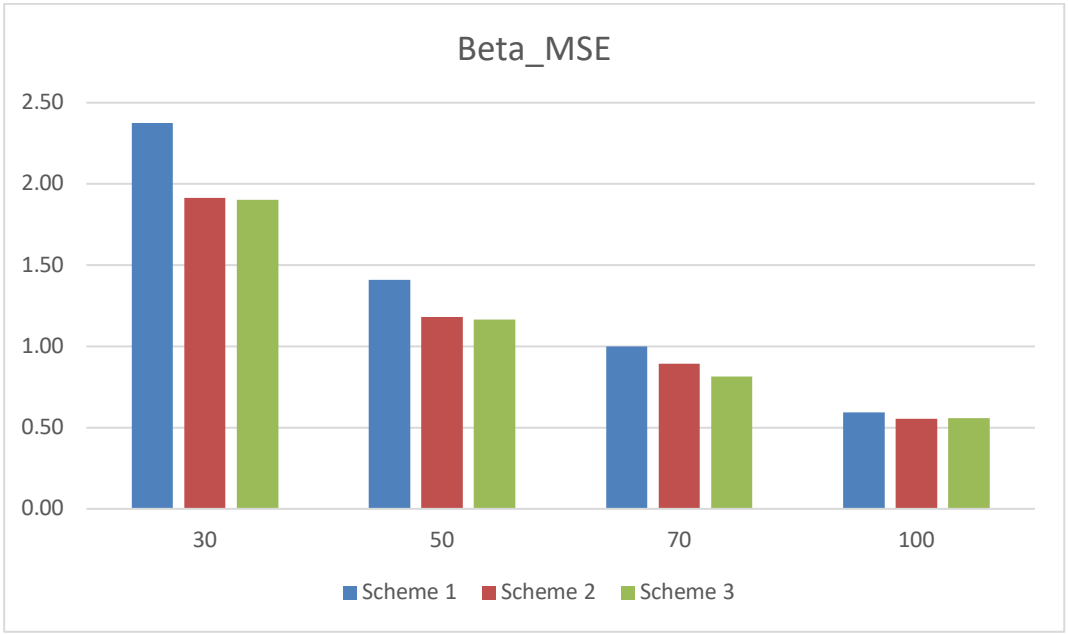




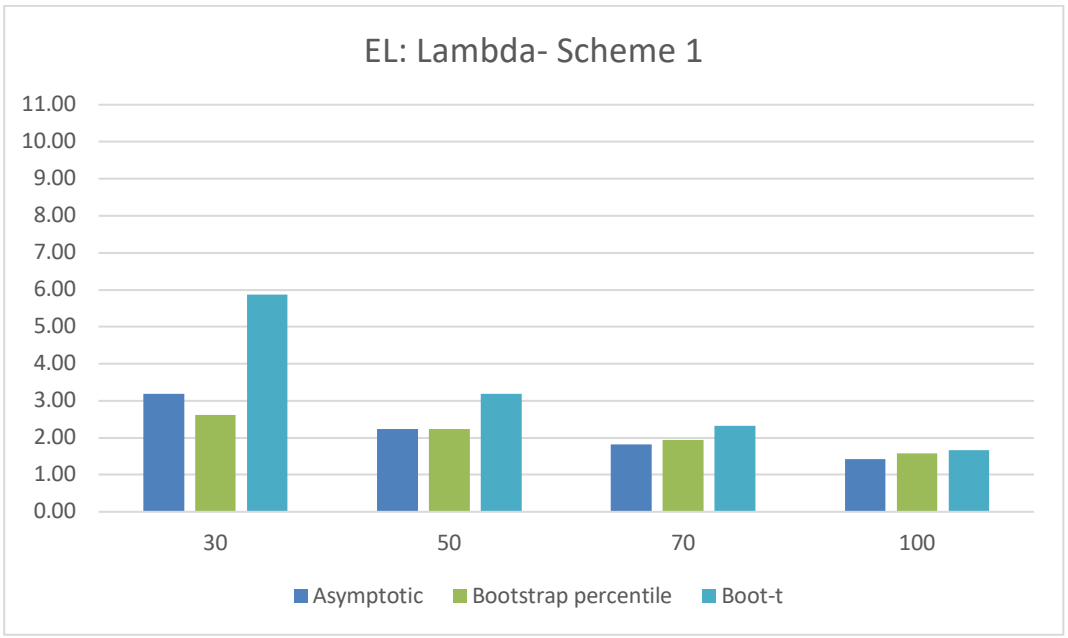

Beta_Bias
0.50
0.45
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
30
50
70
100
Scheme 1
Scheme 2
Scheme 3

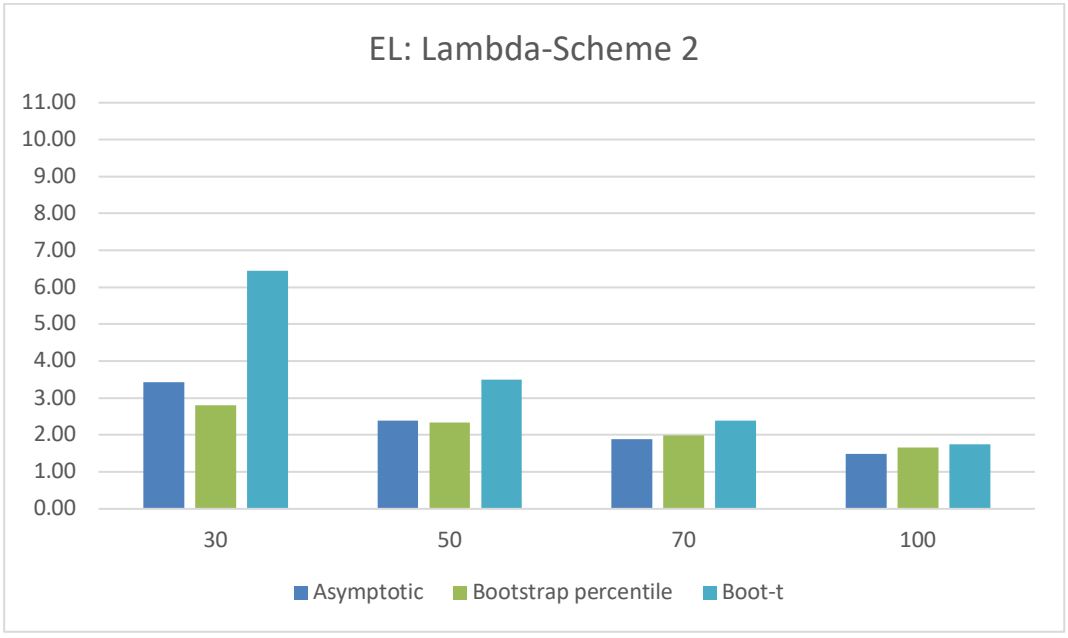

Lambda_MSE
2.50
2.00
1.50
1.00
0.50
0.00
30
50
70
100
Scheme 1
Scheme 2
Scheme 3

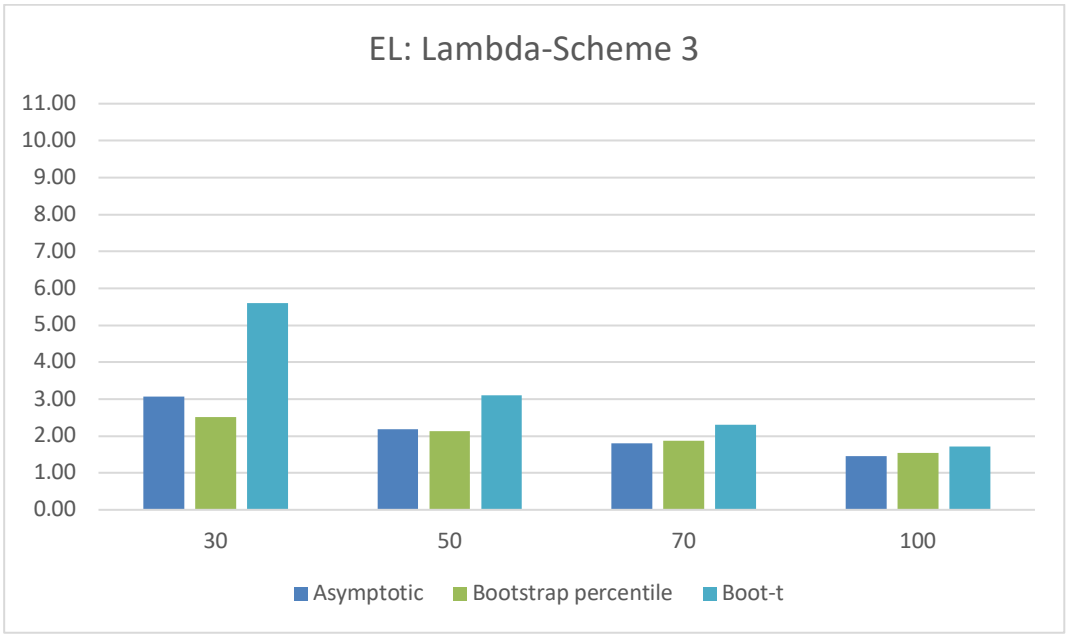

Theta_MSE
2.50
2.00
1.50
1.00
0.50
0.00
30
50
70
100
Scheme 1
Scheme 2
Scheme 3

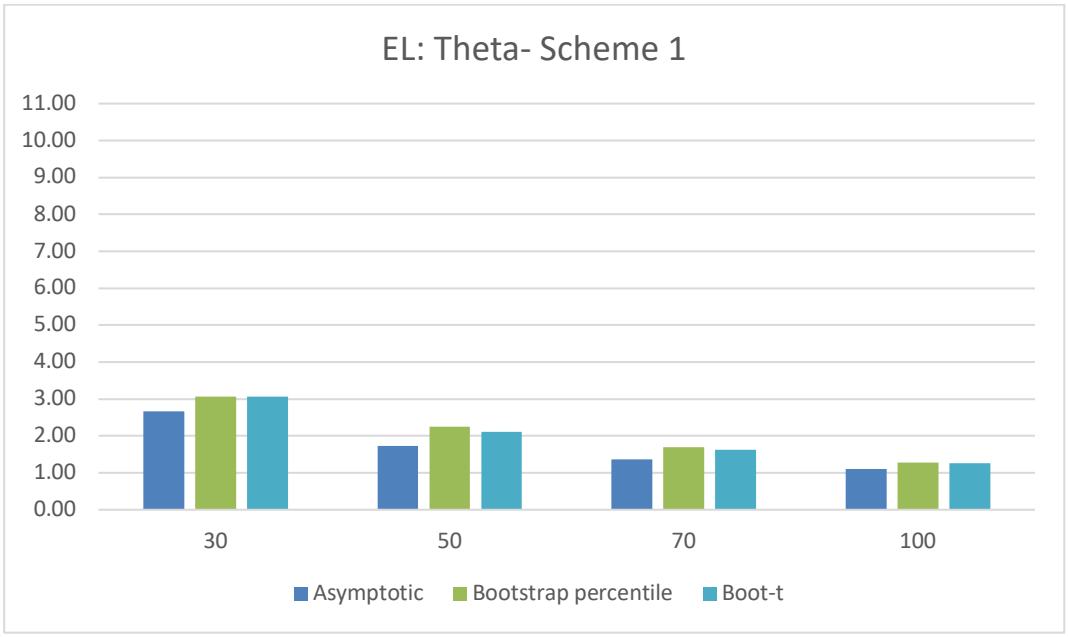

Beta_MSE
2.50
2.00
1.50
1.00
0.50
0.00
30
50
70
100
Scheme 1
Scheme 2
Scheme 3

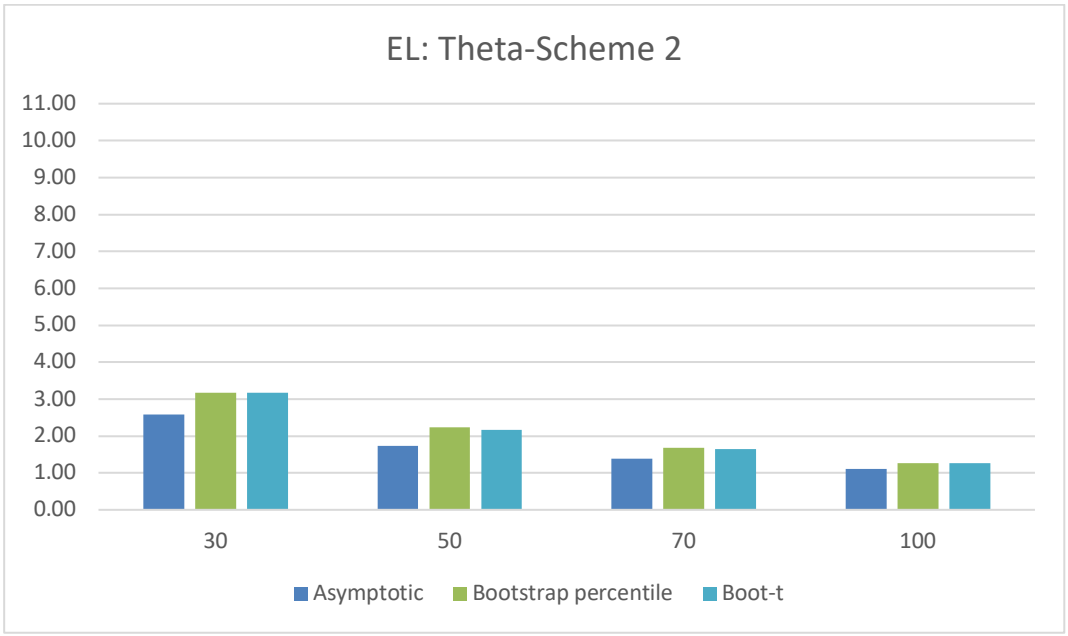
APPENDIX C: BARCHART FOR EXPECTED LENGTH (EL) AND COVERAGE PROBABILITY (CP) FOR THE MLEs

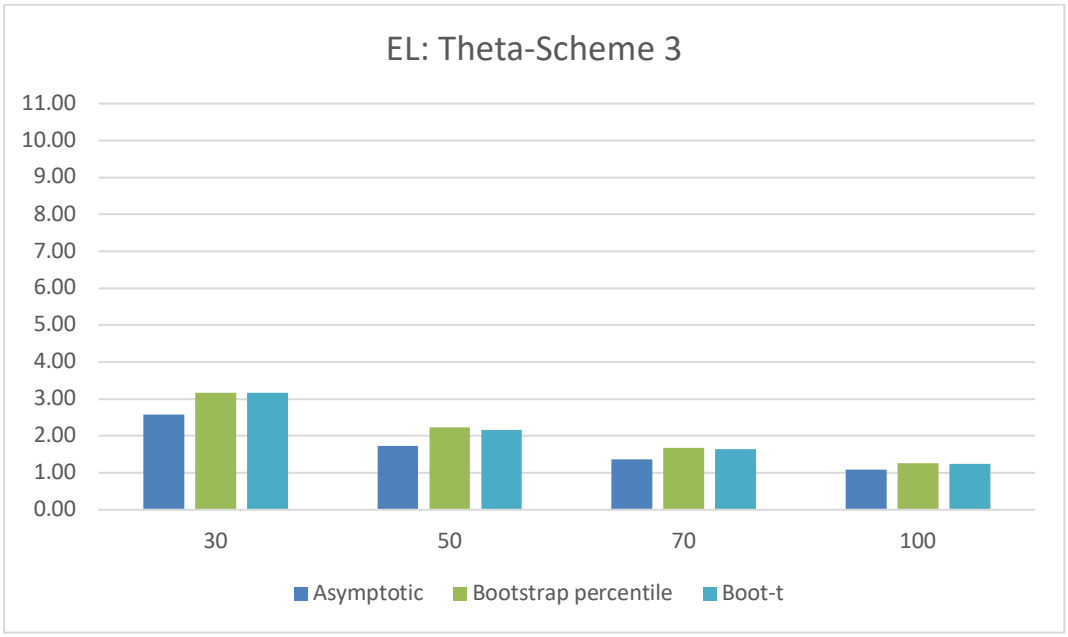


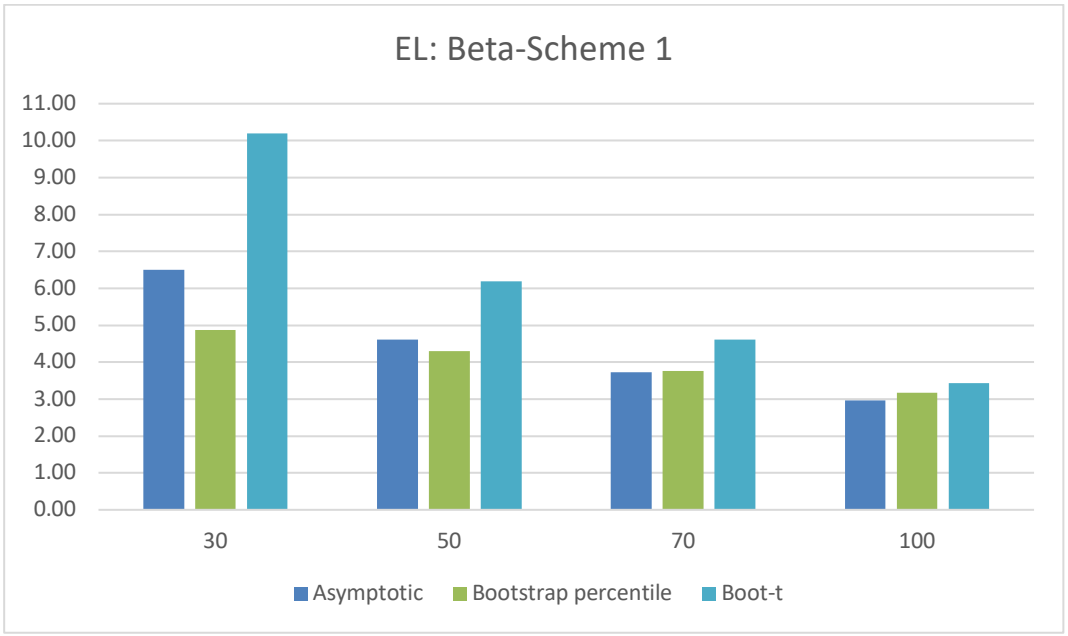


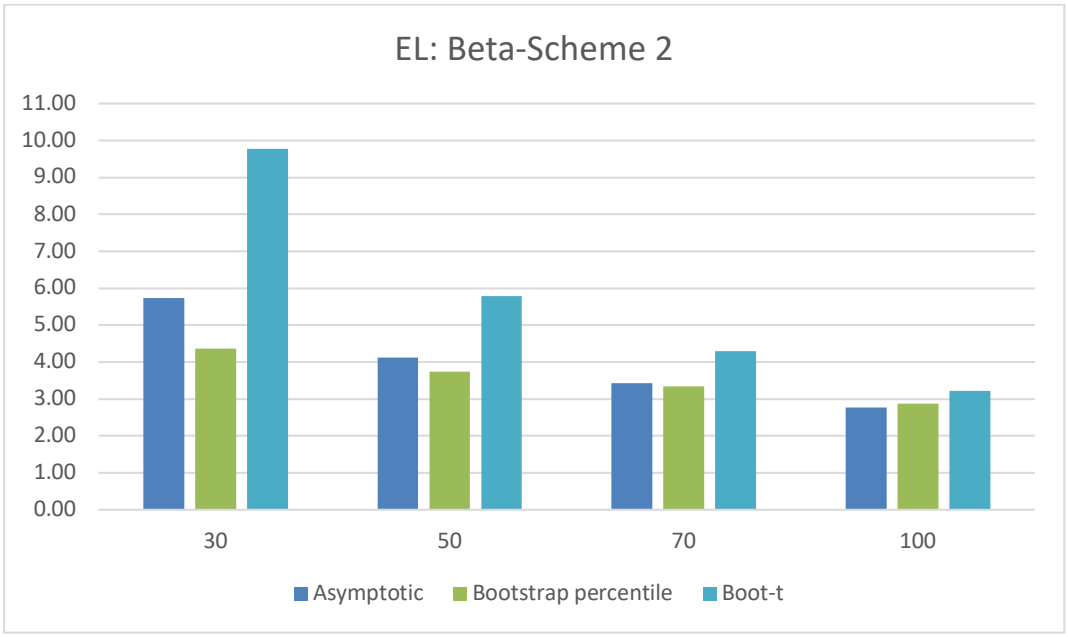

EL: Lambda-Scheme 3
11.00
10.00
9.00
8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

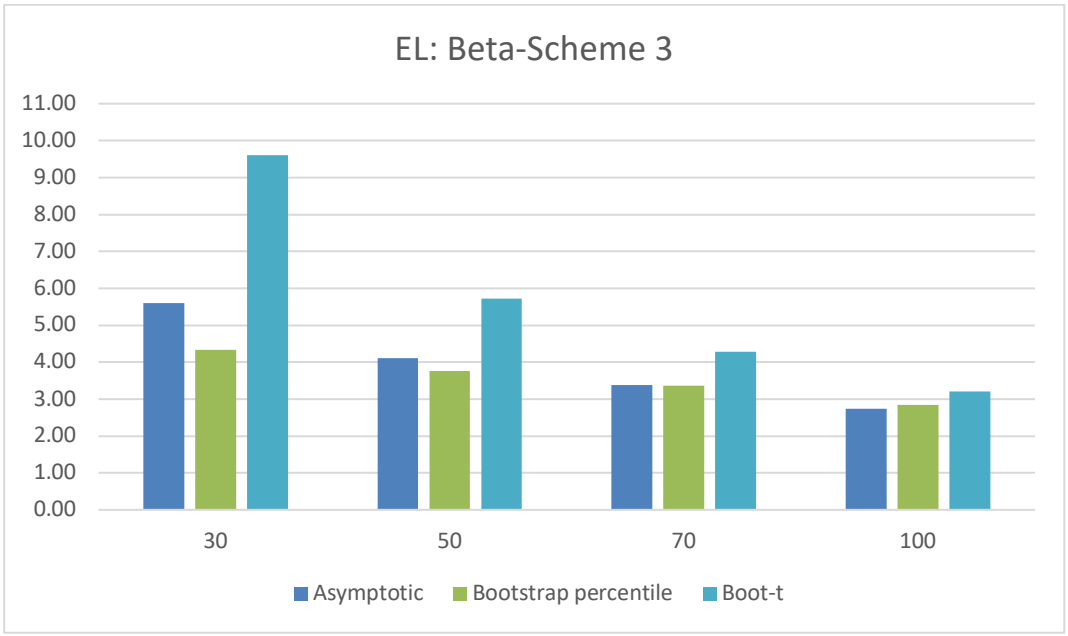

EL: Theta- Scheme 1
11.00
10.00
9.00
8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

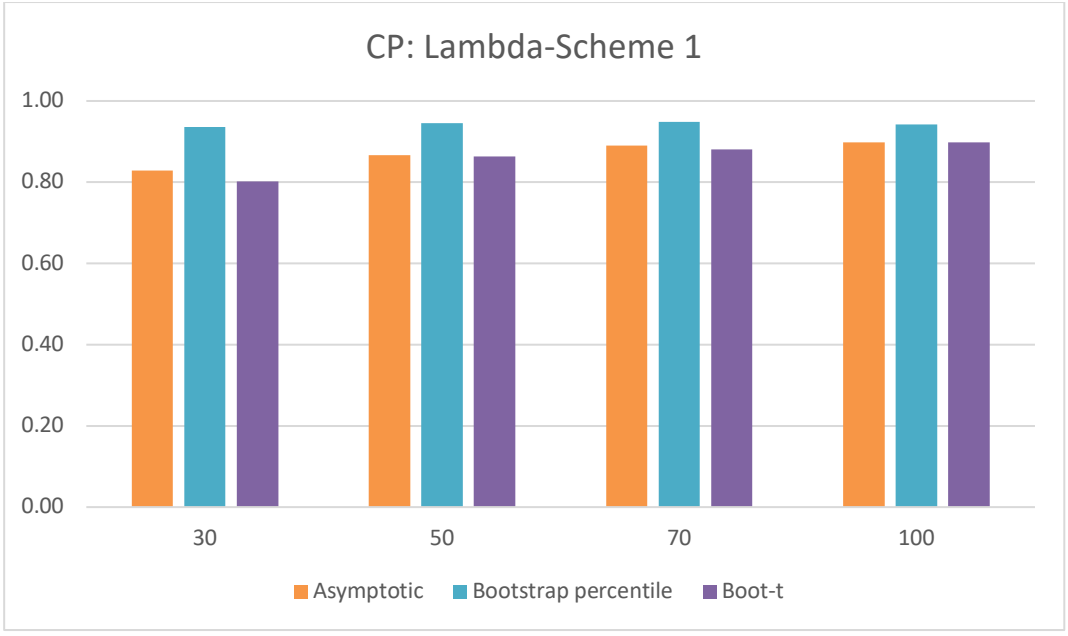

EL: Theta-Scheme 2
11.00
10.00
9.00
8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

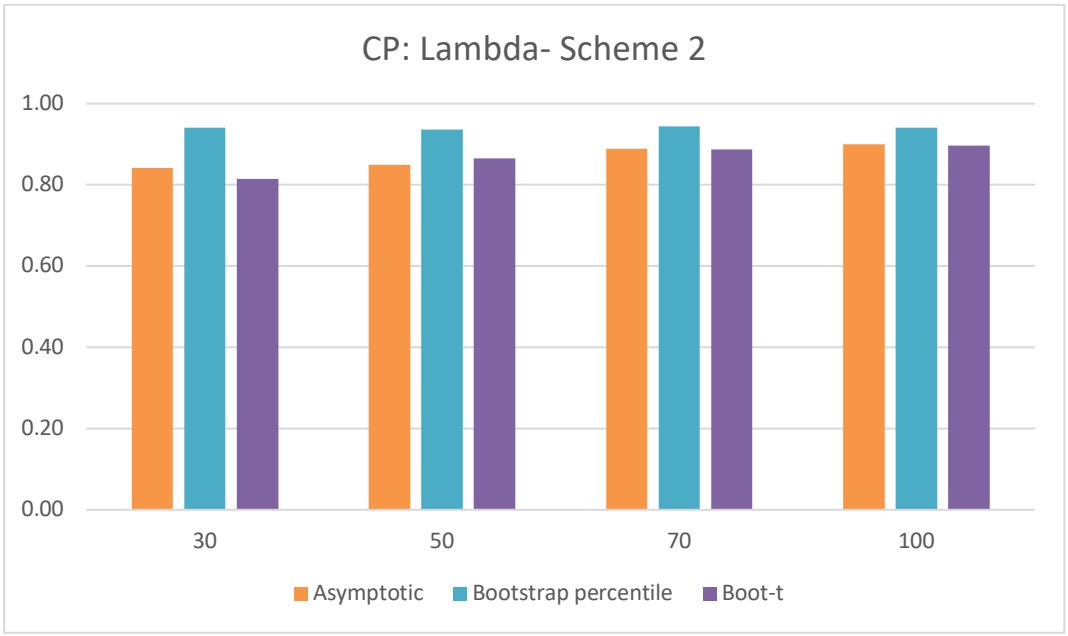

EL: Theta-Scheme 3
11.00
10.00
9.00
8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

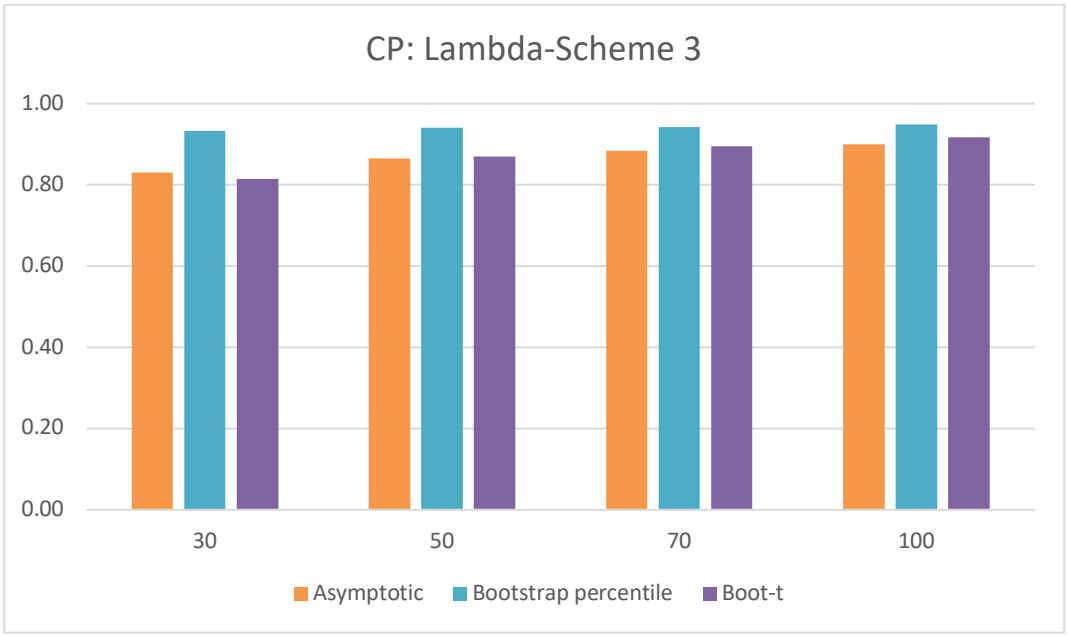

EL: Beta-Scheme 1
11.00
10.00
9.00
8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

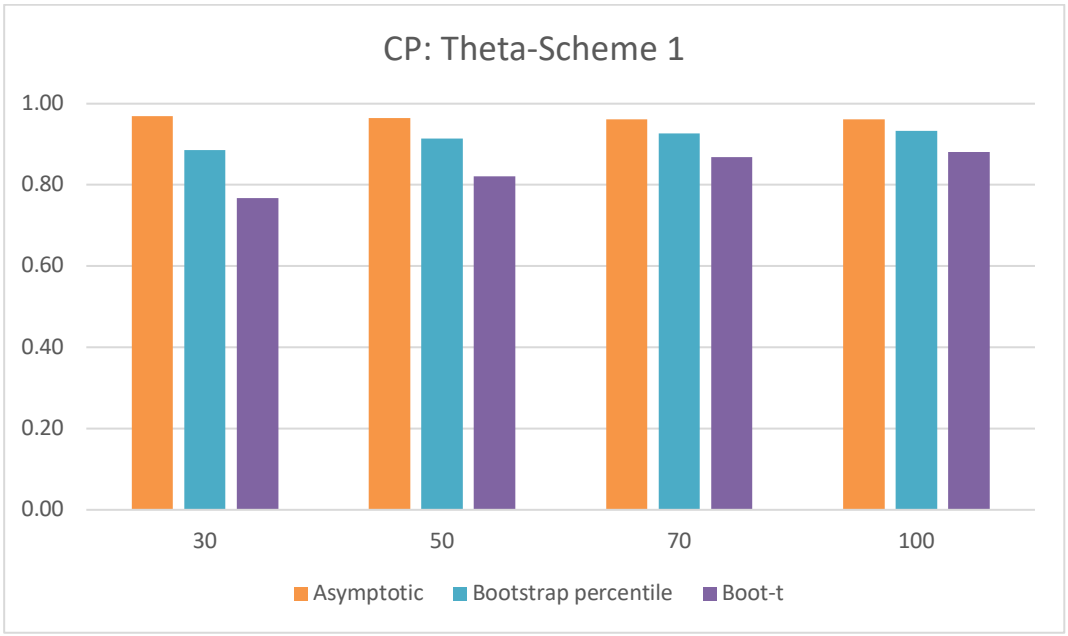

EL: Beta-Scheme 2
11.00
10.00
9.00
8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

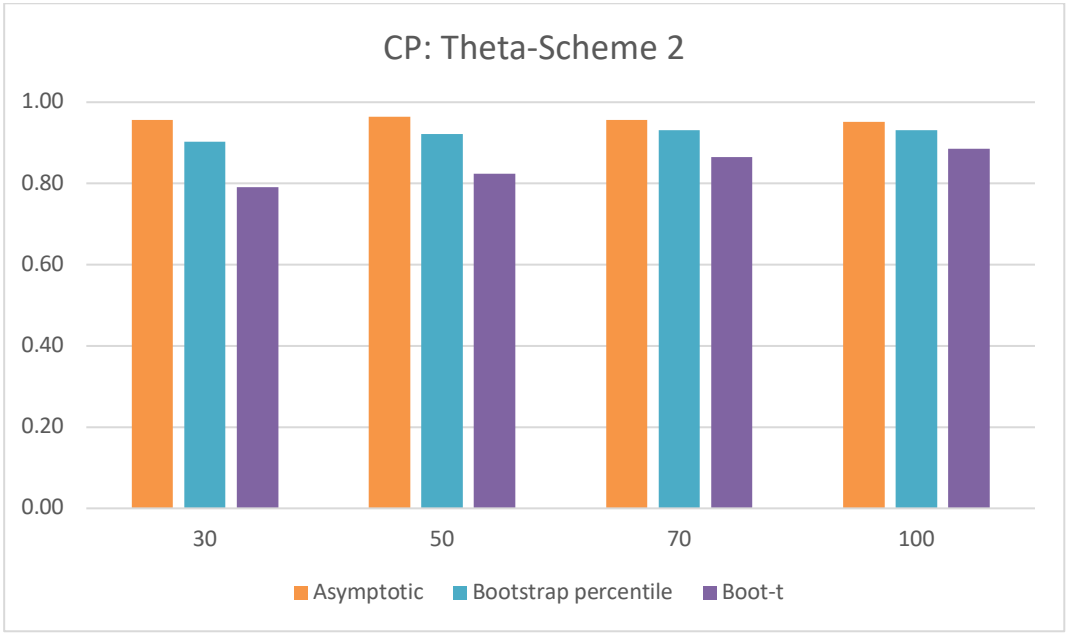

EL: Beta-Scheme 3
11.00
10.00
9.00
8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

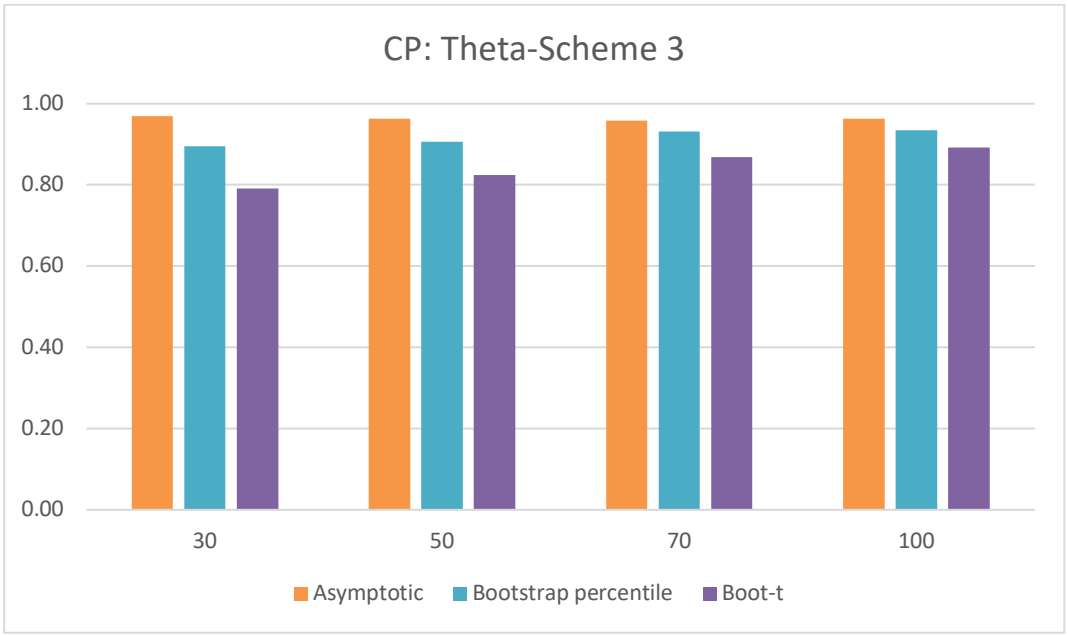

CP: Lambda-Scheme 1
1.00
0.80
0.60
0.40
0.20
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

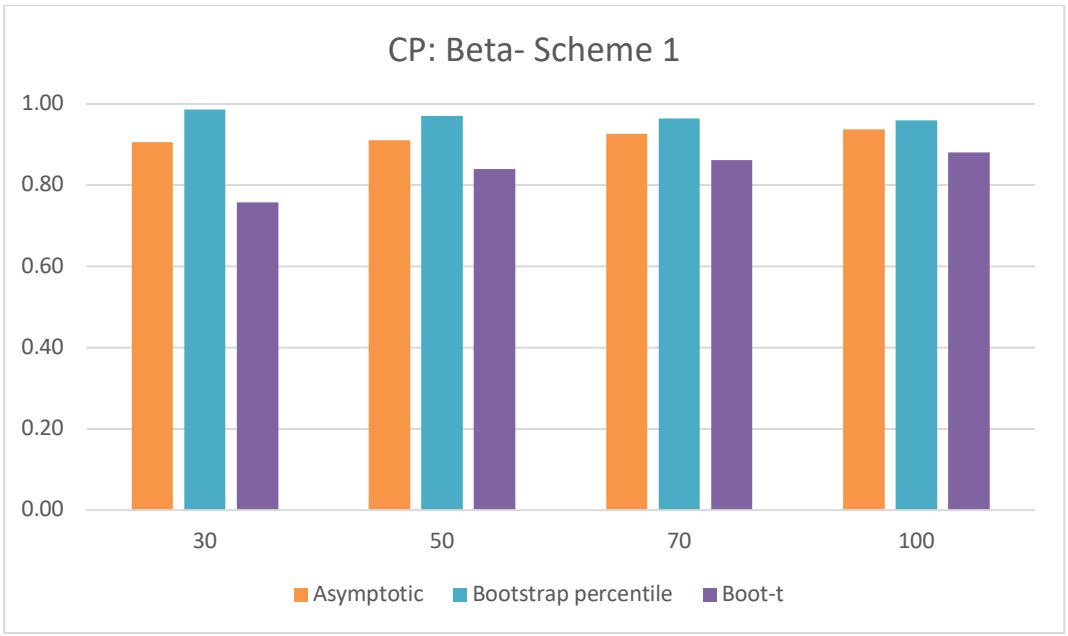

CP: Lambda- Scheme 2
1.00
0.80
0.60
0.40
0.20
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

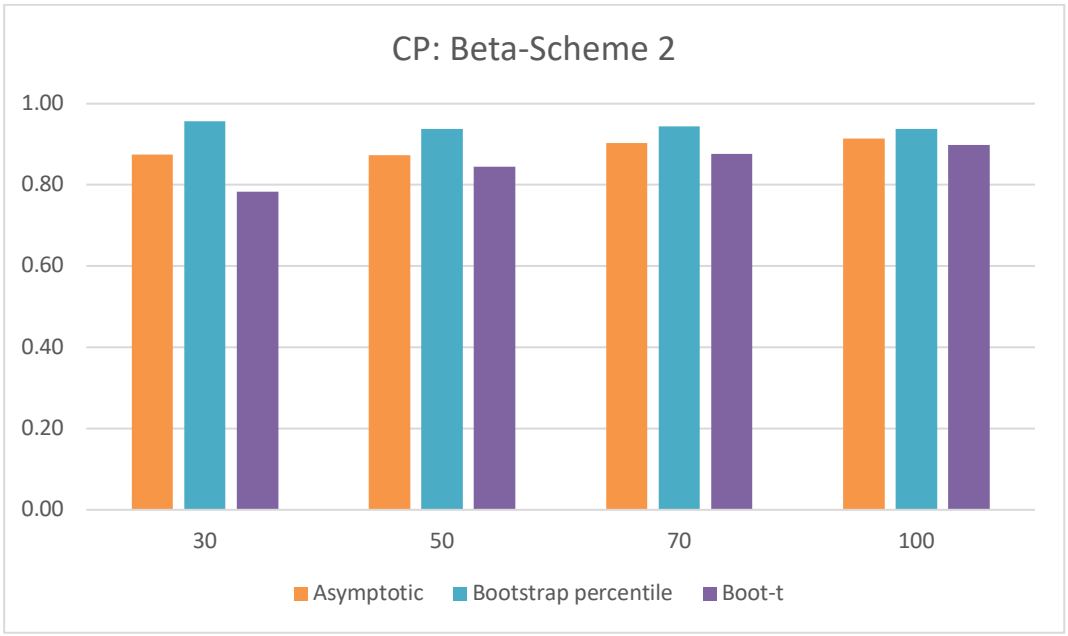

CP: Lambda-Scheme 3
1.00
0.80
0.60
0.40
0.20
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

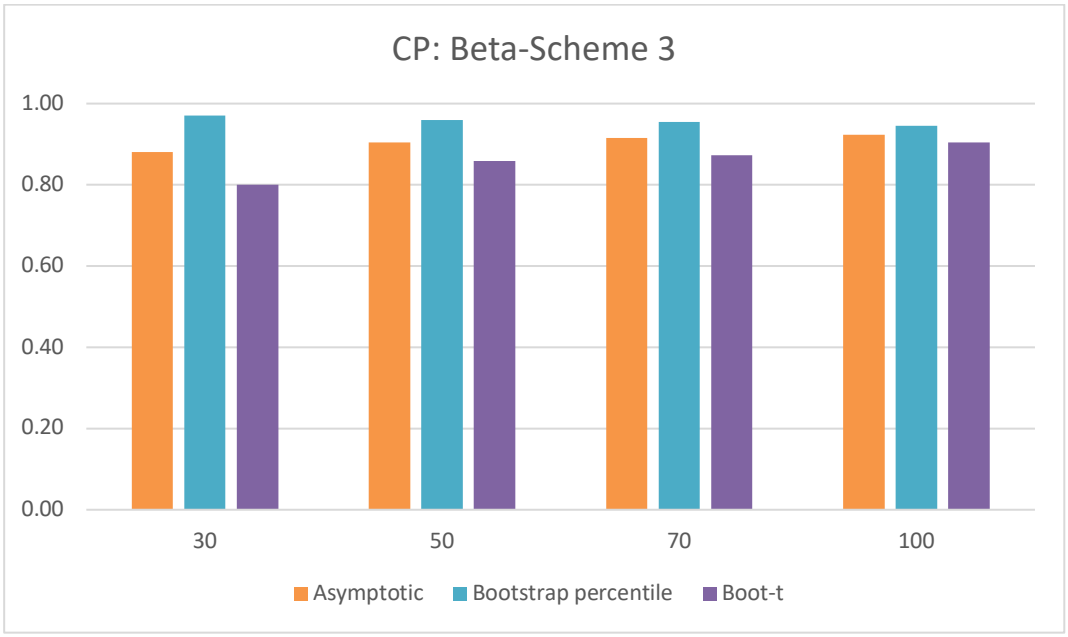

CP: Theta-Scheme 1
1.00
0.80
0.60
0.40
0.20
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t


CP: Theta-Scheme 2
1.00
0.80
0.60
0.40
0.20
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

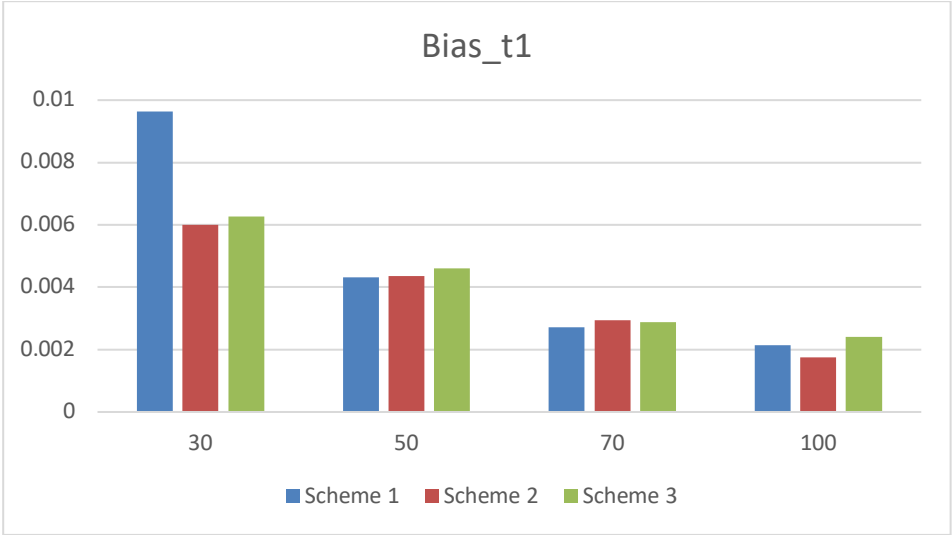

CP: Theta-Scheme 3
1.00
0.80
0.60
0.40
0.20
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

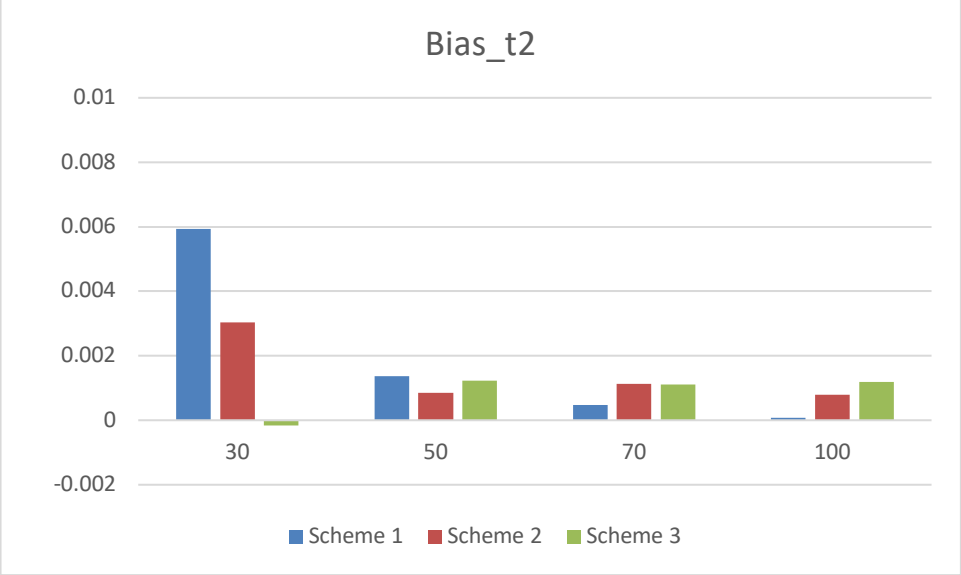

CP: Beta- Scheme 1
1.00
0.80
0.60
0.40
0.20
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

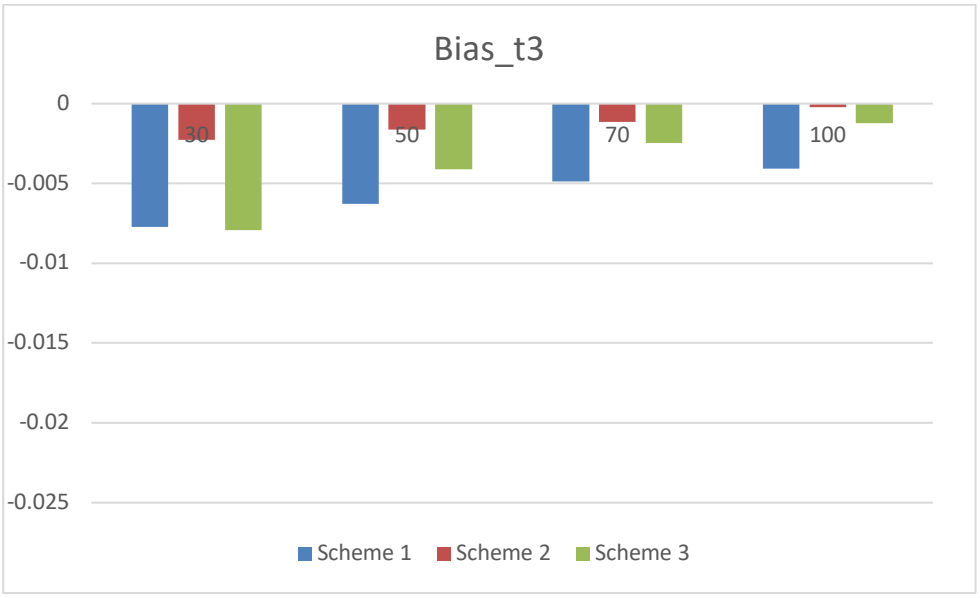

CP: Beta-Scheme 2
1.00
0.80
0.60
0.40
0.20
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

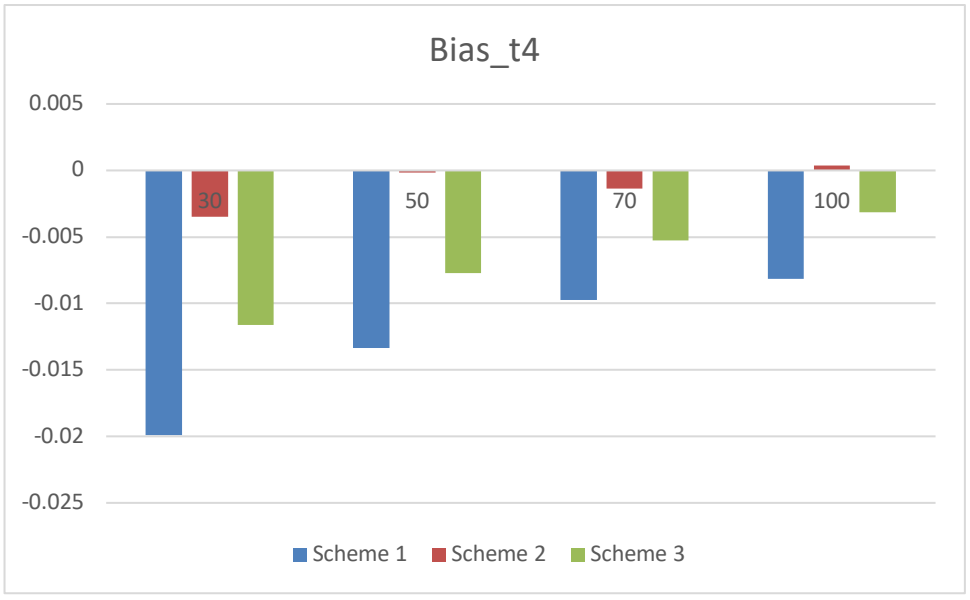

CP: Beta-Scheme 3
1.00
0.80
0.60
0.40
0.20
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

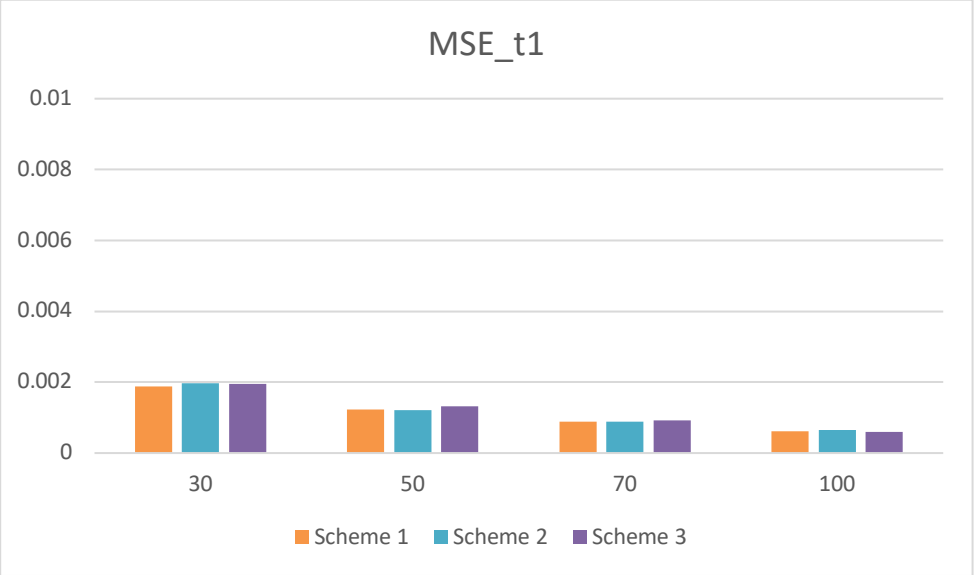
APPENDIX D: BARCHART FOR BIAS AND MSE FOR RF.

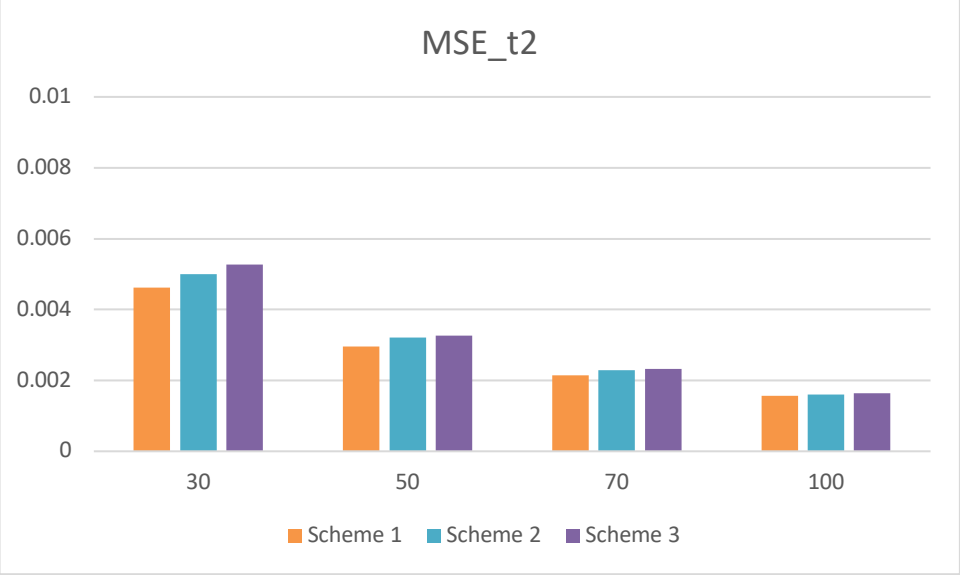




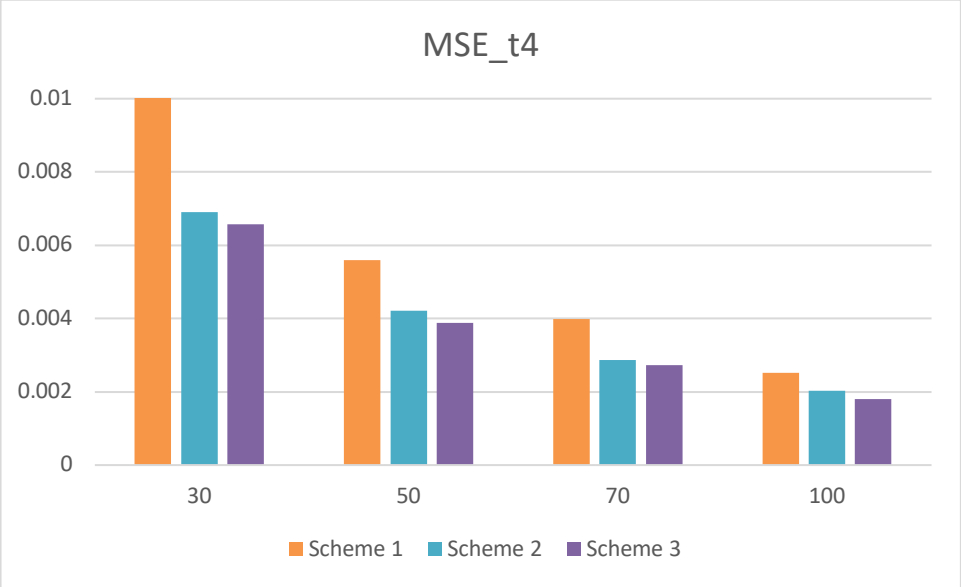

Bias_t3
0
-0.005
-0.01
-0.015
-0.02
-0.025
30
50
70
100
Scheme 1
Scheme 2
Scheme 3


Bias_t4
0.005
0
-0.005
-0.01
-0.015
-0.02
-0.025
30
50
70
100
Scheme 1
Scheme 2
Scheme 3

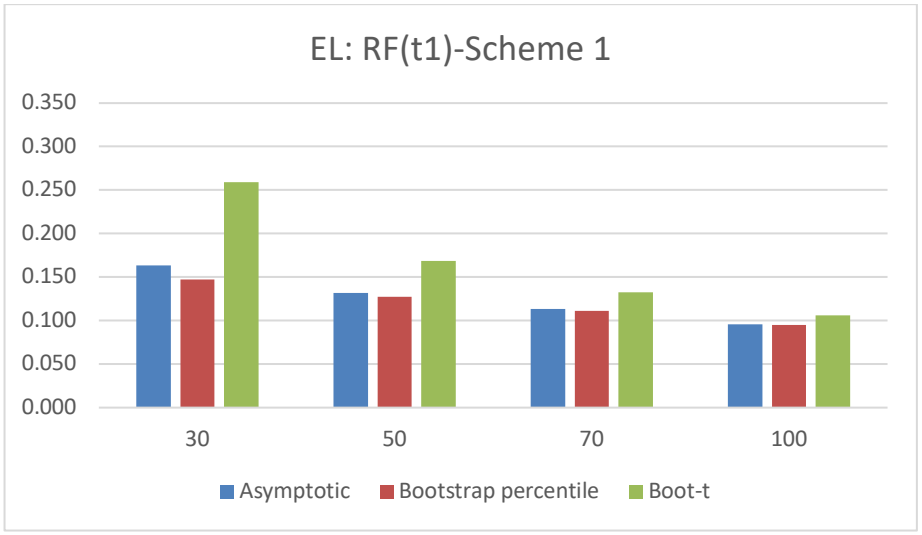

MSE_t1
0.01
0.008
0.006
0.004
0.002
0
30
50
70
100
Scheme 1
Scheme 2
Scheme 3

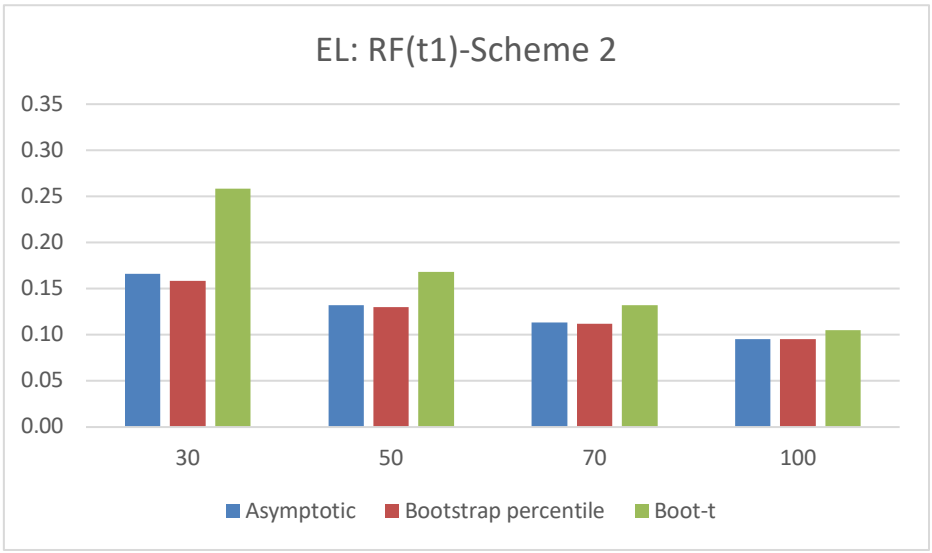

MSE_t2
0.01
0.008
0.006
0.004
0.002
0
30
50
70
100
Scheme 1
Scheme 2
Scheme 3

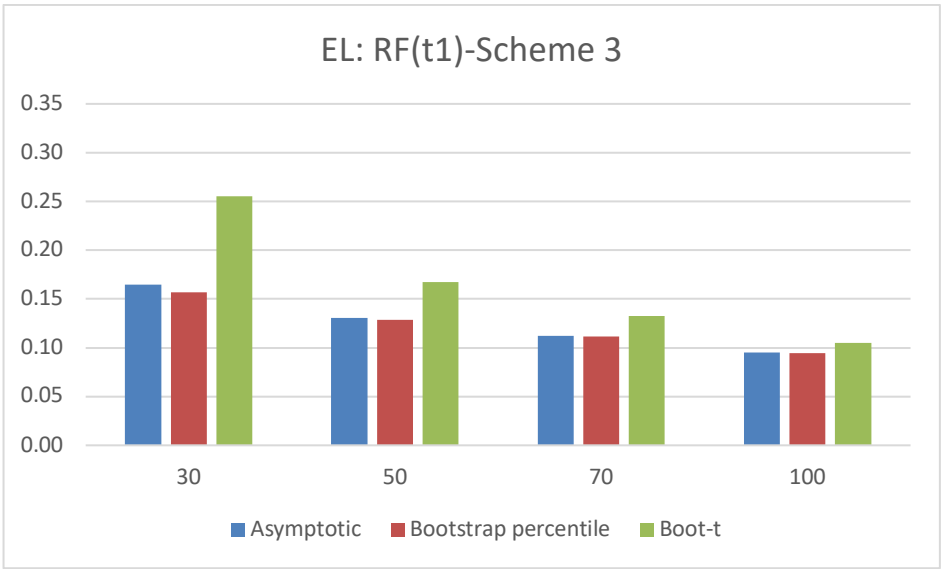

MSE_t3
0.01
0.008
0.006
0.004
0.002
0
30
50
70
100
Scheme 1
Scheme 2
Scheme 3

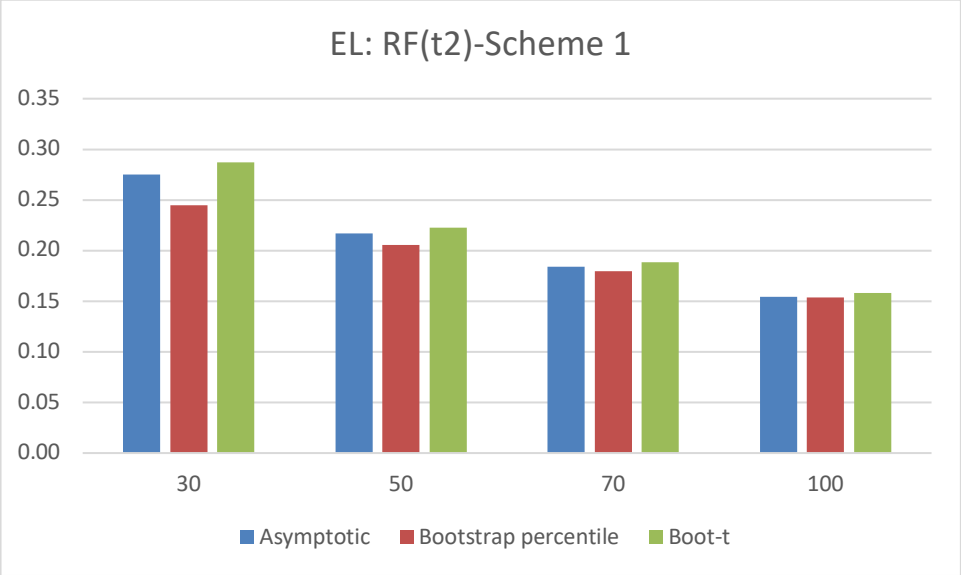

MSE_t4
0.01
0.008
0.006
0.004
0.002
0
30
50
70
100
Scheme 1
Scheme 2
Scheme 3

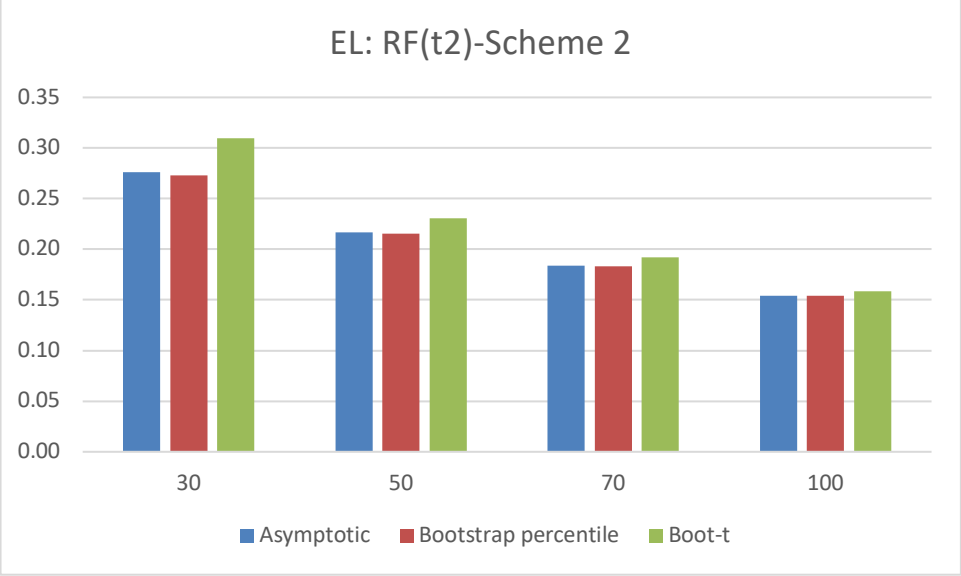
APPENDIX E: BARCHART FOR EXPECTED LENGTH (EL) AND COVERAGE PROBABILITY (CP) FOR RF.

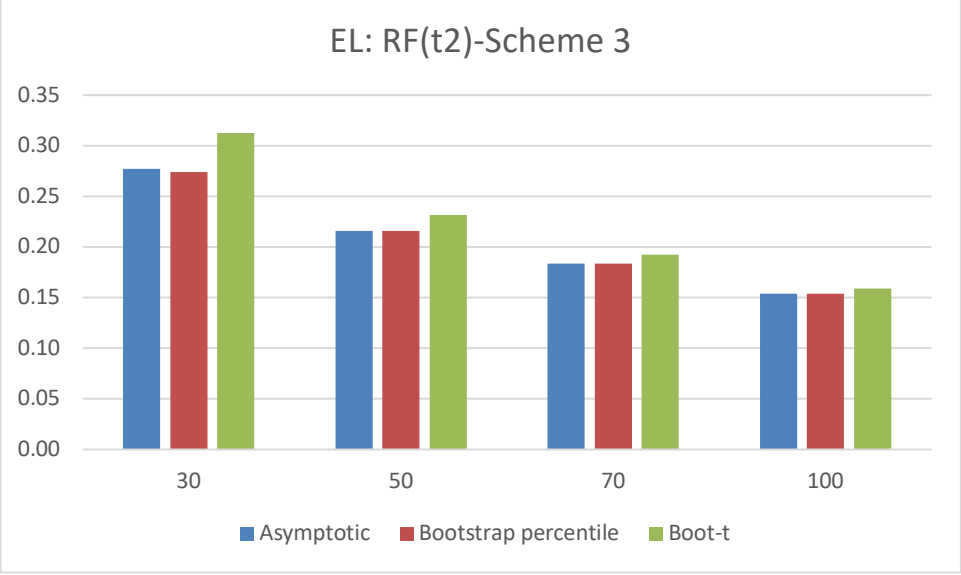


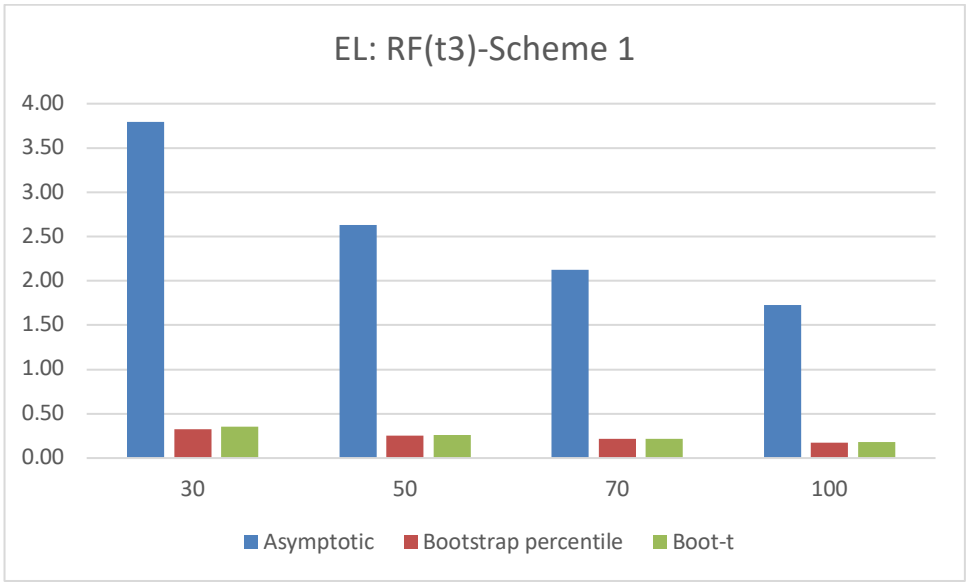


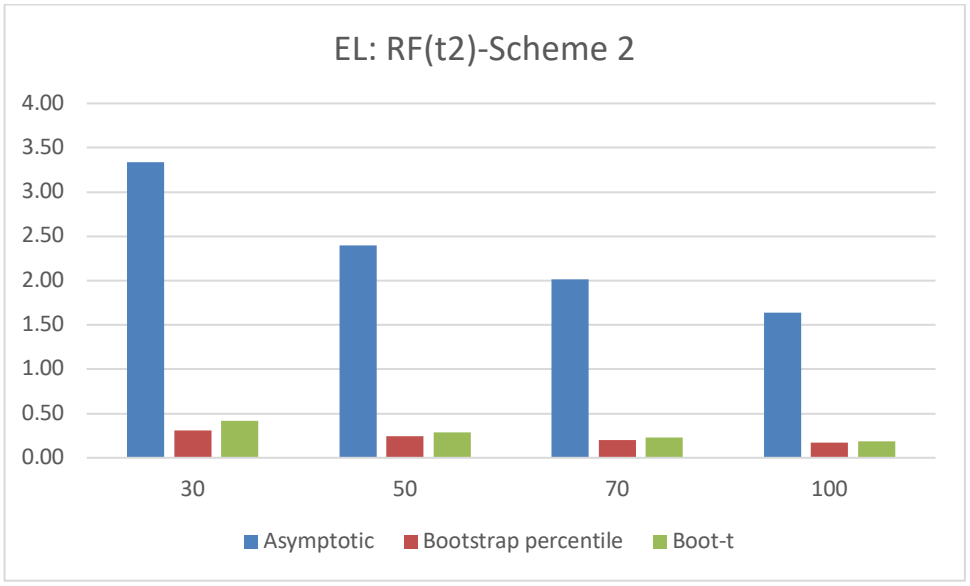


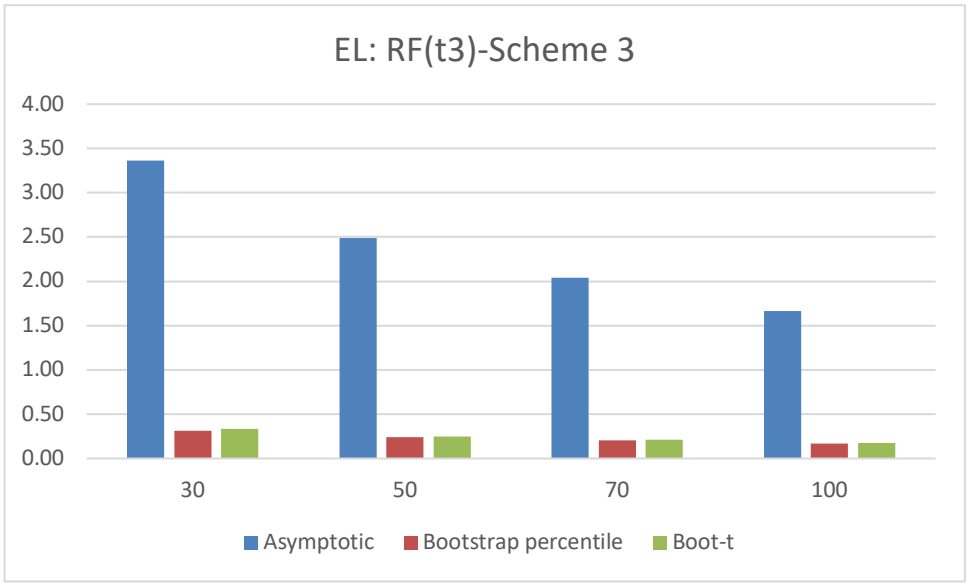

EL: RF(t2)-Scheme 1
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

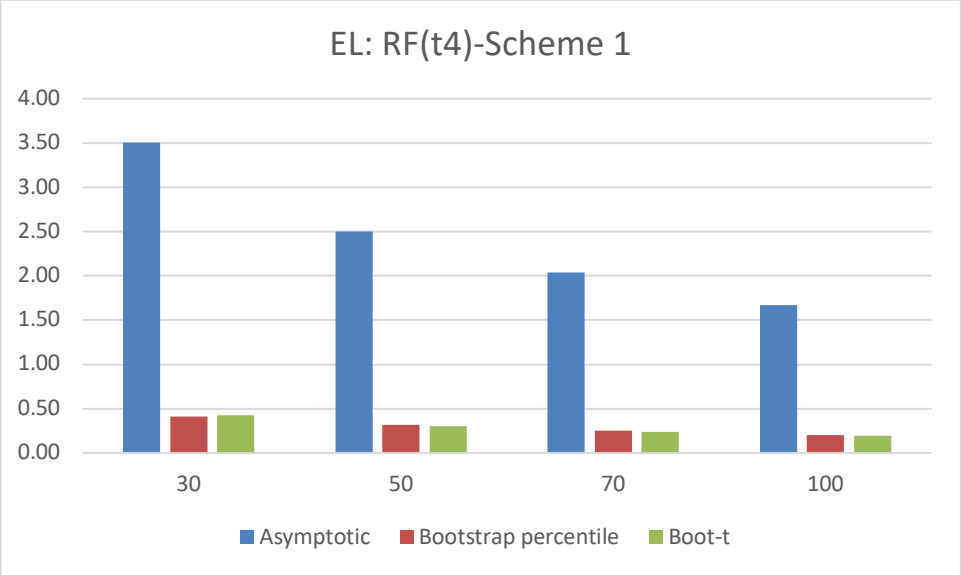

EL: RF(t2)-Scheme 2
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

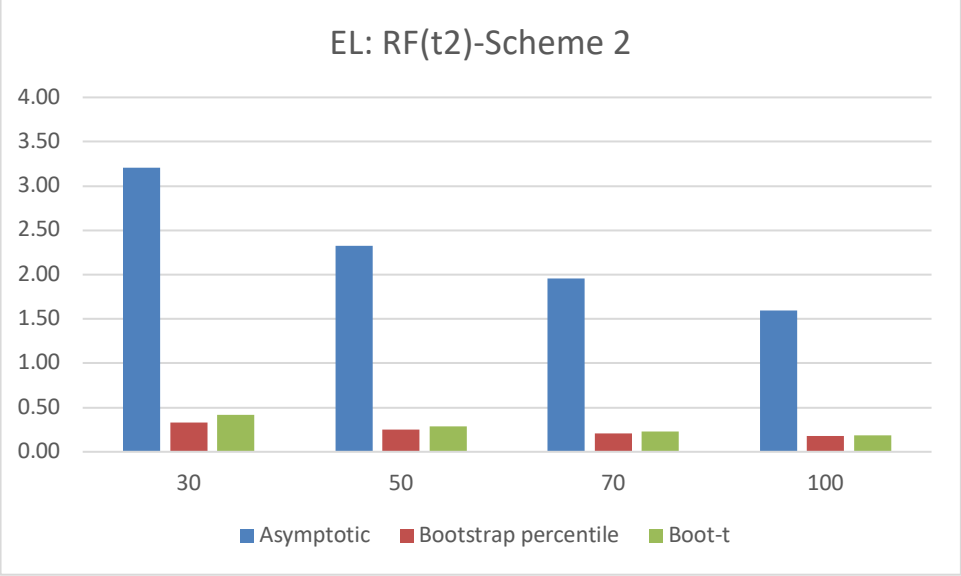

EL: RF(t2)-Scheme 3
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

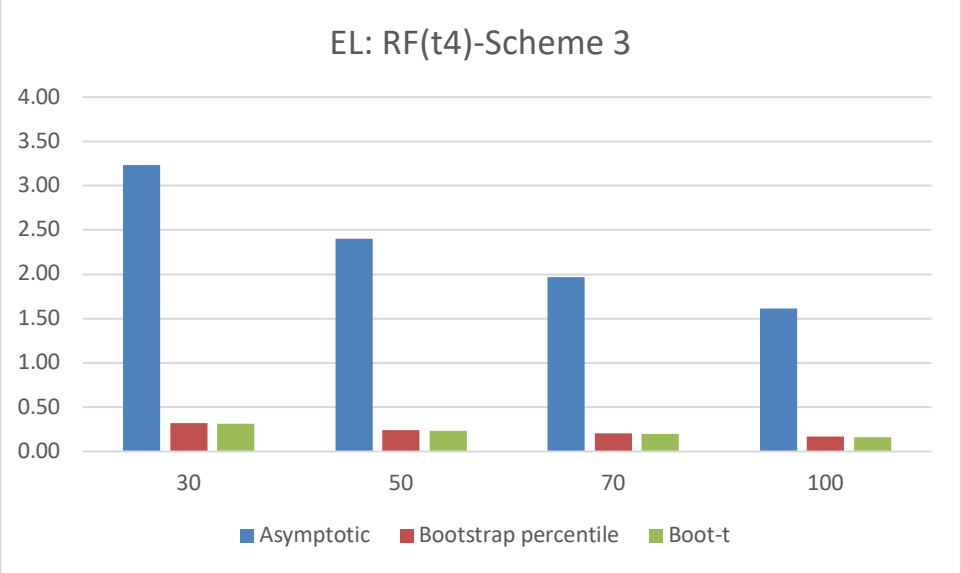

EL: RF(t3)-Scheme 1
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

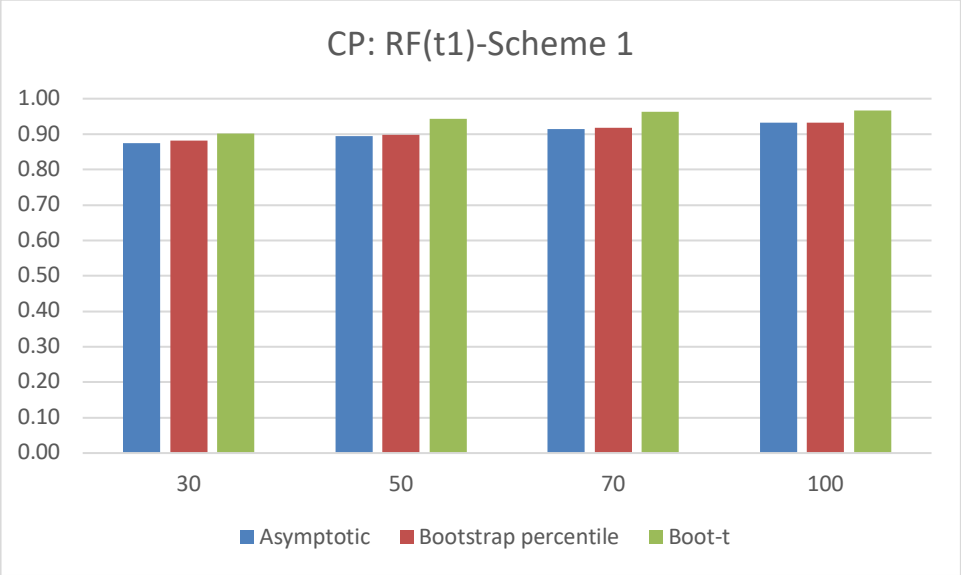

EL: RF(t2)-Scheme 2
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

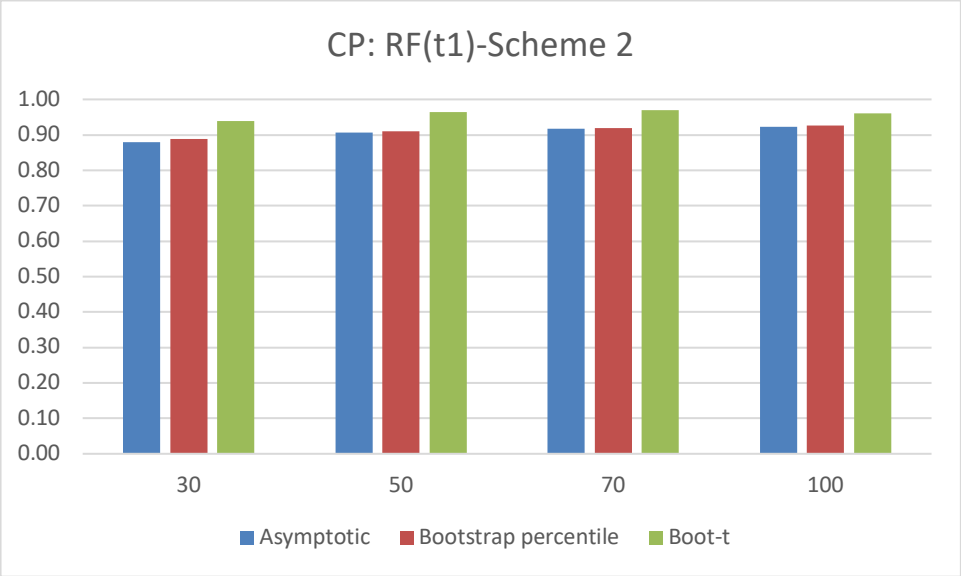

EL: RF(t3)-Scheme 3
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

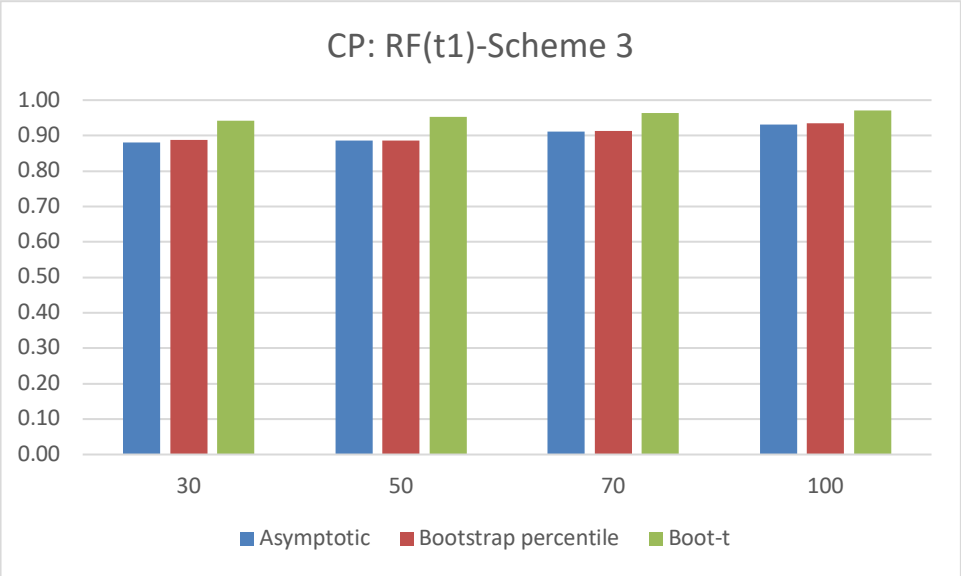

EL: RF(t4)-Scheme 1
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

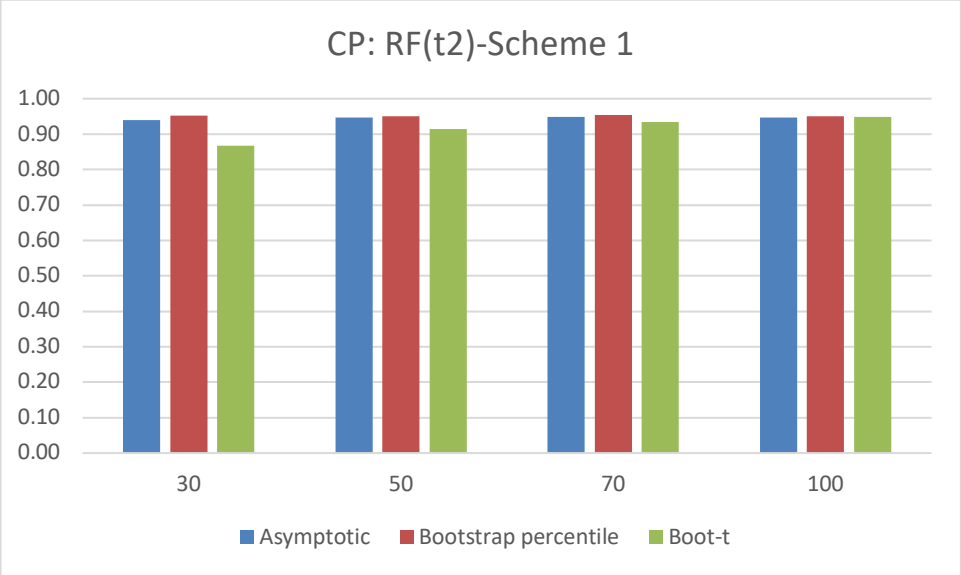

EL: RF(t2)-Scheme 2
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

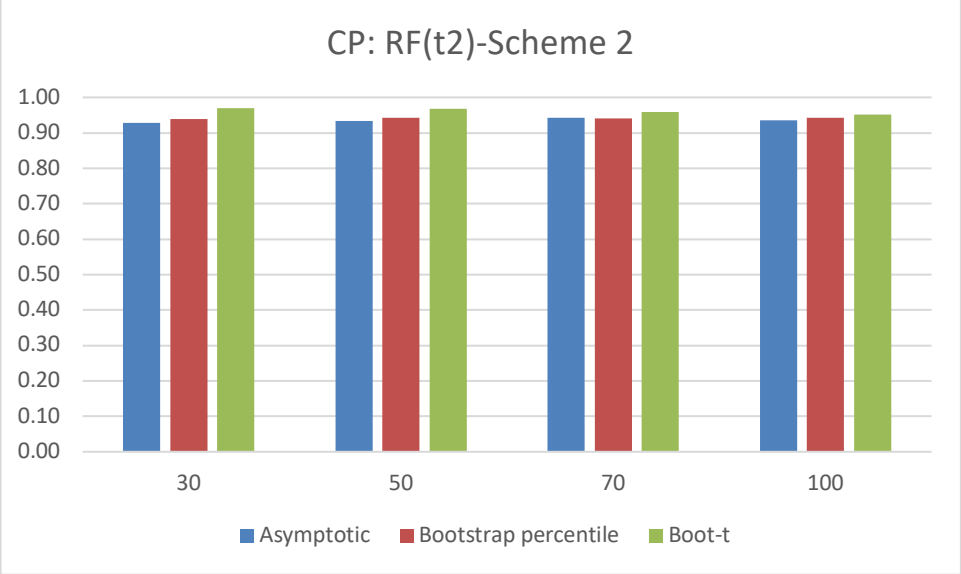

EL: RF(t4)-Scheme 3
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

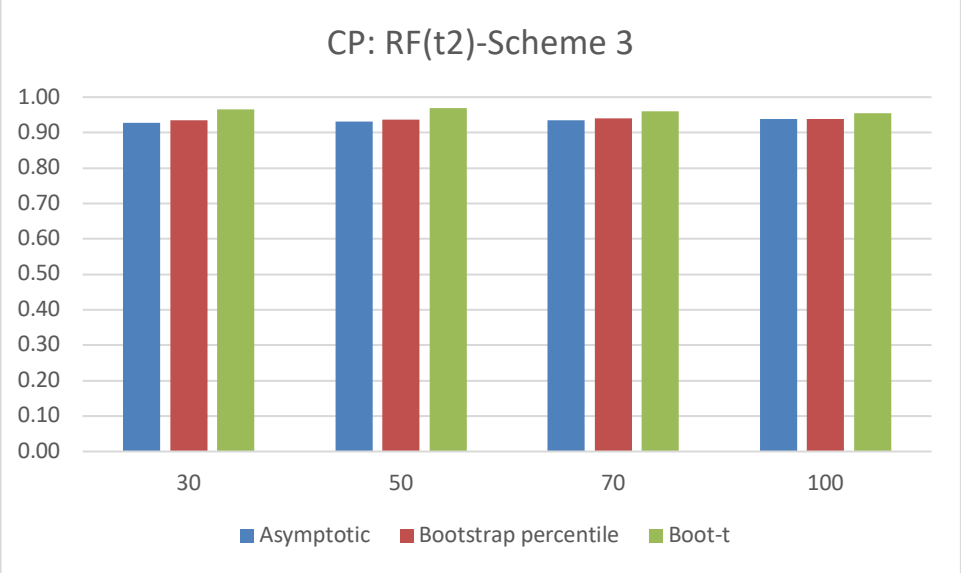

CP: RF(t1)-Scheme 1
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

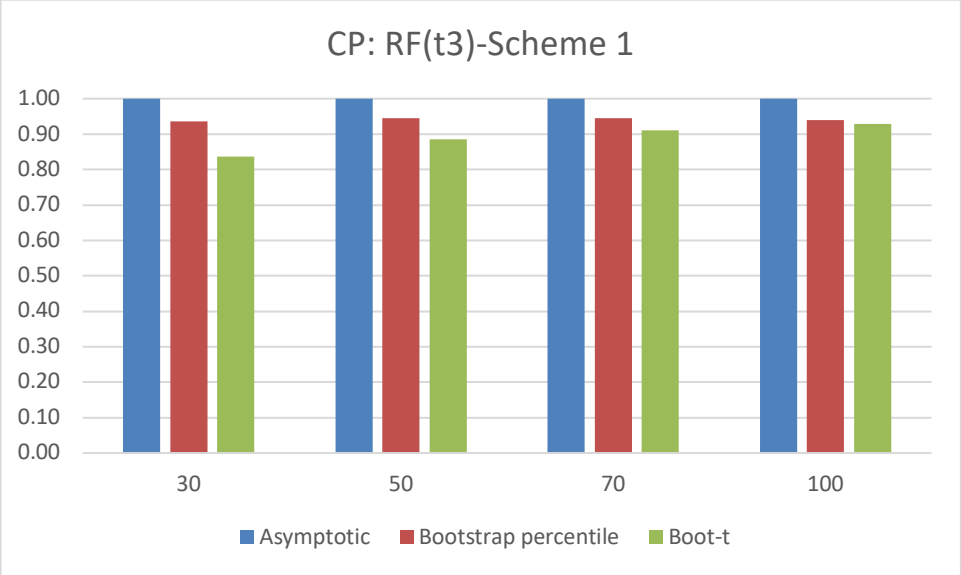

CP: RF(t1)-Scheme 2
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

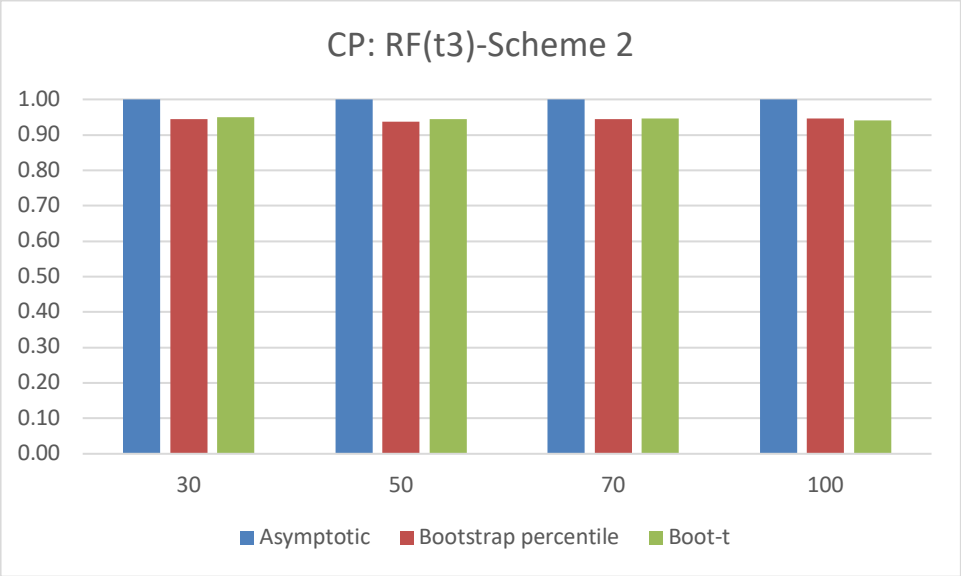

CP: RF(t1)-Scheme 3
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

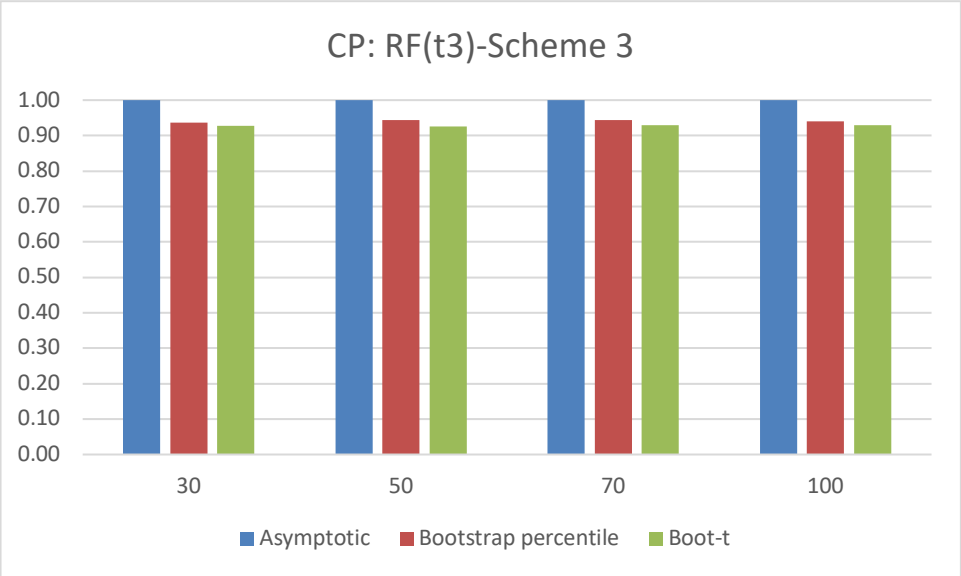

CP: RF(t2)-Scheme 1
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

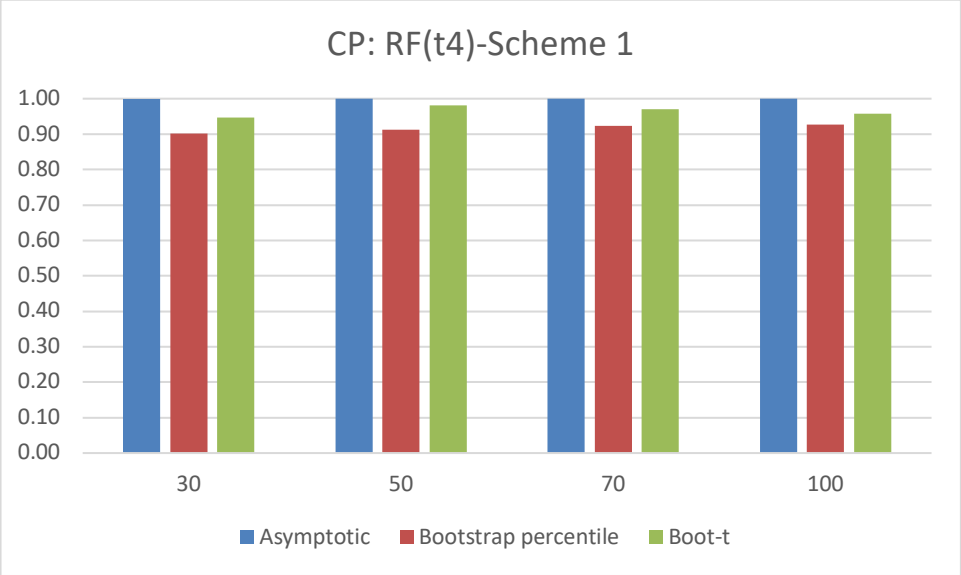

CP: RF(t2)-Scheme 2
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

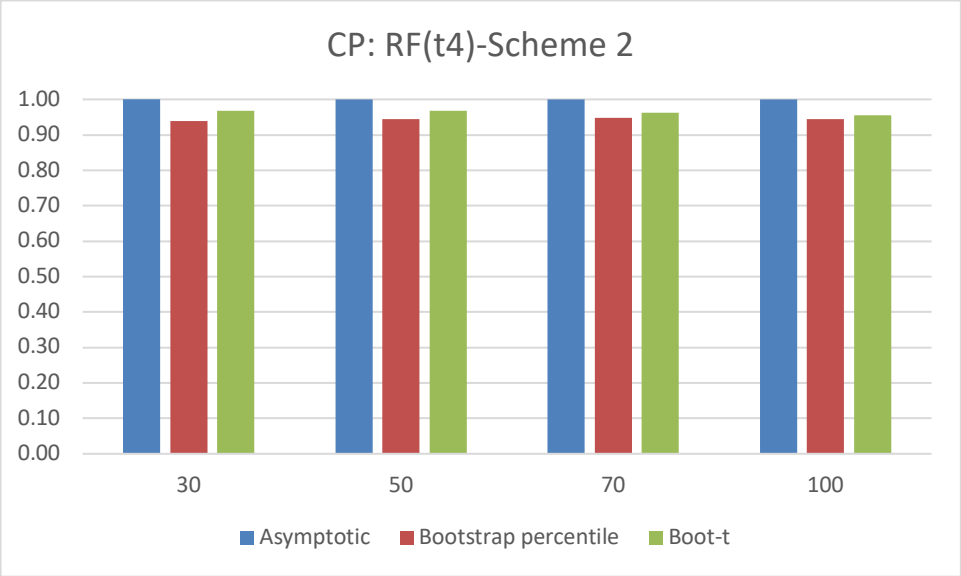

CP: RF(t2)-Scheme 3
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

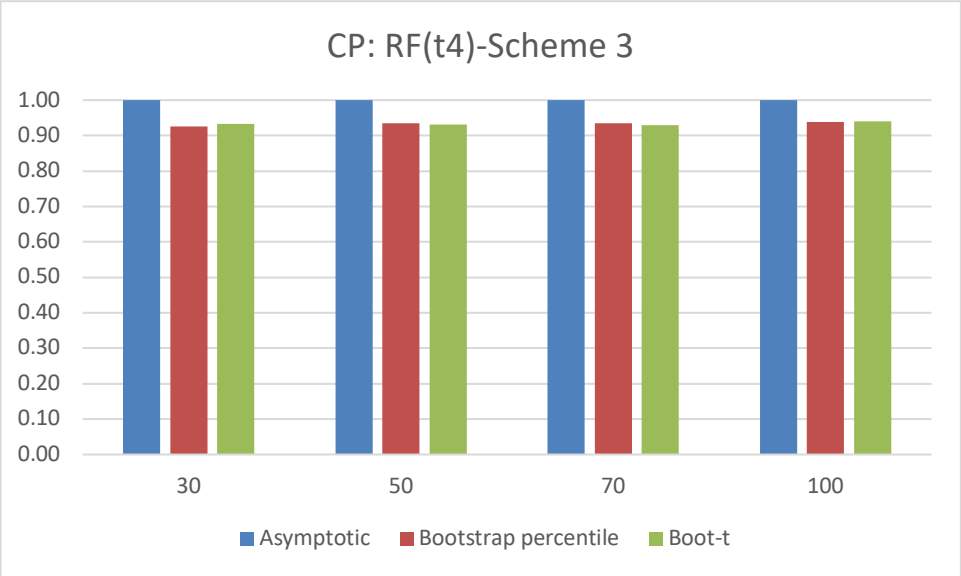

CP: RF(t3)-Scheme 1
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t


CP: RF(t3)-Scheme 2
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

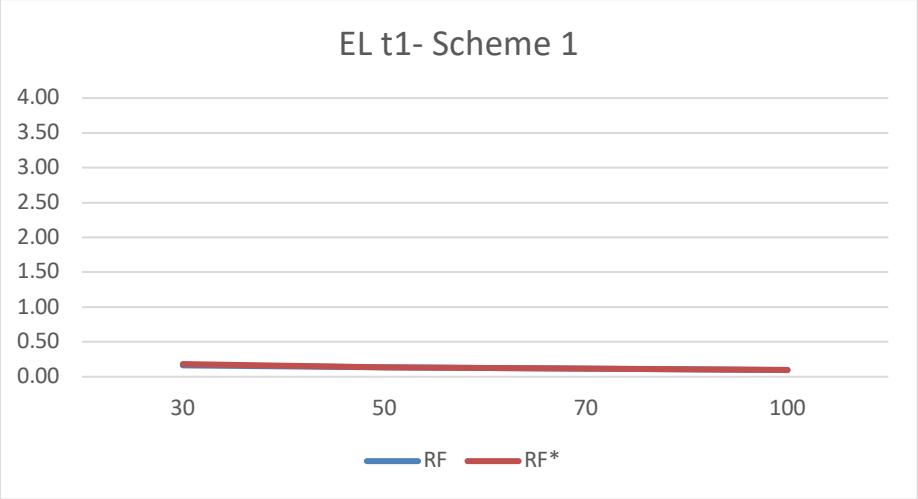

CP: RF(t3)-Scheme 3
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

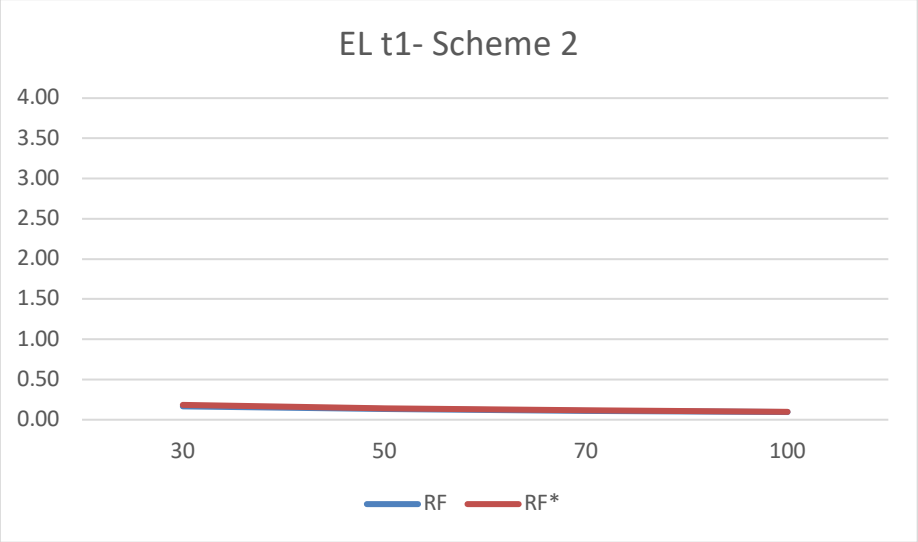

CP: RF(t4)-Scheme 1
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

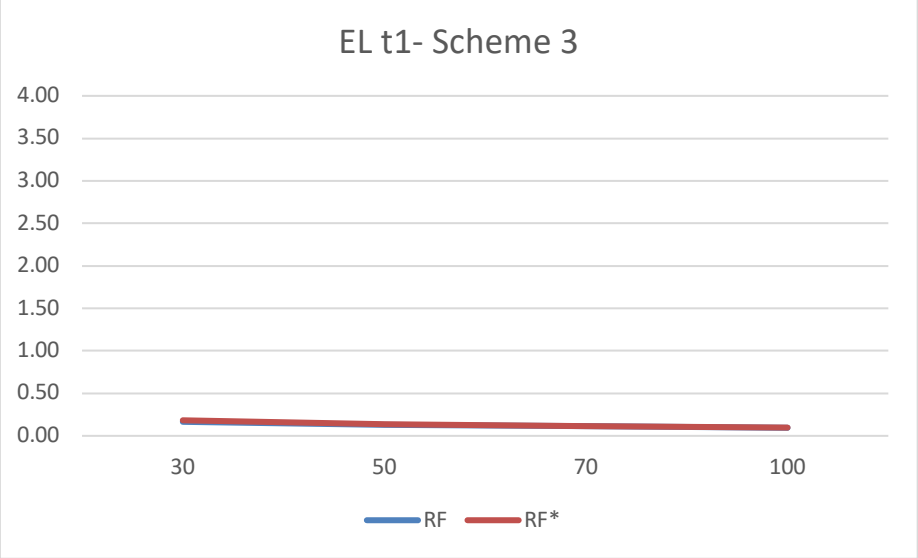

CP: RF(t4)-Scheme 2
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

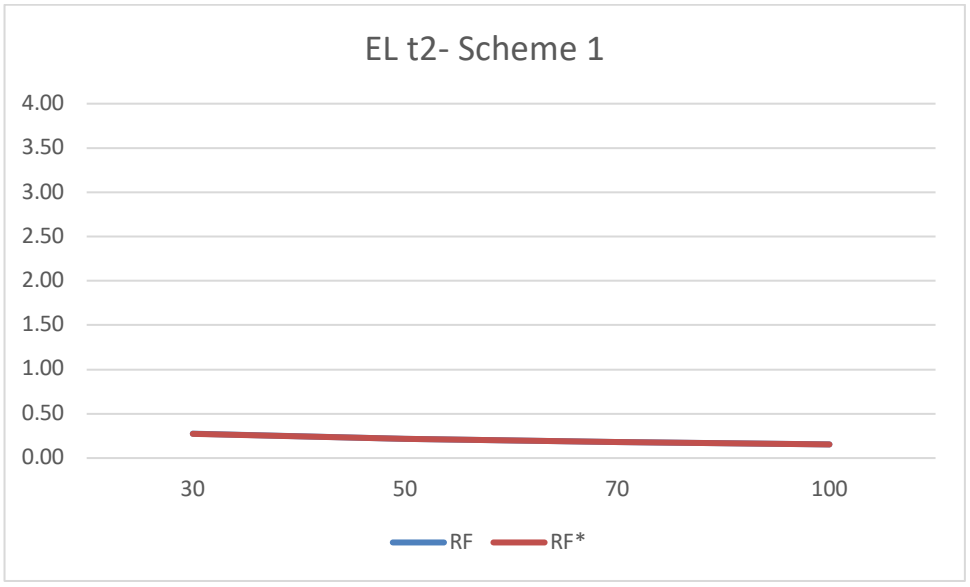

CP: RF(t4)-Scheme 3
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
30
50
70
100
Asymptotic
Bootstrap percentile
Boot-t

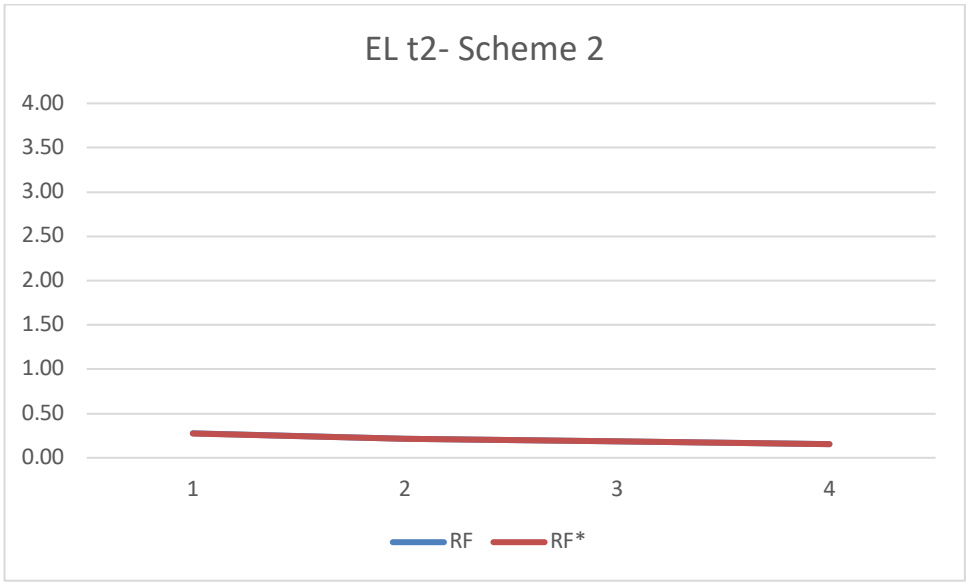
APPENDIX F: BARCHART FOR EXPECTED LENGTH (EL) AND COVERAGE PROBABILITY (CP) FOR RF AND LOG RF.

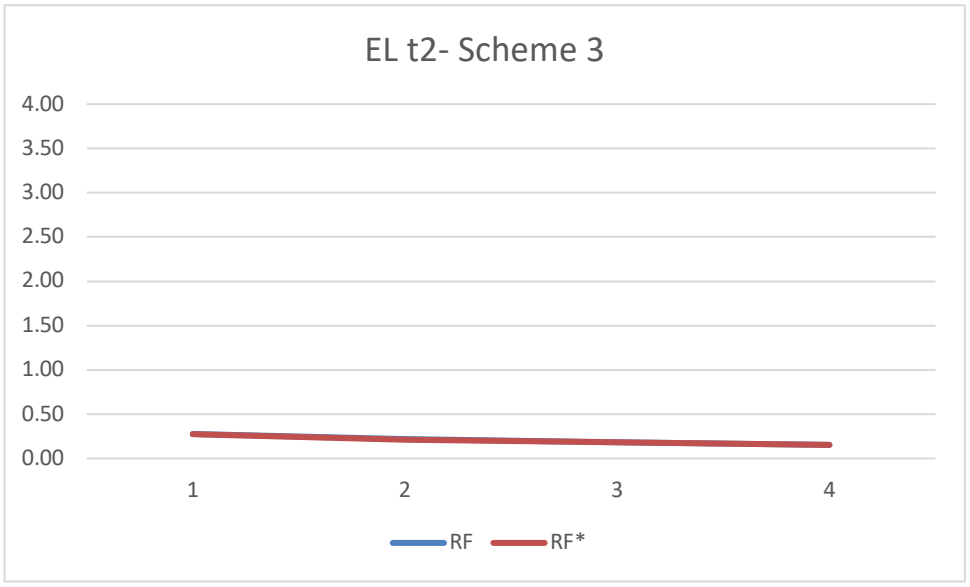


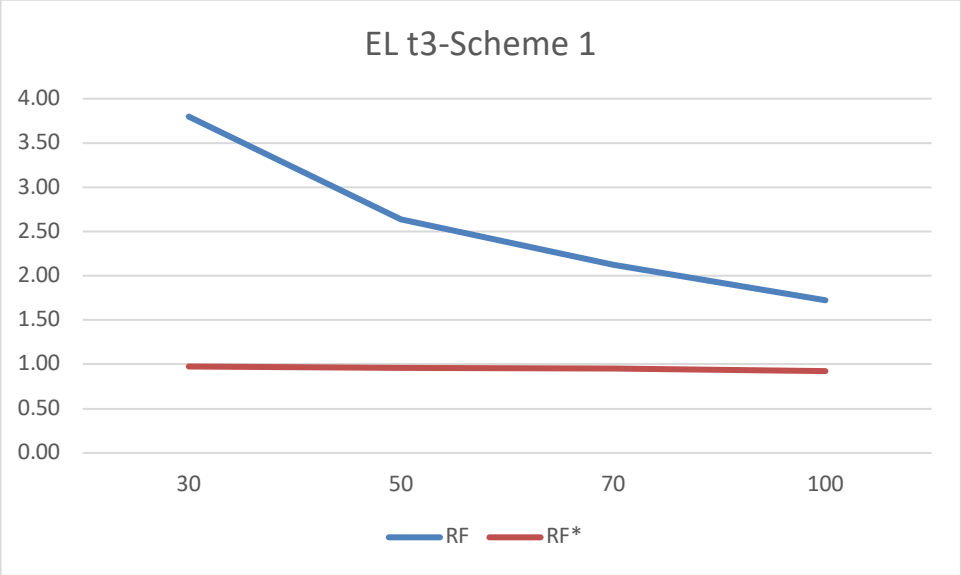


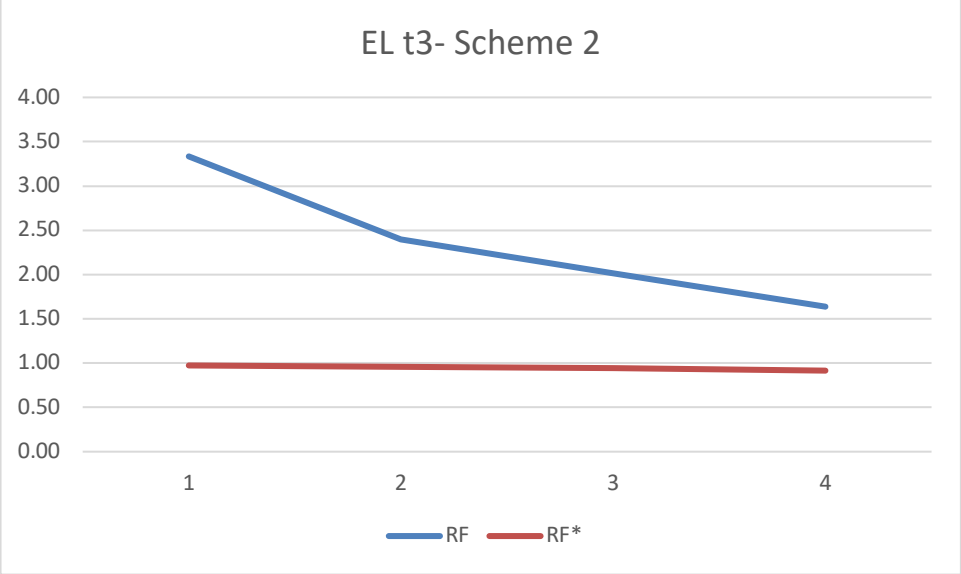


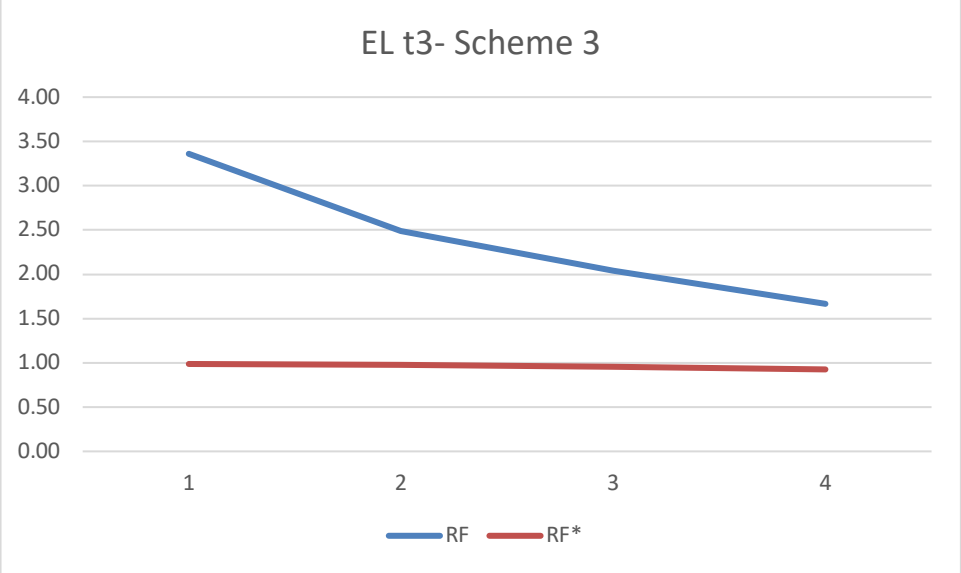

EL t2- Scheme 1
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
30
50
70
100
RF
RF*

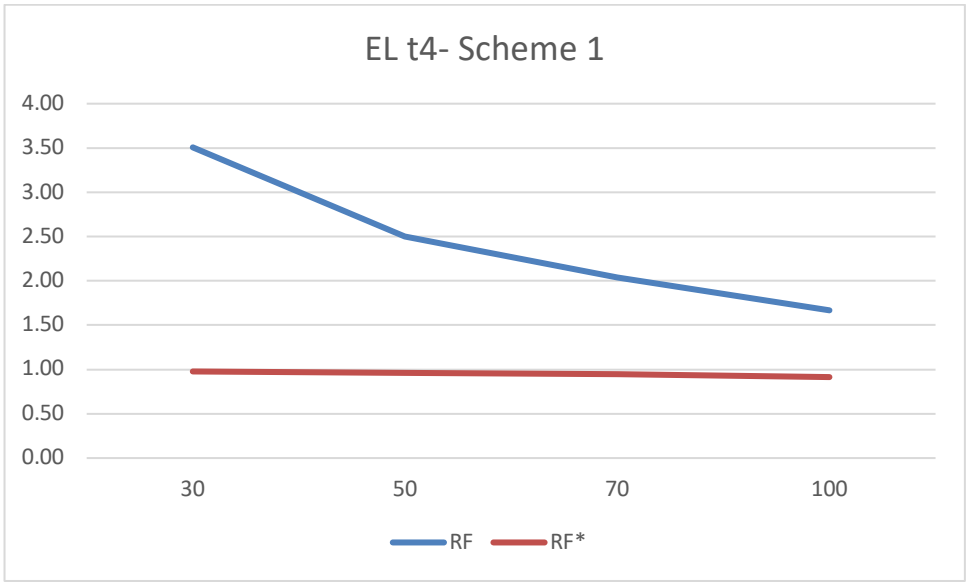

EL t2- Scheme 2
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
1
2
3
4
RF
RF*

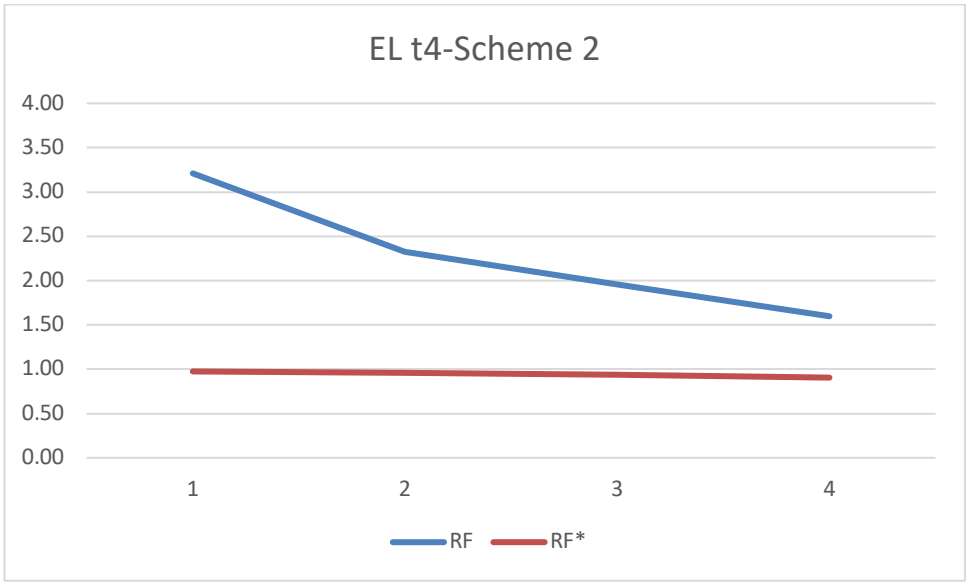

EL t2- Scheme 3
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
1
2
3
4
RF
RF*

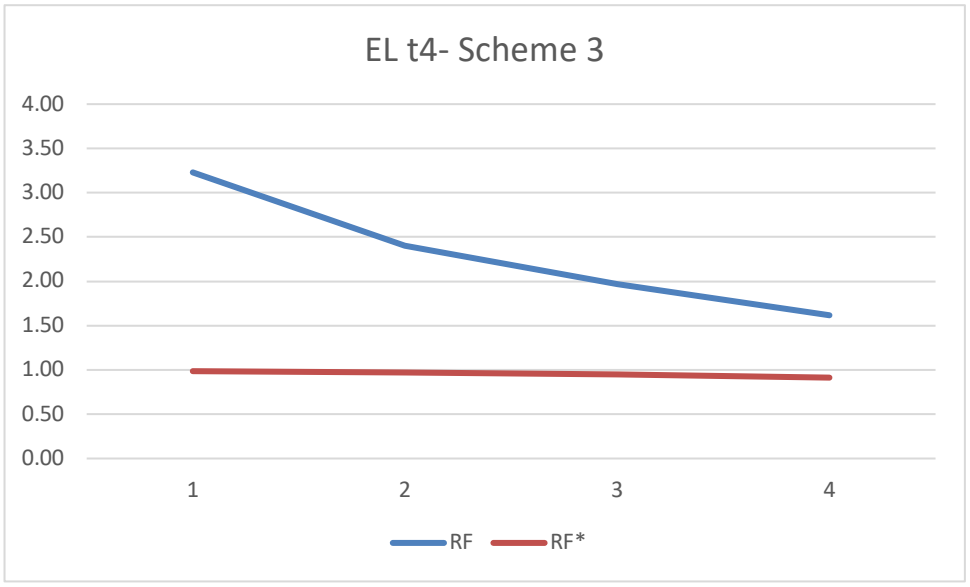

EL t3-Scheme 1
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
30
50
70
100
RF
RF*

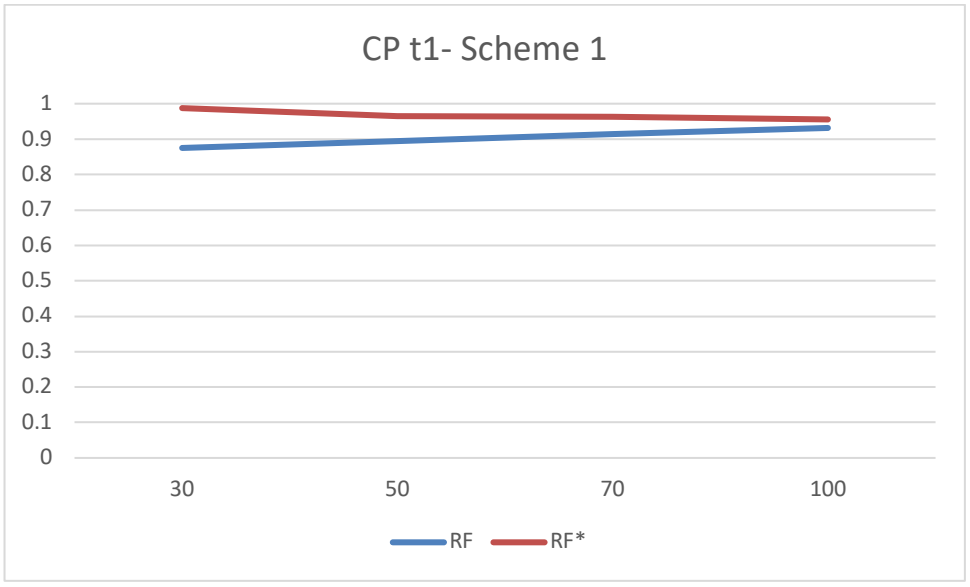

EL t3- Scheme 2
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
1
2
3
4
RF
RF*

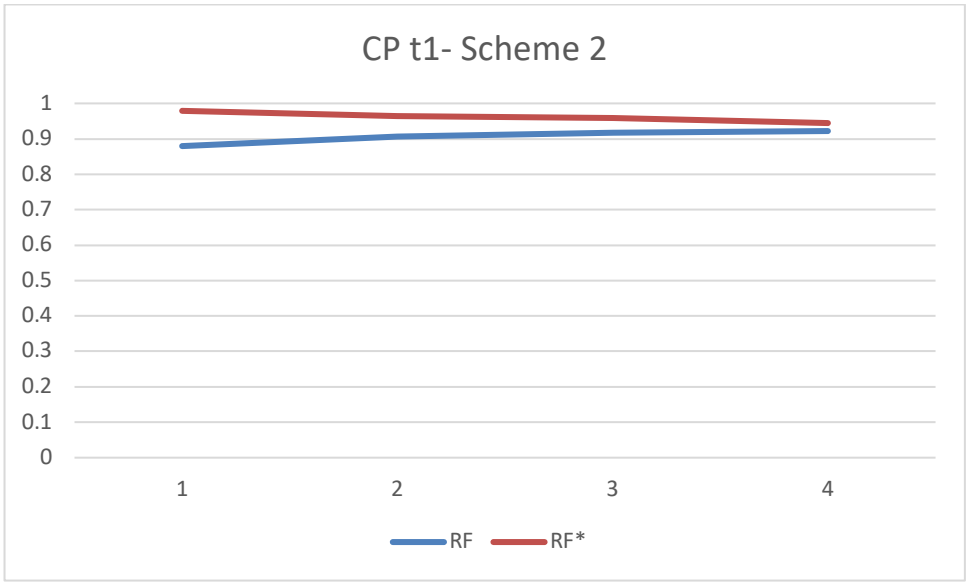

EL t3- Scheme 3
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
1
2
3
4
RF
RF*

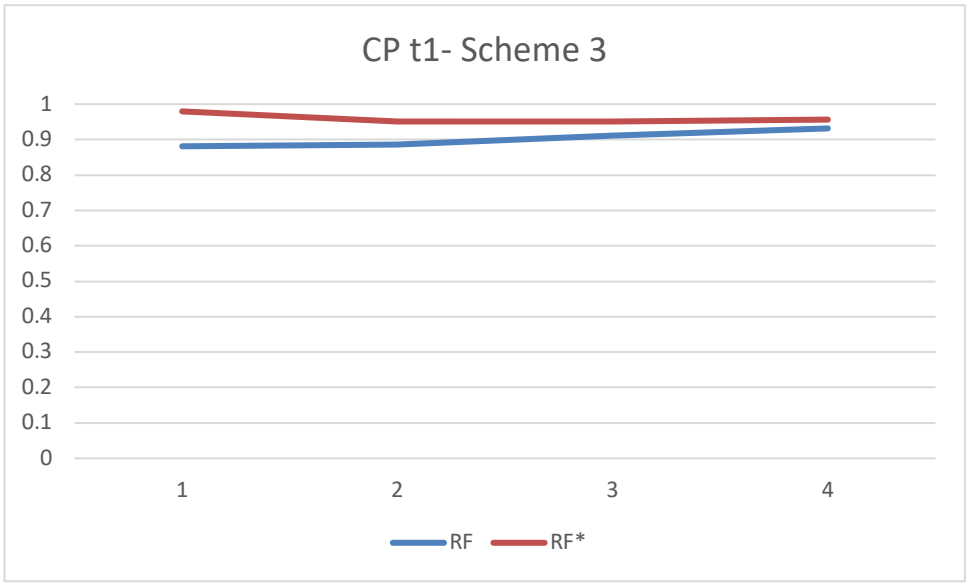

EL t4- Scheme 1
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
30
50
70
100
RF
RF*

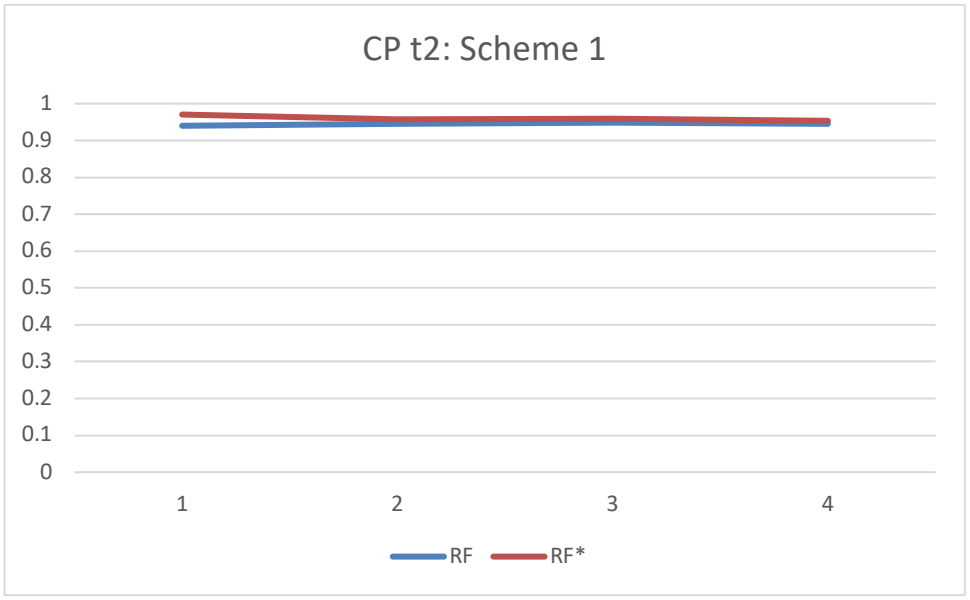

EL t4-Scheme 2
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
1
2
3
4
RF
RF*

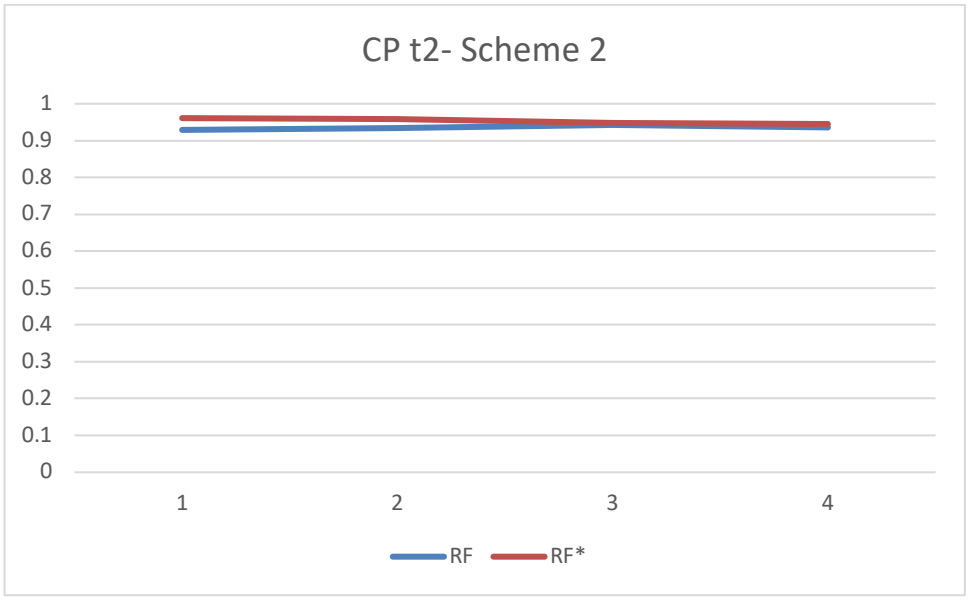

EL t4- Scheme 3
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
1
2
3
4
RF
RF*

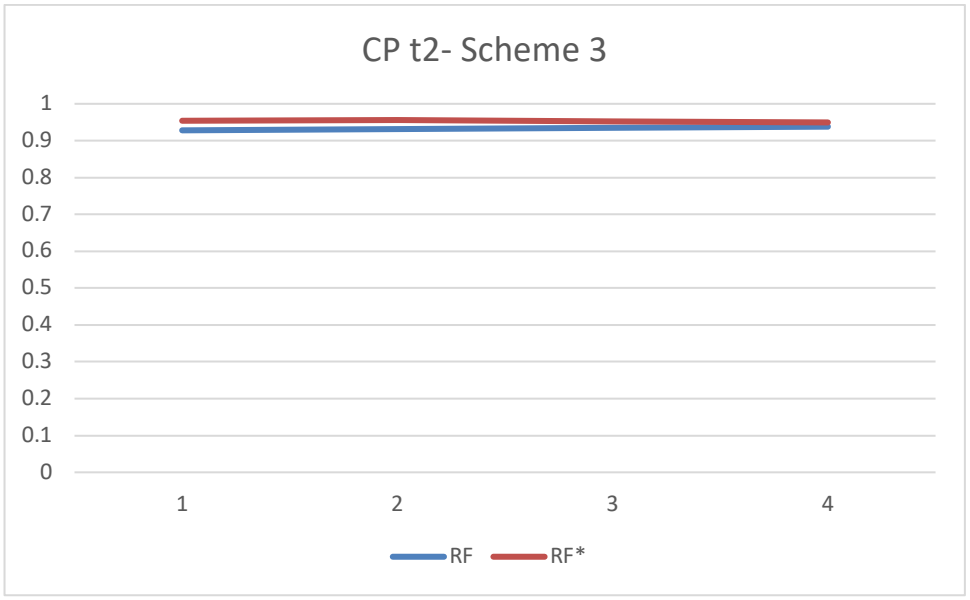

CP t1- Scheme 1
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
30
50
70
100
RF
RF*

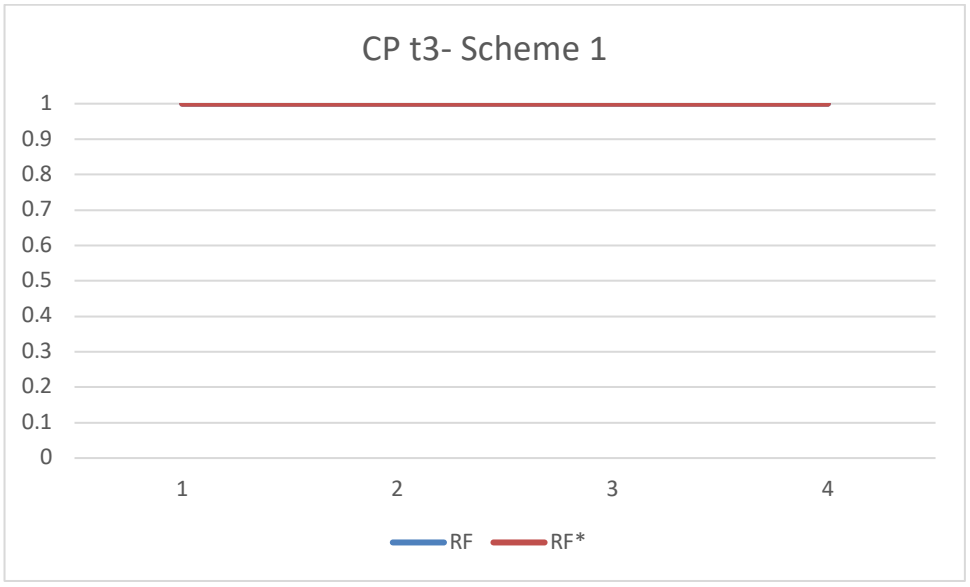

CP t1- Scheme 2
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
RF
RF*

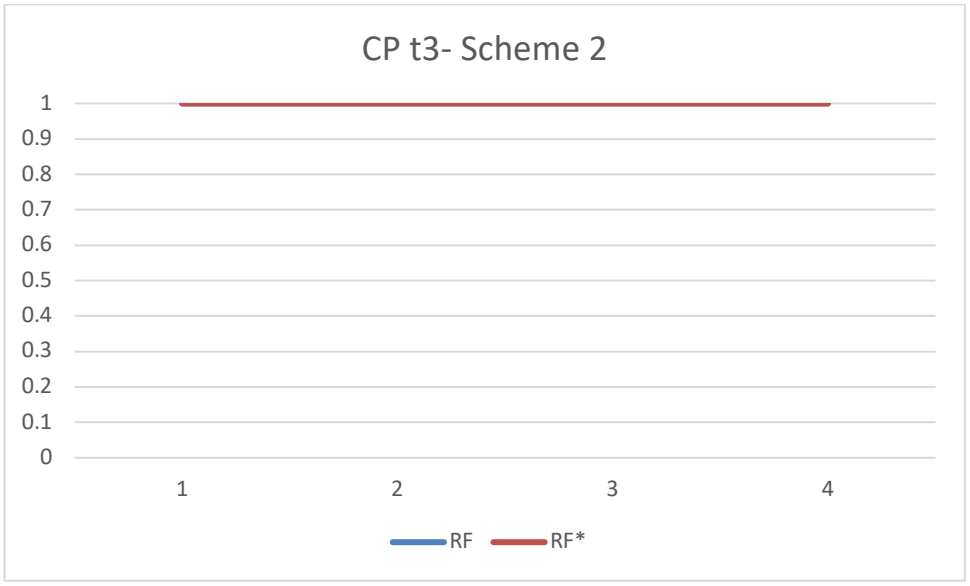

CP t1- Scheme 3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
RF
RF*

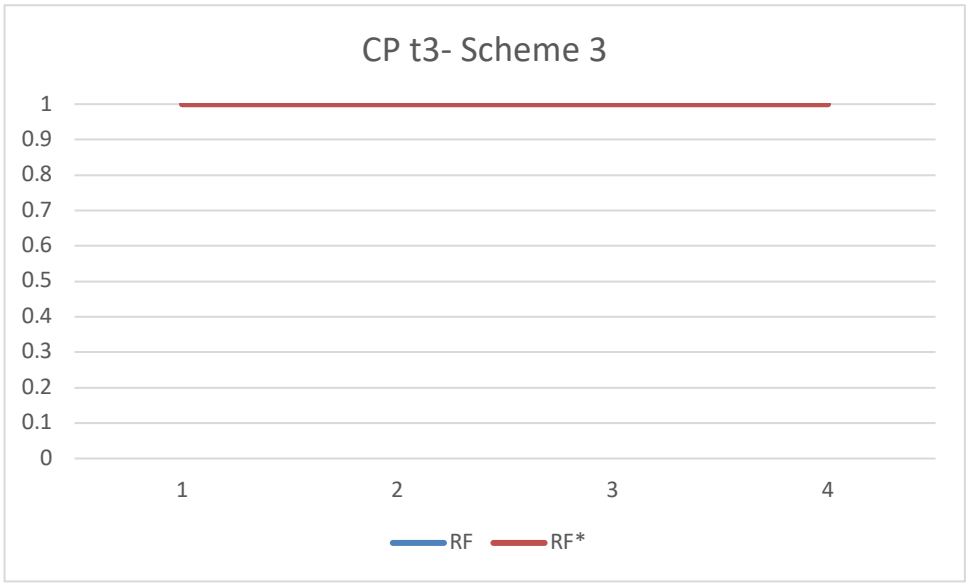

CP t2: Scheme 1
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
RF
RF*

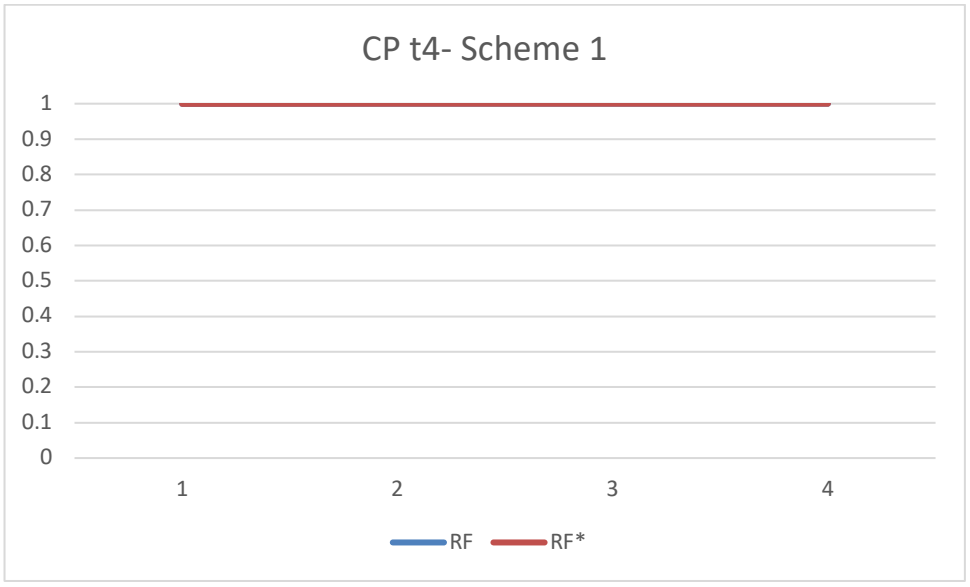

CP t2- Scheme 2
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
RF
RF*

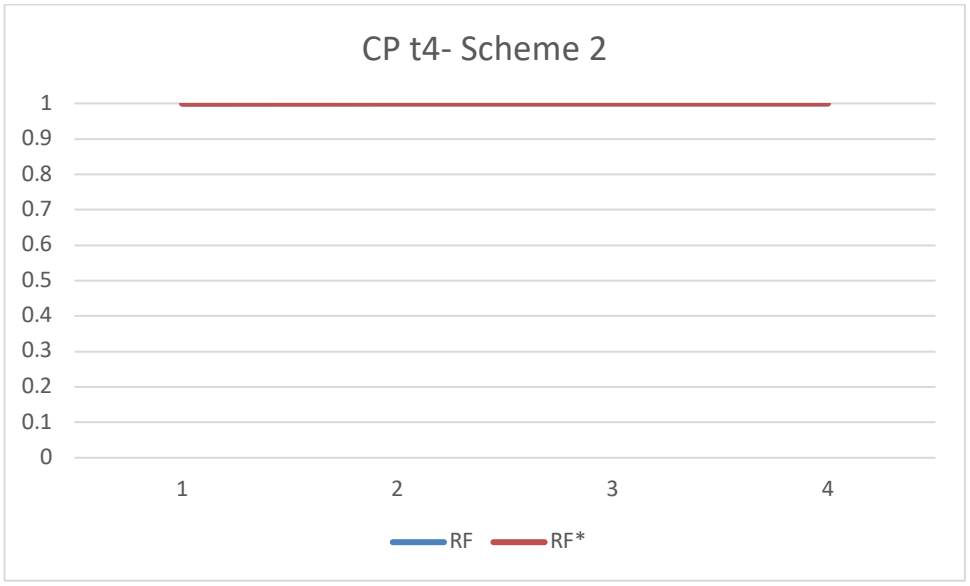

CP t2- Scheme 3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
RF
RF*

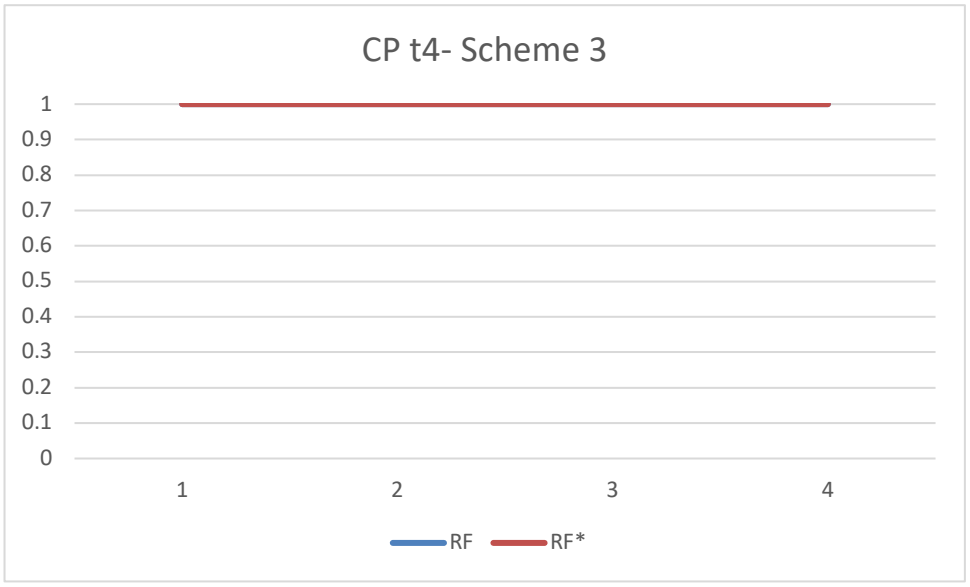

CP t3- Scheme 1
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
RF
RF*


CP t3- Scheme 2
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
RF
RF*


CP t3- Scheme 3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
RF
RF*


CP t4- Scheme 1
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
RF
RF*


CP t4- Scheme 2
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
RF
RF*


CP t4- Scheme 3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
RF
RF*

APPENDIX D: R CODE FOR DATA ANALYSIS EXAMPLES

```
nr=1; Bnr=1000; n=50; r=0.5*n; tau=0.4567534; eta=0.6961452
lmd=1; tht=1.2; beta=2; m=0; alpha=0.05
#Sample1
#t=0.1587378
#t=0.3783076
#t=0.610000
#t=0.601
# Creating Empty Vectors/Matrix
#Parameter
P.All=matrix(NA,nrow=nr,ncol=3)  #to save all parameter estimators before Bootstrap
colnames(P.All)=c("Est.lambda","Est.Theta","Est.Beta")
Var.P.All=matrix(NA,nrow=nr,ncol=3)
colnames(Var.P.All)=c("Var.lambda","Var.Theta","Var.Beta")
C.I.P.All=matrix(NA,ncol=6,nrow=nr)
colnames(C.I.P.All)=c("lower.Lambda","Upper.Lambda","Lower.Theta","Upper.Thet
a","Lower.Beta","Upper.Beta")
#Reliability Function Main loop
RF.All=c()
Var.RF.All=c()
d.l1=c()
d.t1=c()
d.l2=c()
d.t2=c()
```

```
d.b2=c()
#Expected length
Ex.Lambda=c()
Ex.Theta=c()
Ex.Beta=c()
Ex.R.F=c()
Ex.T.R.F=c()
#Coverage probability
C.Prob.L= c()
C.Prob.T=c()
C.Prob.B=c()
C.Prob.RF=c()
C.Prob.T.RF=c()
#Bootstrap
B=matrix(NA,nrow=Bnr,ncol=3)
colnames(B)=c("B.lambda","B.Theta","B.Beta")
Bz=c()
BR.F=c()
Bcov=matrix(NA,nrow=Bnr,ncol=3)
colnames(Bcov)=c("BVar.lambda","BVar.Theta","BVar.Beta")
#Bootstrap-t
Boot.t=matrix(NA,nrow=nr,ncol=8)   # for Boot T for parameters
colnames(Boot.t)=c("lower.BTLambda","Upper.BTLambda","Lower.BTTheta","Upp
er.BTTheta","Lower.BTBeta","Upper.BTBeta","LowerR.F","Upper.R.F")
```

```
Zstar=matrix(NA,ncol=3,nrow=Bnr)   #to save values of parameters Z* Bootstrap t
#Bootstrap Reliability
BRF=c()
BVarR.F=matrix(NA,nrow=nr,ncol=1)  #to save values of variance of R.F
Bd.l1=c()
Bd.t1=c()
Bd.l2=c()
Bd.t2=c()
Bd.b2=c()
Boot.t.RF.L=c()                #R.F lower bound Boot-t
Boot.t.RF.U=c()                #R.F upper bound Boot-t
ZRFstar=c()
#Bootstrap-p
Boot.P=matrix(NA,nrow=nr,ncol=8)
colnames(Boot.P)=c("lower.BLambda","Upper.BLambda","Lower.BTheta","Upper.B
Theta","Lower.BBeta","Upper.BBeta","LowerR.F","Upper.R.F")
#Boothstrap Expected length
P.E.L.L=c()
P.E.L.T=c()
P.E.L.B=c()
P.E.L.R=c()
T.E.L.L=c()
T.E.L.T=c()
T.E.L.B=c()
T.E.L.R=c()
```

```
#Start of Main loop
for (jj in 1:nr){
  U=runif(n)
  US=sort(U) #Sort the uniform random numbers
  F1= function(x) (1-exp(-x/lmd))^tht  # function for 1-exp((-tau)/lmd))^tht
  F2=function(x) (1-exp(-(tau+beta*(x-tau))/lmd))^tht
  U1=US[US<=F1(tau)] #U1 consists of the uniform random numbers that are <= F(tau)
  U2=US[US>F1(tau)] #U2 consists of the uniform random numbers that are  >F(tau)
and are <= F(eta)
  x1=c()
  xx1=c()
  x2=c()
  y=c()
  xx1 = -lmd * log(1-U1^(1/tht))
  nnu=length(xx1)
  nu=min(nnu,r)
  x1=xx1[xx1<=xx1[nu]]
  y= tau+(-lmd*log(1-U2^(1/tht))- tau)/beta  #Use U2 to generate observations from f_2
(x), find their number (n_a)
  # print(y)
  if (nu<r){x2=y[y<=min(y[r-nu],eta)]} else {print ("Number exceeding or equal to pre-
determined failures")}
  Sample=c(x1,y)
  x=c(x1,x2)
  nu=length(x1)
```

```
  na=length(x2)
  p=c(1,1.2,2)
  if (na>0 & nu>0){
    ## a[1]= Lambda, a[2]=Theta, a[3]=Beta
    L1=function(a){-(
      nu*log(a[2]/a[1])
      - sum(x1)/a[1]
      +(a[2]-1)*sum(log(1-exp(-x1/a[1])))
      +(r-nu)*log(a[3]*a[2]/a[1])
      -(1/a[1])*sum(tau+a[3]*(x2-tau))
      +(a[2]-1)*sum(log(1-exp((-1/a[1])*(tau+a[3]*(x2-tau)))))
      +(n-r-m)*log(1-(1-exp((-1/a[1])*(tau+a[3]*(x[r]-tau))))^a[2]))}
    L2=function(a){-(
      nu*log(a[2]/a[1])
      - sum(x1)/a[1]
      + (a[2]-1)*sum(log(1-exp(-x1/a[1])))
      +na*log(a[3]*a[2]/a[1])
      -(1/a[1])*sum(tau+a[3]*(x2-tau))
      +(a[2]-1)*sum(log(1-exp((-1/a[1])*(tau+a[3]*(x2-tau)))))
      +(n-(nu+na)-m)*log(1-(1-exp((-1/a[1])*(tau+a[3]*(eta-tau))))^a[2]))}
    try({if (nu+na==r){
      res=optim(p,L1,method="L-BFGS-B",lower=c(0.1,0.1,0.3),upper =
c(5,5,6),hessian = TRUE)
      z=res$par
      CovMatrix=solve(res$hessian)
```

```
   } else
   {
     res=optim(p,L2,method="L-BFGS-B",lower=c(0.1,0.1,0.3),upper =
c(5,5,6),hessian = TRUE)
     z=res$par
     CovMatrix=solve(res$hessian)
   }},silent=T)
  } #end of if
  P.All[jj,1]=z[1]   #lambda
  P.All[jj,2]=z[2]    #theta
  P.All[jj,3]=z[3]     #beta
  Var.P.All[jj,1]= CovMatrix[1,1]
  Var.P.All[jj,2]= CovMatrix[2,2]
  Var.P.All[jj,3]= CovMatrix[3,3]
  C.I.P.All[jj,1]= z[1] - qnorm (1 - (alpha/2)) *sqrt(CovMatrix[1,1])
  C.I.P.All[jj,2]= z[1] + qnorm (1 - (alpha/2)) *sqrt(CovMatrix[1,1])
  C.I.P.All[jj,3]= z[2] - qnorm (1 - (alpha/2)) *sqrt(CovMatrix[2,2])
  C.I.P.All[jj,4]= z[2] + qnorm (1 - (alpha/2)) *sqrt(CovMatrix[2,2])
  C.I.P.All[jj,5]= z[3] - qnorm (1 - (alpha/2)) *sqrt(CovMatrix[3,3])
  C.I.P.All[jj,6]= z[3] + qnorm (1 - (alpha/2)) *sqrt(CovMatrix[3,3])
  # True parameter R.F
  p=c(1,1.2,2)
  if (t < tau) {
    P.R.F=1-(1-exp(-t/p[1]))^p[2]
  } else if
```

```
(t < eta)
{P.R.F= 1-(1-exp((-1/p[1])*(tau+(p[3]*(t-tau)))))^p[2]
} else if
(t>eta)
{ #print("Invalid:t greater than eta")
  P.R.F="Invalid:t greater than eta"
}
# Reliability Function (Main loop)
if (t < tau) {
  RF=round(1-(1-exp(-t/z[1]))^z[2],6)
} else if
(t < eta)
{ RF= round(1-(1-exp((-1/z[1])*(tau+(z[3]*(t-tau)))))^z[2],6)
} else if
(t > eta)
  RF="Invalid:t is greater than eta"
RF.All[jj]=RF
c4=RF.All-P.R.F
#Variance of R.F (Main loop)
if (t<tau){
  d.l1= (z[2]*t*(1-exp(-t/z[1]))^z[2])/(((z[1])^2)*(exp(t/z[1]) -1)) # R.F1 Bootstrap
derv of lambada
  d.t1=-((1-exp(-t/z[1]))^z[2]) * (log(1-exp(-t/z[1]))) # R.F1 Bootstrap derv of theta
  gradient.T=matrix(c(d.l1,d.t1),ncol=2,byrow=F)
```

```
    Var.RF=(gradient.T)%*% CovMatrix[1:2,1:2] %*% t(gradient.T)
  } else if (t<eta){
    d.l2=(1/(z[1])^2)*(z[2]*(z[3]*((t-tau)+tau)))*((1-exp((-1/z[1])*(z[3]*(t-
tau)+tau)))^(z[2]-1))*(exp((-1/z[1])*z[3]*(t-tau)+tau)) #derivative of Reliability
function 2 from main loop with respect to lambda
    d.t2=-1*((1-exp((-1/z[1])*((z[3]*(t-tau))+tau)))^(z[2]))*log(1-exp((-
1/z[1])*((z[3]*(t-tau))+tau))) ##derivative of Reliability function 2 from main loop
with respect to theta
    d.b2=-1/(z[1]*(exp((1/z[1])*((t-tau)*z[3])+tau)-1))*(z[2]*(t-tau))*(1-exp((-
1/z[1])*((z[3]*(t-tau))+tau)))^(z[2]) ##derivative of Reliability 2 function  from main
loop with respect to beta
    gradient.T=matrix(c(d.l2,d.t2,d.b2),ncol=3,byrow=F)
    Var.RF=(gradient.T)%*% CovMatrix %*% t(gradient.T)
  }else if (t>eta){
    c="Invalid:t is greater than eta"
  }
  Var.RF.All[jj]=Var.RF
  #Start of Bootstrap
  for (i in 1:Bnr) {
    BU=runif(n)
    BUS=sort(BU) #Sort the uniform random numbers
    BF1= function(x) (1-exp(-x/z[1]))^z[2]  # function for 1-exp((-tau)/lmd))^tht
    BF2=function(x) (1-exp(-(tau+z[3]*(x-tau))/z[1]))^z[2]
    BU1=BUS[BUS<=BF1(tau)] #U1 consists of the uniform random numbers that are
<= F(tau)
```

```
    BU2=BUS[BUS>BF1(tau)] #U2 consists of the uniform random numbers that are
>F(tau) and are <= F(eta)
    Bx1=c()
    Bxx1=c()
    Bx2=c()
    By=c()
    Bxx1 = -z[1] * log(1-BU1^(1/z[2]))
    Bnnu=length(Bxx1)
    Bnu=min(Bnnu,r)
    Bx1=Bxx1[Bxx1<=Bxx1[nu]]
    Bx1 = -z[1] * log(1-BU1^(1/z[2]))   #
    Bnu=length(Bx1)
    By= tau+(-z[1]*log(1-BU2^(1/z[2]))- tau)/z[3]  #Use BU2 to generate observations
from f_2 (x), find their number (n_a)
    # print(By)
    if (Bnu<r){Bx2=By[By<=min(By[r-Bnu],eta)]} else {print ("Number exceeding or
equal to pre-determined failures")}
    #print(Bx1)
    #print(Bx2)
    Bx=c(Bx1,Bx2)
    Bna=length(Bx2)
    By= tau+(-z[1]*log(1-BU2^(1/z[2]))- tau)/z[3] #Use U2 to generate observations
from f_2 (x), find their number (n_a)
    # print(By)
    if (Bnu<r){Bx2=By[By<=min(By[r-Bnu],eta)]} else {print ("Number exceeding or
```

```
equal to pre-determined failures")}
   Bx=c(Bx1,Bx2)
   Bnu=length(Bx1)
   Bna=length(Bx2)
   if (Bna>0 & Bnu>0){
    ## a[1]= Lambda, a[2]=Theta, a[3]=Beta
    BL1=function(a){-(
     Bnu*log(a[2]/a[1])
     - sum(Bx1)/a[1]
     +(a[2]-1)*sum(log(1-exp(-Bx1/a[1])))
     +(r-Bnu)*log(a[3]*a[2]/a[1])
     -(1/a[1])*sum(tau+a[3]*(Bx2-tau))
     +(a[2]-1)*sum(log(1-exp((-1/a[1])*(tau+a[3]*(Bx2-tau)))))
     +(n-r-m)*log(1-(1-exp((-1/a[1])*(tau+a[3]*(Bx[r]-tau))))^a[2]))}
    BL2=function(a){-(
     Bnu*log(a[2]/a[1])
     - sum(Bx1)/a[1]
     + (a[2]-1)*sum(log(1-exp(-Bx1/a[1])))
     +Bna*log(a[3]*a[2]/a[1])
     -(1/a[1])*sum(tau+a[3]*(Bx2-tau))
     +(a[2]-1)*sum(log(1-exp((-1/a[1])*(tau+a[3]*(Bx2-tau)))))
     +(n-(Bnu+Bna)-m)*log(1-(1-exp((-1/a[1])*(tau+a[3]*(eta-tau))))^a[2]))}
    try({if (Bnu+Bna==r){
     Bres=optim(p,BL1,method="L-BFGS-B",lower=c(0.1,0.1,0.3),upper =
c(5,5,6),hessian = TRUE)
```

```
      Bz=Bres$par
      BCovMatrix=solve(Bres$hessian)
    } else
    {
      Bres=optim(p,BL2,method="L-BFGS-B",lower=c(0.1,0.1,0.3),upper =
c(5,5,6),hessian = TRUE)
      Bz=Bres$par
      BCovMatrix=solve(Bres$hessian)
    }},silent=T)
  }
  B[i,1]=Bz[1]
  B[i,2]=Bz[2]
  B[i,3]=Bz[3]
  Bcov[i,1]=BCovMatrix[1,1]
  Bcov[i,2]=BCovMatrix[2,2]
  Bcov[i,3]=BCovMatrix[3,3]
  # Reliability function Bootstrap
  if (t<tau){
 Bd.l1= (Bz[2]*t*(1-exp(-t/Bz[1]))^Bz[2])/(((Bz[1])^2)*(exp(t/Bz[1]) -1))    # R.F1
Bootstrap derv of lambada
    Bd.t1=-((1-exp(-t/Bz[1]))^Bz[2]) * (log(1-exp(-t/Bz[1]))) # R.F1 Bootstrap derv of
theta
    gradient=matrix(c(Bd.l1,Bd.t1),ncol=2,byrow=F)

    BVar.R.F=round((gradient)%*% BCovMatrix[1:2,1:2] %*% t(gradient),6)
```

```
} else if (t<eta){
  Bd.l2= (1/(Bz[1])^2)*(Bz[2]*(Bz[3]*((t-tau)+tau)))*((1-exp((-1/Bz[1])*(Bz[3]*(t-
tau)+tau)))^(Bz[2]-1))*(exp((-1/Bz[1])*Bz[3]*(t-tau)+tau)) #derivative of  Bootstrap
R.F2 with respect to lambda
  Bd.t2=-1*((1-exp((-1/Bz[1])*((Bz[3]*(t-tau))+tau)))^(Bz[2]))*log(1-exp((-
1/Bz[1])*((Bz[3]*(t-tau))+tau))) ##derivative of  Bootstrap R.F2 with respect to theta
  Bd.b2=-1/(Bz[1]*(exp((1/Bz[1])*((t-tau)*Bz[3])+tau)-1))*(Bz[2]*(t-tau))*(1-
exp((-1/Bz[1])*((Bz[3]*(t-tau))+tau)))^(Bz[2]) ##derivative of  Bootstrap R.F2 with
respect to beta
  gradient=matrix(c(Bd.l2,Bd.t2,Bd.b2),ncol=3,byrow=F)
  BVar.R.F= (gradient)%*% BCovMatrix %*% t(gradient)
} else if (t>eta){
  BVar.R.F="Invalid:t is greater than eta"
}
if (t < tau) {
  BRF=1-(1-exp(-t/Bz[1]))^Bz[2]
} else if
(t < eta)
{  BRF=  1-(1-exp((-1/Bz[1])*(tau+(Bz[3]*(t-tau)))))^Bz[2]  #using  invariance
property of MLE for accelerated condition
} else if
(t>eta)
  BRF="Invalid:t is greater than eta"
```

```
  BR.F[i]=BRF   #for Bootstrap
  # Boot-t
  if (BCovMatrix[3,3]=="NaN") cat(jj,i,BCovMatrix[3,3],"\n")
  Zstar[i,1]= (Bz[1]-z[1])/sqrt(BCovMatrix[1,1])   # Z star for each Boot
  Zstar[i,2]= (B[2]-z[2])/sqrt(BCovMatrix[2,2])
  Zstar[i,3]= (Bz[3]-z[3])/sqrt(BCovMatrix[3,3])
  #cat("yes",BRF,RF,sqrt(BVar.R.F),"\n")
  ZRFstar[i]=(BRF - RF)/sqrt(as.numeric(BVar.R.F))
}  #end of Boot loop
# Boot-t
zRF= quantile(sort(ZRFstar),probs=c(1-(alpha/2),alpha/2),na.rm=F)
# Boot-t Confidence interval
Boot.t.RF.L[jj]=RF-(zRF[1]*sqrt(as.numeric(Var.RF)))
Boot.t.RF.U[jj]=RF-(zRF[2]*sqrt(as.numeric(Var.RF)))
zL=quantile(sort(Zstar[,1]),probs=c(1-(alpha/2),alpha/2))
zT=quantile(sort(Zstar[,2]),probs=c(1-(alpha/2),alpha/2))
zB=quantile(sort(Zstar[,3]),probs=c(1-(alpha/2),alpha/2))
Boot.t[jj,1]=z[1]-(zL[1]*sqrt(CovMatrix[1,1]))
Boot.t[jj,2]=z[1]-(zL[2]*sqrt(CovMatrix[1,1]))
Boot.t[jj,3]=z[2]-(zL[1]*sqrt(CovMatrix[2,2]))
Boot.t[jj,4]=z[2]-(zL[2]*sqrt(CovMatrix[2,2]))
Boot.t[jj,5]=z[3]-(zL[1]*sqrt(CovMatrix[3,3]))
Boot.t[jj,6]=z[3]-(zL[2]*sqrt(CovMatrix[3,3]))
Boot.t[jj,7]= Boot.t.RF.L[jj]
Boot.t[jj,8]= Boot.t.RF.U[jj]
```

```
 # Boot-p
 low=round((alpha/2)*Bnr,0)
 Up=round((1-(alpha/2))*Bnr,0)
 SortB=apply(B,2,sort)              #Parameter Estimate
 SortBR.F=apply(matrix(BR.F,ncol=1,nrow=Bnr),2,sort)       #Reliability Functions
 Pa.Boot=
matrix(c(SortB[low,1],SortB[low,2],SortB[low,3],SortBR.F[low,1],SortB[Up,1],Sort
B[Up,2],SortB[Up,3],SortBR.F[Up,1]),ncol=2,nrow=4,byrow=FALSE)               #C.I
Bootstrap-Percentile
 colnames(Pa.Boot)=c("Lower Bound", "Upper Bound")
 row.names(Pa.Boot)=c("Lambda","Theta","Beta","Reliability")
 Boot.P[jj,1]=Pa.Boot[1,1]
 Boot.P[jj,2]=Pa.Boot[1,2]
 Boot.P[jj,3]=Pa.Boot[2,1]
 Boot.P[jj,4]=Pa.Boot[2,2]
 Boot.P[jj,5]=Pa.Boot[3,1]
 Boot.P[jj,6]=Pa.Boot[3,2]
 Boot.P[jj,7]=Pa.Boot[4,1]
 Boot.P[jj,8]=Pa.Boot[4,2]
} #End of  main loop
# Log Transformation of Reliability Function (From main loop)
R.Fstar=log(RF.All/(1-RF.All))
d.R.Fstar= -(1/((RF.All-1)*RF.All))  #derivative for log transformation of R.F
Var.Rstar=(d.R.Fstar^2)*Var.RF.All  #variance for log transformation of R.F
```

```
# Asymptotic Confidence Intervals
# Transformed R.F (Log R.F)
l.R.Flog= R.Fstar - qnorm (1 - (alpha/2)) *sqrt(Var.Rstar) #Lower bound
u.R.Flog= R.Fstar + qnorm (1 - (alpha/2)) *sqrt(Var.Rstar) #upper bound
# RF** (Transformed back to R.F)
T.l.R.F=exp(l.R.Flog)/(1+exp(l.R.Flog))  #Lower bound for C.I after transforming back to R.F
T.u.R.F=exp(u.R.Flog)/(1+exp(u.R.Flog))  #upper bound for C.I after transforming back to R.F
# Confidence Width of R.F**
Tciw=T.u.R.F-T.l.R.F
#Confidence interval of Original R.F (without Transformation)
l.R.F=RF.All - qnorm (1 - (alpha/2)) *sqrt(Var.RF.All)   #lower bound of C.I
u.R.F=RF.All + qnorm (1 - (alpha/2)) *sqrt(Var.RF.All)  # upper bound of C.I
# Confidence Width of R.F
ciw=u.R.F-l.R.F
ciwstar=T.u.R.F-T.l.R.F
C.I.RF=matrix(c(l.R.F,u.R.F,ciw,T.l.R.F,T.u.R.F,ciwstar),ncol=6,nrow=nr)
colnames(C.I.RF)=c("Lower.RF","Upper.RF", "C.I Width","Lower.RF**","Upper.RF**","C.I Width**")
#Printing Results
#Test for data coming from Generalized Exponential distribution
ks.test(Sample,F1)
#Details about the sample
```

```
print(list("Sample"=Sample,"Failure times at use condition"=x1,"No of failed items in use condition"=nu,"Failure times at accelerated condition"=x2,"No. of failed items in accelerated condition"=na,"Pre-determined failed items"=r))
# Parameter Estimates, their Variances and Confidence Intervals (MLE)
print(list("All MLE"=P.All,"Variance MLE"=Var.P.All,"Asymptotic C.I MLE"=C.I.P.All))
#Confidence interval
print(list("Ays.CI.RF"=C.I.RF))
print(list("Asymptotic C.I"=C.I.P.All,"Boot.P C.I"=Boot.P,"Boot.t"=Boot.t))
# Reliability Function
print(list("RF"=RF.All,"RF*"=R.Fstar, "Variance RF"=Var.RF.All, "Variance RF*"=Var.Rstar))
```