Transnational University Limburg is a unique collaboration of two universities in two countries: the University of Hasselt and Maastricht University.

UHASSELT
KNOWLEDGE IN ACTION

www.uhasselt.be
Universiteit Hasselt
Campus Hasselt:
Martelarenlaan 42 | 3500 Hasselt
Campus Diepenbeek:
Agoralaan Gebouw D | 3590 Diepenbeek

UHASSELT
KNOWLEDGE IN ACTION

Maastricht University

Faculty of Sciences
School for Information Technology
Master of Statistics and Data Science

Master's thesis

Simulating the study design in industrial studies.

NSOH TANIH DAM
Thesis presented in fulfillment of the requirements for the degree of Master of Statistics and Data Science, specialization Biostatistics

SUPERVISOR :
Prof. dr. Roel BRAEKERS

2020
2021

Maastricht University

Faculty of Sciences
School for Information Technology
Master of Statistics and Data Science

Master's thesis

Simulating the study design in industrial studies.

NSOH TANIH DAM

Thesis presented in fulfillment of the requirements for the degree of Master of Statistics and Data Science, specialization Biostatistics

SUPERVISOR :

Prof. dr. Roel BRAEKERS

Abstract

Background: This study focuses on measuring the uncertainty around the estimate of the influence of temperature on the survival times for technical devices with small sample sizes. Industrial or technical devices are expensive, as a result their failure proves to be destructive and very expensive. Insufficient data arises since engineers cannot wait for large number of devices to fail before measuring the impact of temperature. As a result the estimate of the influence of temperature is sometimes not accurate and has large standard error.

Objectives: The aim of this project is to verify under which conditions the best estimate of the influence of temperature on these technical devices can be obtained. To provide a description of the uncertainty surrounding this estimate under different observational schemes or design. Finally, to analyse the accuracy of this estimate on the survival time in order to provide engineers with the most cost efficient study design.

Methodology: Different designs or censoring schemes were simulated in order to reflect real world experimental scenarios. The event-times were generated from a Weibull distribution since the Weibull distribution is appropriate to describe survival times of industrial machines. For the simulation, sample sizes of 60, 40 and 20 were simulated with varying proportions of censoring. An accelerated failure model is fitted to simulated datasets where 1000 different dataset are simulated for every design case. The estimate of the temperature coefficient is of interest and the uncertainty per scenario and/or sample size is studied. The coverage probability, mean square error and the bias of the estimate is analysed. Furthermore, the maximum likelihood reduced biased adjustment is used to adjust coverage probabilities, mean square error and bias. Also, the estimate values were averaged over a 1000 different samples.

Results: The results obtained showed differences in the estimates gotten from the various scenarios or designs. The interval designs demonstrated the poorest estimation of the temperature effect. The type I design showed the best coverage with the most precised estimation. Other designs followed in the following order; uncensored designs, mixture intervals and type III. These designs had a good coverage of the true population temperature estimate.

Conclusions: The amount of coverage reduced as percentage of censoring increased and sample size reduced. The MSE and bias of the estimate under interval designs were extremely high when compared to others. The type I right censoring design outperformed all other design schemes in terms of lower MSE and better coverage of the true population temperature value.

Key Words: Mean Squre Error(MSE), Reduced Bias adjustments(RBA),Accelerated Failure Time (AFT), Weibull, Simulation, Censoring, Coverage, Bias.

Contents

List of Figures

List of Tables

Acknowledgement

Special thanks to Prof. dr. Roel Braekers for giving me the opportunity and also, for his sufficient support through out the thesis. My deepest gratitude also goes to my family and friends for their never ending support and encouragement. For all this, I am truly grateful. To God Almighty, thank You for granting me the strength and wisdom.

Nsoh Tanih Dam

Hasselt, June 16, 2021

1 Introduction

Temperature remains one of the most important factors influencing the survival or the time to failure of technical devices in industries. As a result, proper methods of estimating its effects and measuring the uncertainty around these estimates across various temperature ranges and different observational schemes is fundamental in ensuring cost effective design and implementation. Every company dealing with technical devices seeks to ensure these devices are reliable by understanding how long it takes for these machines to fail under a certain temperature. Measuring the effects of temperature in a precised and more accurate way will provide engineers with better estimates and precised information with regards to the time to failure of these technical devices. However, waiting for these machines to fail in order to get sufficient data to estimate the effects of temperature takes a very long time. These long duration needed to gather sufficient data poses a challenge to engineers who are in dire need of answers about the reliability and efficiency of these technical devices. A solution to this challenge is to accelerate the time failure of these technical devices under higher temperatures. Engineers perform accelerated test on their products by exposing them to harsher/stress conditions in order to generate failures and useful reliability information more quickly [5].

Simulating the failure time of these technical device with an accelerated testing approach increases the possibility that these machines will wear out and eventually fail earlier than being under normal temperatures. Therefore the aim of applying these accelerated approach is to obtain results within a shorter time frame. However, precised knowledge of theory relating the accelerating variable and the time to failure is not always available and the model is chosen on the basis of previous experience in similar situations. When estimating the influence of temperature on failure times of technical devices, engineers are often faced with challenges such as; insufficient amount of data since these technical devices are expensive and cannot be allowed to undergo several failure events. Secondly different censoring mechanisms are observed while collecting the data. As a result problems of in-precise estimates steps in.

Luis and William [5] in their paper of review of accelerated test models pointed out that information from tests at high levels of one or more accelerating variables (e.g., use rate, temperature, voltage or pressure) is extrapolated, through a physically reasonable statistical model, to obtain estimates of life or long-term performance at lower, normal levels of the accelerating variable(s). Moreover, in order to carry out reliable projections based on failure data from high stress test, the correct acceleration model for the failure mechanism under investigation and the correct life distribution of the device must be correctly chosen or known [7]. The Weibull life distribution is the most widely used distribution to describe the survival time for technical devices and the pecks model in engineering is a widely used model to effect of stresses on the lifetime of technical devices.

Failure of technical devices proves to be destructive and very expensive. Therefore, implementing flexible observation schemes or using various censoring schemes during experiments is paramount [3]. Furthermore, many practical issues such as smaller to moderate sample sizes urges the need to quantify the uncertainty surrounding the estimates gotten [8]. Maximum likelihood methods can be used to fit regression models used in survival analysis and also do covariate adjustments [14]. Obtaining estimates for the Weibull distribution under progressive type-I interval censoring using maximum likelihood method was examine [15]. Inference on reliability estimation with the maximum likelihood estimate of Weibull parameters is conducted for Type-I censored data [19]. The parameters

of a life stress model was estimated for Smart Electricity Meter based upon the life-stress model with a fit to a Weibull distribution and efficient parameters were obtained [2]. Also, in cases of unavailability of analytical results, simulations which mimic actual data generating schemes can be used to supplement and verify the adequacy of finite sample properties[8].

In this project, several simulation studies are carried by generating Weibull distributed survival times of a technical device in which temperature has an influence on the survival times through the Peck's model. Each simulation study has different characteristics such as varying sample size, varying proportions of censoring and interval censoring in order to reflect real world scenarios. Finally for each simulated dataset the influence of the temperature on the survival time is measured and the uncertainty around this estimate is measured and compared.

1.1 Objectives

The aim of this project is to verify under which conditions the best estimate of the influence of temperature on these technical devices can be obtained. To provide a description of the uncertainty surrounding this estimate under different observational schemes. Finally, to analyse the accuracy of this estimate on the survival time in order to provide engineers with the most cost efficient study design.

2 Methodology

2.1 The Weibull Distribution

The Weibull is sufficient in describing observed failures for various types of technical components and scenarios [16]. As a result it is effective in providing proper description for time to survival for technical devices. Another important characteristic of the Weibull distribution is that is has a limiting distribution of minima [8]. The Weibull cdf of a technical component is given as follows;

$$F(t;\mu,\sigma) = 1 - exp\left[-\left(\frac{t}{\beta}\right)^{\alpha}\right] = \Phi_{sev}\left[\frac{log(t) - log(\beta)}{\frac{1}{\alpha}}\right], t > 0$$

where $\Phi_{sev}(z) = 1 - exp[-exp(z)]$, $\alpha > 0$ is the Weibull shape parameter, $\beta > 0$ is the Weibull scale parameter(the 0.632 quantile) and $t \geq 0$ represents time. Also, the relationship between a Weibull random variable and the logarithm of a Weibull distribution random variable has a smallest extreme value distribution. This implies $\mu = log(\beta)$ is the SEV location parameter and $\sigma = 1/\alpha$ is the SEV scale parameter [8].

The SEV is adequate for technical device failures related to stress and can be used to model minimum values. Questions such as what is the minimum temperature needed for a technical device to fail can be answered. Moreover, the hazard function of a SEV distribution is suitable for modeling the survival time of technical devices that can experience very fast wear out (for example the final stages of the bathtub curve).

The hazard rate or failure rate for the Weibull is given by

$$h(t) = \frac{f(t)}{F(t)} = \frac{\alpha}{\beta}\left(\frac{t}{\beta}\right)^{\alpha-1}$$

Different values of the shape parameter have different effects on the Weibull distribution. The slope or shape of the Weibull distribution describes the failure rate of the technical device. A slope value of less than 1 ($\alpha < 1$) indicates the device has decreasing failure rate this mode of failure is usually referred to as infant mortality, where devices fail at the beginning but failure rate decreases over time. If $\alpha = 1$, devices have a constants failure rate and if $\alpha > 1$, devices undergo increasing failure rate. The scale parameter has an effect on the x-axis. A change in the value of the scale parameter while holding α fixed has the effect of either stretching in or stretching out the pdf of the Weibull distribution.This implies the peak of the pdf curve of a Weibull distribution will decrease with an increase in the scale parameter since the pdf will be stretched along the x-axis.

2.2 Pecks model

The Peck's model properly describes the relationship between the accelerated lifetime of a technical device, temperature and/or relative humidity [2].

$$T = A_0\,(RH)^{-N}\,exp^{\left(\frac{-E_a}{Kt}\right)}$$

where;

- T is the lifetime of a technical device.
- A_0 is the shape factor.
- RH is the relative humidity.
- N is a constant (n is between 1 and 12, typically n = 3).
- Ea is the activation energy in electron volts (in the range of 0,3 to 1,5, typically Ea = 0,9).
- k is the Boltzmann constant (8,617 × 10–5 eV/K).
- t is the temperature in K.

In order to evaluate the influence of temperature on the lifetime of a technical device from the equation above the relative humidity (RH) is kept constant such that the T depends only on (t). The model becomes:

$$T = A_0 exp^{\left(\frac{-E_a}{Kt}\right)}$$

$$\beta = E[T] = A_0 exp^{\left(\frac{-E_a}{Kt}\right)} = e^{\alpha_0 + \alpha_1\left(\frac{1}{Kt}\right)}$$

Since, different technical devices are tested at higher values of temperature (stress) levels, the Weibull distribution becomes well suited to describe the time to failure of these devices. Therefore, ($T \sim Weibull(\beta, \alpha)$) and the pdf is given by;

$$f(t) = \exp^{-\left(\frac{t}{\alpha}\right)^{\beta}}$$

where,

$$\beta = e^{\alpha_0 + \alpha_1\left(\frac{1}{Kt}\right)}$$

The time to failure follows a Weibulll distribution with scale parameter $\beta = e^{\alpha_0 + \alpha_1\left(\frac{1}{Kt}\right)}$ and shape parameter (α). Therefore, Weibull distributed eventtimes are generated at different temperatures through the Peck's relationship as shown above.

2.3 Censoring

Generally in experiments involving time to failure for technical devices, not all information regarding the time to failure for all the devices in the experiments are available and as such the resulting data are called censored data. However, this censoring mechanism is effective in managing the cost incurred and also the total time required to gather data on these experimental units. Failure of this experimental units are usually destructive and expensive. Therefore in order to ensure flexibility and capture a real world scenarios, two types of censoring schemes will be considered during this experiments. In this case the right censoring and interval censoring.

2.3.1 Right Censoring

The survival time(T) of an experimental unit is censored if all we know about T is that T is greater than some value c. This implies a technical device is said to be censored if terminated before an event (in this case failure of the technical device) occurs. Generally, censoring maybe classified into three types; Type I, Type II or Type III (random) right censoring.

The Type I censoring refers to the fact that the censoring time is controlled by the investigator [17] and pre-specified. This implies the time which the experiment stops is determined by the investigator. This type of right censoring is commonly used in industrial engineering and medical studies. For example, in an experiment to monitor the effect of temperature on the survival times of technical devices, all devices may start at a specific time and followed until a pre-defined ending time. As a result, any device which survives above the pre-defined time is marked as censored by the investigator.

The type II censoring refers to a case where a total number of technical devices (n) are monitored until a pre-defined fraction have registered an event. The event times are sorted in ascending order such that the investigator is interested only in the first (m) event times. As a result, $n - m$ are considered censored while m are considered observed. For example, in an experiment to monitor the effect of temperature on the survival times of technical devices , the experiment stops when 40 out of 60 technical devices are observed to fail. This implies the remaining 20 devices are marked as censored.

The type III censoring also known as random censoring. Here censoring can occur in a random manner which is not controlled by the investigator or pre-defined. For example, in an experiment to monitor the effect of temperature on the time to failure for technical devices, a technical device may fail as a result of other factors different from temperature or the the technical device may dropout. This scenario is common in the industrial setting. With random censoring, sampling is done from the censoring distribution and compared with the event times and event times which comes first are marked observed while the others are marked censored. Here, the proportion of censoring can be controlled. To generate random censoring, the censoring distribution has to be independent (non-informative) from the observed distribution, so as to ensured unbiased survival estimates [17].

Therefore, in the chapters below, data will be generated with the above scenarios using different proportions of censoring and different sample sizes to analyse the influence of temperature on the survival times of technical devices. Moreover, the maximum likelihood methods are proper in deriving parameter survival estimates for this kind of censoring schemes[13].

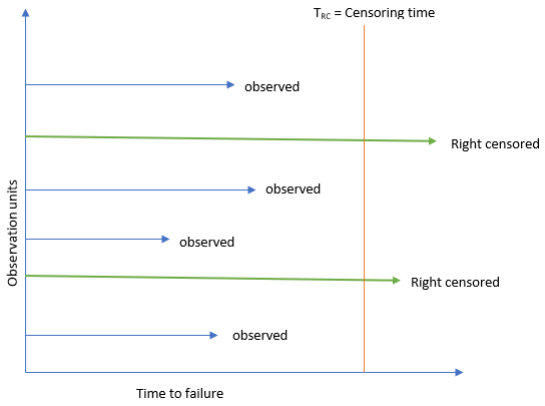

Figure 1: Illustrating right censored scheme

2.3.2 Interval Censoring

Interval censoring; Here the failure times of the experimental technical devices are not known exactly and as a result only the interval times L and R within which the event occurred is observed. This interval censoring scheme can be described as follows; assume n experimental units are placed under observation and let $T_1, ..., T_n$ be the survival times of this experimental units. Then a unit is interval censored if the time to failure/event occurs within the interval L and R. Furthermore in generating this censoring intervals L and R it is assumed that the intervals L and R are independent of T_i where T_i is observed only if T_i belongs to the interval L and R otherwise it is not observable. The assumption of non informative censoring is very common in the survival or reliability analysis and also happens quite naturally in real life scenarios [13]. In this project, four different schemes of interval censored data will be considered as follows;

1. **Fixed lower and Fixed right interval**; for example, in an experiment to monitor the effect of temperature on the survival times of technical devices, all devices begin the experiment at the same time (fixed lower).Later on, the investigator checks these technical devices at a fixed time and discovers a failure. However, the exact time of failure is not known implying the investigator records the visit time as the right interval time.Here all the machines have the same entry time (lower) and the same visit time (right).

2. **Fixed lower and Random right interval**; for example, in an experiment to monitor the effect of temperature on the survival times of technical devices, all devices begin the experiment at the same time (fixed lower). Later on, the investigator checks these technical devices at random times and discovers a failure. However, the exact time of failure is not known implying the investigator records the visit time as the right interval time.Here the machines have a fixed entry time but different visit times (right).

3. **Random lower and Random right interval**; for example, in an experiment to monitor the effect of temperature on the survival times of technical devices, all devices begin the experiment at the at different times (random lower). Later on, the investigator checks these technical devices at random times and discovers failures. Since, the exact time of failure is not known the investigator records the visit time as the right interval

time. Here the machines have different entry times and different visit times.

4. **Mixed interval** ; consider an experiment to monitor the effect of temperature on the survival times of technical devices, the exact times of failure or some devices are known while for other devices only the interval in which these events occurred. In this thesis, this type of scenario is referred to as mixed interval.

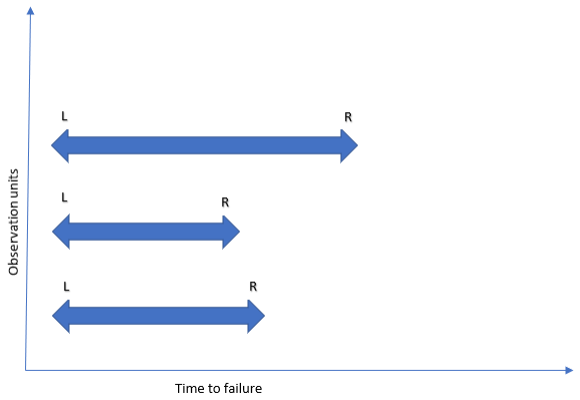

Figure 2: Illustrating interval censored scheme

2.4 Maximum Likelihood Estimation

The maximum likelihood method is the most versatile and most common method used in estimating the parameters of a Weibull distribution because the likelihood function is a sufficient statistic. Moreover, performing parametric estimation is easier when applying the Maximum likelihood framework to the chosen distribution in making parameter inferences [12].

A comparison of maximum likelihood and median rank regression estimation for a Weibull model, showed that under mild conditions and based on several studies the ML estimators remain consistent even in smaller sample sizes. Also, the ML is proper when cases like right censoring, interval censoring or when very limited information is observed about the units under experiment [8]. Therefore, adopting the maximum likelihood technique in estimating the influence of temperature on Weibull distributed time to events using a parametric model (AFT) will provide estimates with good statistical properties. A procedure with good statistical properties such as the maximum likelihood provides a good confidence interval for its estimates and vice versa [4]. Maximum likelihood estimates are efficient [6], referring to the fact that the estimates have the smallest variance (at least in large sample sizes). Furthermore, The maximum likelihood method produces estimates that approximately normally distributed in large sample size. The above mentioned characteristics of maximum likelihood methods makes them suitable in estimating parameters of a parametric (AFT) Weibull model. Nevertheless, estimation of parameters with maximum likelihood is complex and iterative, convergence is not usually guaranteed especially in small sample sizes [18].

2.4.1 Likelihood function uncensored observations

In the case of uncensored observations (all events completely observed), the likelihood is given by the product of the probability density functions at the observed time of events.

$$L = \prod_{i=1}^{n} f(T_i)$$

2.4.2 Likelihood function right censored observations

In the case of right censored observations, events are observed and other events occur after the censoring time. To get the likelihood of the entire sample(observed and censored cases), the product of the probability densities for the observed and censored observations is used. As a result, an indicator δ is needed which will differentiate observed and censored events. For right censored observations, the probability of all event times greater than the censored times are integrated. This integrated probability refers to the probability to survive at that censoring time. However, this is valid if the censoring and time to event generating are independent. Therefore the likelihood is given as below;

$$L = \prod_{i=1}^{n} f(T_{RC_i})^{\delta_i} \cdot S(T_{RC_i})^{1-\delta_i}$$

$$L(\beta, \alpha) = \prod_{i=1}^{n} \Big\{ \frac{1}{\alpha} \cdot \beta \cdot \left(\frac{1}{\alpha} \cdot T_{RC_i} \right) \cdot \exp\left[- \left(\frac{1}{\alpha} \cdot T_{RC_i} \right)^{\beta} \right] \Big\}^{\delta_i} \cdot \Big\{ \exp\left[- \left(\frac{1}{\alpha} \cdot T_{RC_i} \right)^{\beta} \right] \Big\}^{1-\delta_i}$$

From this, T_{RC_i} refers to right censored times, with indicator $(\delta) = 1$ if the time is an event and $(\delta) = 0$ if the time is censoring.

2.4.3 Likelihood function interval censored observations

In the case of interval censored observations, two times of the censoring for a device is known (L and R). To get the likelihood of the interval censored observations, the probability of all event times is used. This implies, calculation of the survival function at the lower (L) time minus the survival function at the upper(R) time. However, this is valid if the censoring and time to event generating are independent. Therefore the likelihood is given as below;

$$L = \prod_{i=1}^{n} [S(T_{L_i}) - S(T_{R_i})]$$

$$L(\beta, \alpha) = \prod_{i=1}^{n} \Big\{ \exp\left[- \left(\frac{1}{\beta} \cdot T_{L_i} \right)^{\alpha} \right] \Big\} - \exp\left[- \left(\frac{1}{\beta} \cdot T_{R_i} \right)^{\alpha} \right] \Big\}$$

Where; T_{L_i} refers to lower interval time and T_{R_i}, refers to the upper interval time.

2.5 MLE with Reduced Bias Adjustment (RBA)

The MLE-RBA is a well established theory for statistical quality control. It produces unbiased estimates for the Weibull distribution from the MLE estimate and is more accurate than MLE for smaller sample sizes [18]. In generating the MLE-RBA estimate of a parameter, the square root of an unbiased estimate of variance is used. In order to convert a MLE variance to an unbiased variance estimate,the MLE variance estimate is multiplied by $N/(N-1)$. This correction has been adopted by engineers all though this correction does not entirely unbias MLE variance [18]. As a result,the C_4 factor which eliminates the bias completely was adopted. This led to the RBA factor which is given as shown below;where n = number of events

$$RBA_\sigma = \left(\sqrt{(n/(n-1)}\right) / (C_4)$$

where:

$$C_4 = \sqrt{\frac{2}{n-1}} \frac{\left(\frac{n-2}{2}\right)!}{\left(\frac{n-3}{2}\right)!}$$

Multiplying MLE variance with RBA_σ leads to an unbiased estimate of the variance. Further research in the case of Weibull distributed data shows that multiplying MLE estimates with a larger correction, $(C_4)^6$ proves effective. This lead to an improvement in the estimates and elimination in bias as compared to the MLE [18]

$$MLE_{unbiased} = MLE_{estimate}\left(C_4^6\right)$$

2.6 Parametric Model (AFT)

A parametric model is a model in which the response (survival time) is assumed to follow a distribution (in this case a Weibull distribution) [11]. For the case of parametric survival models, the main assumption is the accelerated failure assumption (AFT). With AFT models, the covariates have a multiplicative effect on the survival times of the responses. For example, in the case of an experiment to describe the influence of temperature on technical devices, an AFT model fitted to Weibull distributed survival times provides an acceleration factor. The acceleration factor which is the estimate of temperature on survival times of these technical devices describes behaviour of temperature on the survival of these devices. Therefore, AFT models have been adopted and are suitable in measuring the effect of temperature on survival times of technical devices using the Pecks relationship [5]. Moreover, estimating the effect of covariates on survival times using these AFT models approach is appropriate with the maximum likelihood technique. This approach is also fully applicable to both right and interval censored data [12] but biased estimates are obtained when applied to responses that do not follow the chosen distribution. Since the Weibull distribution is adopted in this study, evaluation of its appropriates is essential. In order to evaluate the appropriates of a AFT model on a set of fitted responses, the log(-log) of the Kaplan-Meier survival times is plotted against the log of time. A linear relationship is expected to validate a Weibull parametric model fit. The weakness of the above model diagnostic approach is that they do not adjust for the effect of covariates [17]. However, a modified Kaplan Meier survivor functions which adjust for the covariates can be employed and when graphed against the log of time a

straight line indicates proper model fit.

Let T_i be a random variable denoting the event time for the i^{th} technical device and let $X_{1i}, ..., X_i$ be the values of temperature for that same technical device. Then an AFT model is of the form;

$$log(T_I) = \alpha_0 + \alpha_1 X + \sigma\epsilon$$

$$log(T_i) = \alpha_0 + \alpha_1 \left(\frac{1}{Kt_i}\right) + \sigma\epsilon$$

where;

- T_i is a random variable denoting Weibull distributed event time for the i^{th} technical device.
- α_0 is the model intercept which can be used with σ to describes the baseline Weibull distribution.
- α_1 is the estimated temperature coefficient. The acceleration factor is obtained by e^{α_1}. This implies a positive coefficient indicates the covariate causes worse survival or shorter survival times by a factor of e^{α_1} and a negative effect indicates longer survival times by a factor of e^{α_1}.
- $t =$ values of temperature in Kelvin degrees.
- $k =$ Boltzmann constant ($8.617333262 \times 10^{-5} eVK^{-1}$).
- ϵ_i is a random disturbance term with a standard extreme value distribution (SEV). If $0 < \sigma < 0.5$ implying the hazard is increasing at an increasing rate and σ is given by $1/\alpha$
- α_0, α_1 and σ are the parameters to be estimated using the maximum likelihood method.

The estimation of parameters from an AFT model can be done using the survival R packages and the SAS PROC LIFEREG procedure. Both procedures are used to obtain results for the fitted model above. Both methods give approximately equal results of the estimates, standard errors and p-values.

2.7 Mean Square Error

Let $\hat{\theta}$ represent an estimator of an unknown parameter θ from a random sample $X_1, X_2, ..., X_N$. A deviation of $\hat{\theta}$ from the true population value θ, $|\hat{\theta} - \theta|$ or $(\hat{\theta} - \theta)^2$ measures the quality of the estimator. However, since $\hat{\theta}$ is random, the average is often used. The MSE of an estimator $\hat{\theta}$ is given by

$$MSE_{\hat{\theta}} = E(\hat{\theta} - \theta)^2$$
$$= Var(\hat{\theta})^2 + (E(\hat{\theta}) - \theta)^2)$$
$$= Var(\hat{\theta}) + [E(\hat{\theta}) - \theta]^2$$

Thus, MSE has two components; the $Var(\hat{\theta})$ component estimates the variability of the estimator and the $[E(\hat{\theta}) - \theta]^2$ estimates the bias of the estimator. Therefore, the MSE measures the precision and accuracy of an estimator [1].

To evaluate the effect of temperature on the survival times of technical devices, the variance, bias and thus MSE of the estimator is used go describe the precision and accuracy of the estimator. These estimates are gotten from the maximum likelihood after fitting the model. Another, important estimate is the standard error of the estimate.

2.8 Coverage probabilities

The confidence interval is the region that contains the value of interest, in this case the population value. The estimation of this expected value is built under the properties of the AFT model which usually result in estimation errors. Therefore, in order to provide a better description of true population value, an interval is defined which contains a range of values around the estimated value which is likely to contain the the true value (population value). However, to ensure that the intervals produced are independent of a particular sample, 1000 samples are created and the coverage probabilities are used. The coverage probability refers to the probability that constructing random regions will produce intervals covering the true population value [9]. Therefore the coverage probability refers to the count over many replications (in this case a 1000 replications) that the interval contains the target value. From the central limit theorem, the estimated values will be approximately normal if the sample size is not too small. This implies the confidence interval for the estimate (α_1) can be calculated as follows;

$$CI = \alpha_1 \pm Z_c \cdot std.err$$

$$CI_{adj} = \alpha_{1_{adj}} \pm Z_c \cdot std.err_{adj}$$

Where;

- α_1 = MLE for the effect of temperature.
- $\alpha_{1_{adj}} = MLE_{unbiased}$ for the effect of temperature adjusted using RBA.
- $std.err$ = standard error for effect of temperature.
- $std.err_{adj}$ = adjusted standard error for effect of temperature adjusted using RBA.
- Z_c = Z value for confidence level obtained from the area under the normal curve.

The confidence and adjusted confidence intervals are calculated for every sample using the equations above. The coverage probabilities are then calculated for the 1000 samples by counting the number of times the confidence intervals and adjusted confidence intervals contain the target value to derive the coverage probabilities and adjusted coverage probabilities respectively.

2.9 Data Exploration

Graphical techniques are used to explore the generated time to events. The various plots adopted include; the cumulative density plot for generated time events, a histogram of Weibull generated eventtimes using the population

parameters and the Weibull plot. The cumulative distribution function (cdf) calculates the cumulative probability for a given x-value. However, in this case the cdf describes the percentage that will fail at any time or the fraction of technical devices failing over time (T). The histogram shows the distribution of Weibull time to events. The y-axis of the histogram shows the count of records while the x-axis contains event time values. The vertical scale of the Weibull plot shows the cdf which describes the percentage of technical devices that will fail over time. With the Weibull plot, the time parameter is a function of the temperature on the horizontal scale. In constructing the Weibull plot, the event times are ranked from the earliest failures to the latest failures.

2.10 Statistical Software

Through out the project R-Studio Version 1.4.1106 and SAS 9.4 were used to conduct statistical analysis and make plots. The packages purr, dplyr, altair and ggplot2 were used for data management and visualization. The package survival was used for fitting the parametric models. Hypothesis were tested at a 5% significance level.

3 Simulation of Dataset

3.1 Overview

Different datasets were generated which represents different design scenarios during this study. Datasets are generated for uncensored units, right censored units (type I, type II, type III) and interval censored units. The aim is to compare the estimates obtained from the various scenarios with the population estimate. Consequently, the coverage probabilities, MLE with reduced bias adjustment, mean estimates, mean biases, mean variances and mean square error of the estimates are measured under the various scenarios.

3.1.1 Procedure

- 1000 different datasets of size (n) with survival times which are from a Weibull distribution with shape(α) and scale(β) parameters as shown below are generated;

$$T \sim Weibull(\beta, \alpha)$$

  where ; $\beta = e^{\alpha_0 + \alpha_1\left(\frac{1}{Kt}\right)}$ with $\alpha_0 = -31.39$, $\alpha_1 = 0.98$ and $\alpha = 3.02$ gotten from the population. $K = (8,61710\text{–}5eV/K)$ and $t =$ temperature in Kelvin.

- An AFT model with the generated time to events and temperature (Kelvin) as an explanatory variable is fitted to the 1000 different datasets simulated from a particular observational scheme. The model below illustrates the model fitted to one dataset of size(n);

$$Y = log(T_i) = \alpha_0 + \alpha_1\left(\frac{1}{Kt_i}\right) + \epsilon(\sigma_i) \tag{1}$$

  where;

  - T_i is a random variable denoting Weibull distributed event time for the i^{th} technical device.
  - α_0 is the model intercept which can be used with σ to describes the baseline Weibull distribution.
  - α_1 is the estimated temperature coefficient. The acceleration factor is obtained by e^{α_1}. This implies a positive coefficient indicates the covariate causes worse survival or shorter survival times by a factor of e^{α_1} and a negative effect indicates longer survival times by a factor of e^{α_1}.
  - $t =$ values of temperature in Kelvin degrees.
  - $k =$ Boltzmann constant ($8.617333262 \times 10^{-5} eVK^{-1}$).
  - ϵ_i is a random disturbance term with a standard extreme value distribution (SEV). If $0 < \sigma < 0.5$ implying the hazard is increasing at an increasing rate and σ is given by $1/\alpha$
  - α_0, α_1 and σ are the parameters to be estimated using the maximum likelihood method.

- Various characteristics of this estimate (α_1) are measured from fitting the model to the thousand datasets. These include; the coverage probability, the MLE with reduced bias adjustment, the mean estimations, mean biases, mean variances and Means square error.
- The above procedures are repeated for the various censoring scheme with varying sample sizes and the uncertainty around the estimates are measured.
- A particular seed is used to ensure the same results are gotten all the time.

3.2 Uncensored Data

Uncensored data refers to technical units whose time to failure was observed (known). In the case of uncensored datasets, the time to event is generated as shown above and with the assumption that all event times are observed time.

$$T_i = rweibull(n, scale = \beta, shape = \alpha)$$

Where ;

- T_i = represents the time to failure for a technical device.
- $\beta = e^{\alpha_0+\alpha_1\left(\frac{1}{K_t}\right)}$ = scale paramaeter. With α_0 and α_1 from the population.
- n = sample size needed.
- α = shape parameter taken from the population.
- $rweibull$ = R function which generates Weibull distributed event times.

An AFT model is fitted to 1000 datasets of size (n), with temperature as an explanatory variable and the uncertainty around the influence of temperature is examine.

3.3 Right Censored Data

3.3.1 Generating Type I dataset

In type 1 right censoring, the censoring times are pre-specified. In order to ensure various proportions of censoring in the various datasets, the following equation 2 is used to determine the censoring time based on the desired proportion of censoring . Equation 3 is used to generate event times with Weibull parameters;

$$cens_T = qweibull(rate, scale = \beta, shape = \alpha) \tag{2}$$

$$T_i^* = rweibull(n, scale = \beta, shape = \alpha) \tag{3}$$

Where;

- $cens_T$ =The time to censoring event.
- T_i^* = represents the time to failure for a technical device.
- $rate$ = proportion of censoring required in sample dataset.
- $\beta = e^{\alpha_0+\alpha_1\left(\frac{1}{Kt}\right)}$ = scale parameter. With α_0 and α_1 from the population.
- α = shape parameter.
- *scale* and *shape* parameters are independent from true time to event.
- *qweibull* = R function which computes the quantile of a Weibull distribution in R.
- n = sample size needed.

Fitting the model in equation 1 above , requires an indicator (δ) to distinguish observed events from censored events. The observed time becomes $T = min(T_i^*, cens_T)$. This implies $\delta = 1$ if T is observed time failure and $\delta = 0$ if T is censored time. An AFT model is fitted with different proportions of censoring.

3.3.2 Generating Type II dataset

In type 2 right censoring, a case where a total number of technical devices (n) are monitored until a pre-defined fraction have registered an event. The procedure for generating type 2 is as follows Procedure;

1. Weibull event times are generated with scale and shape parameters using equation 2 above.
2. The generated event times T_i are sorted in ascending order.
3. The first m eventtimes are marked as observed and $n - m$ becomes censored. Thus an indicator variable is created with m eventtimes observed($\delta = 1$) and n-m eventtimes censored($\delta = 0$).
4. Data is generated for different m values and an AFT model is fitted.

3.3.3 Generating Type III dataset

Type 3 censoring occurs in a random manner which is not controlled by the investigator or pre-defined. Parameter estimates obtained when censoring is non-informative or censoring is independent of outcome does not introduce bias as compared to the case where the censoring is informative[13]. As a result censoring time is generated independent of event time. The procedure for generating type 3 is as follows Procedure;

$$T_i^* = rweibull(n, scale = \beta, shape = \alpha) \tag{4}$$

$$cens_T = runif(n, min = a, max = b) \tag{5}$$

1. Weibull event times T_i^* are generated with scale and shape parameters using equation 4 above.

2. The R function $runif$ from equation 5 above, is used to generate uniformly distributed values as the censoring time. Where n =sample size, min = min value and max = maximum value. The values ranged from $min = 0.1$ to $max = 6$.

3. The observed time becomes $T = min(T_i^*, cens_T)$. This implies $\delta = 1$ if T is observed time failure and $\delta = 0$ if T is censored time.

4. Data is generated for different proportions of censoring by increasing the maximum value in the censoring time equation(5).

3.4 Interval Censored Data

In the case of interval censored data, two different broad scenarios are considered. Firstly, the mixed interval scenario where, for some devices the exact times of failure are known and for others only the interval at which the failure occurs is known. Secondly, the other intervals where only the intervals are known and no exact failure times included.

3.4.1 Mixed Intervals

In order to generate mixed intervals the following procedure is used;

$$T_i^* = rweibull(n, scale = \beta, shape = \alpha)$$
$$cens_T = runif(n, min = 0.1, max = 9)$$
$$status = ifelse(t_c < T_i^*, 1, 0)$$
$$L = ifelse(status == 1, T_i^*, 0.1)$$
$$R = ifelse(status == 0, cens_T, T_i^*)$$

1. Weibull event times T_i^* are generated with scale and shape parameters using equation 4 above

2. The R function $runif$ above, is used to generate uniformly distributed values as the censoring time. Where n =sample size, min = min value and max = maximum value. The values ranged from $min = 0.1$ to $max = 6$.

3. The observed time becomes $T = min(T_i^*, cens_T)$. This implies $\delta = 1$ if T is observed time failure and $\delta = 0$ if T is censored time.

4. In order to create interval censored data, a vector L & R is created. Vector L is such that devices with exact failure times have a lower interval equal to their exact failure time and censored observation have lower intervals equal to 0.1 (0.1 is used to ensure model convergence).

5. To create the right interval, devices whose exact time to failure are not known $\delta = 0$ have the value $R = cens_T$.

6. Finally events with $\delta = 1$ have L & R to be their exact failure times and observations with $\delta = 0$ have $L = 0.1$ and $R = cens_T$.

7. Following this procedure the number of interval censored observations is increased by increasing the max value in the runiff function. This procedure thereby generates a dataset with a mixture of devices with exact time of events and devices where only the interval within which the events occurs are known.

3.4.2 Other Intervals

- **Fixed left (L) and random (R) right boundary**; here, all technical devices begin the experiment at the same time. This implies, the time when the technical device is put into the experiment is recorded as the left boundary (L) while the upper boundary is some random time when the experimenter checks the machine for an event (T). It is assumed that, the event occurred between the interval L and R and are independent from the time to event (T). With these method, a time grid is set from 0.1 to 10 on the time axis, leading to n different intervals on the time grid. In order to generate these censoring intervals independently from the time to event, the values of L and R are generated from a continuous uniform distribution in the interval (0.1,10).

- **Random left (L) and random (R) right boundaries**; here, all technical devices have a random time of entry into the experiment and random time of which the machine is inspected for an event. Implying, random left and right boundaries for the interval censored observations. It is assumed that, the event occurred between the interval L and R and are independent from the time to event. In generating these intervals, it is assumed that lower temperatures will have longer time to failure as compared to higher temperatures as such longer random inspection times are used. With these method, a time grid is set from 0.1 to 10 on the time axis leading to n different intervals on the time grid with different L and R values. In order to generate these censoring intervals independently from the time to event, the values of L and R are generated from a continuous uniform distribution in the interval (0.1,10).

- **Fixed left (L) and (R) right boundaries**; here, all technical devices have a fixed time of entry into the experiment and fixed time of which the machine is inspected for an event. Implying, fixed left and right boundaries for the interval censored observations. It is assumed that, the event occurred between the interval L and R and are independent from the time to event. In generating these intervals, it is assumed that lower temperatures will have longer time to failure as compared to higher temperatures as such longer random inspection times are used. With these method, a time grid is set from 0.1 to 10 on the time axis and based on the level of the temperature. This lead to 3 different intervals on the time grid with different L and R values. In order to generate these censoring intervals independently from the time to event, the values of L and R are generated from a continuous uniform distribution in the interval (0.1,10).

4 Results

Figures 3 and 4 shows the distribution of the 1000 eventtimes from a Weibull distribution using the shape and scale parameters as mentioned above. Figure 4 illustrates the distribution per level of temperature with temperatures of 95 degrees having shorter time to events than lower temperatures. Also, figure 3 shows that the distribution is right skewed.

4.1 Exploratory Data Analysis

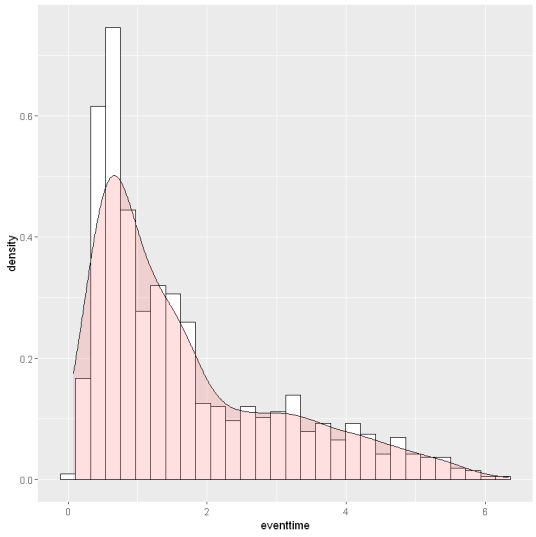

Figure 3: Distribution of event times

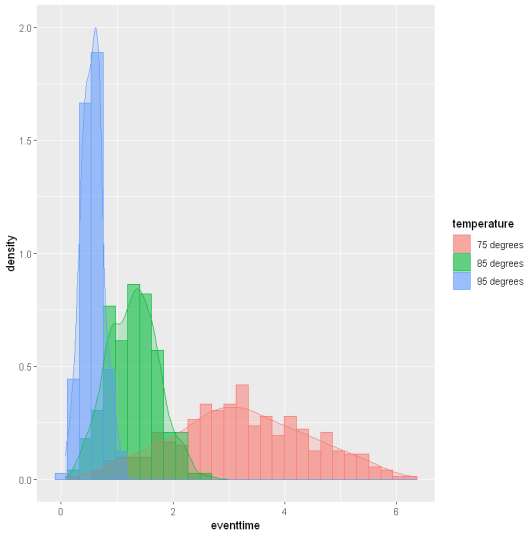

Figure 4: Distribution of event times by temperature

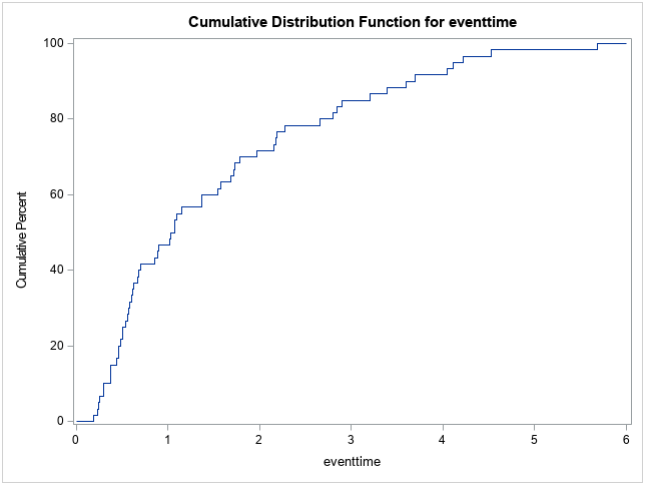

Figure 5: Cumulative Density Plot

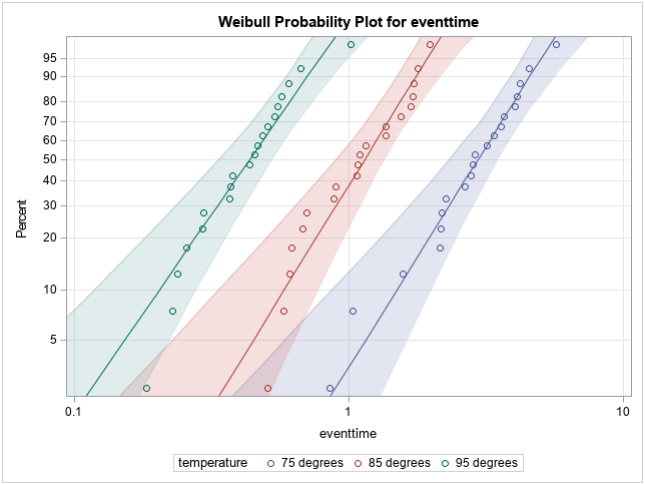

Figure 6: Weibull Probability Plot

Figures 5 shows the cumulative density plot for generated event times using the population parameters. This plot shows that 60% of the technical device will fail between 1 and 2 on the time axis. Figure 6 describes the percentage of devices that will fail up to time(t). The horizontal axis is the time to failure and the vertical axis is the CDF

describing the the percentage that will fail at a given time. From the plot, devices under temperature 95 degrees have shorter times to failure, followed by 85 degrees and 75 degrees in that order. Another use of the figure 6 is for model diagnostics. The plot shows maximum likelihood fitted lines and confidence intervals to each temperature level. Since, the horizontal lines fall within the confidence band and as such the event times are from a Weibull distribution.

4.2 Sample Output of Fitted Models

In this section, two sample examples of the results obtained from fitting an AFT model is investigated. Here two datasets are considered; the uncensored dataset and a type I right censored data with 25% censored events. The intercept, temperature and shape estimates can be used to define the baseline Weibull distribution as described in section 2.6. The true population parameters value for the Weibull distribution as described in section 2.6 are given as $\alpha_0 = -31.39, \alpha_1 = 0.98$ and $alpha = 3.02$. From the table, $\alpha_0 = intercept$, $\alpha_1 = temperature$ and $\alpha = shape$. Therefore, the confidence limits from table 1 shows that the population parameter estimates lie within the 95% confidence limits of the sample estimates. The scale parameter here does not define the Weibull distribution but can be used to describe the hazard rate. Since the scale value lies between 0 and 0.5, the hazard is increasing at an increasing rate. Additionally, the temperature estimate is highly significant implying the temperature has an effect on the log survival times of the technical devices. The positive temperature estimate illustrates that a unit increase in temperature causes shorter survival times by a factor of about 2.5 for the uncensored and about 2.7 in the case of type I right censored.

However, the goal through out this thesis is to compare the influence of the temperature on technical devices within different design scenarios. The design scenarios here refers to the various censoring schemes discussed in section 2.3. For an ideal design, the estimate of temperature in that design should lie very close to the true population value. As a result, 1000 different random samples of size n and based on the various design factors are generated, a model is fitted to the data and the properties of the temperature estimate is diagnosed. Additionally, comparisons are made between the various designs in terms of MSE of the estimators, variance, bias and coverage.

	Uncensored Data (sample size = 60)					
	Estimate	std.error	95% confidence limits		Chi-square	p-value
Intercept	-29.6381	1.8077	-33.1811	-26.0952	268.83	<.0001
temperature	0.9231	0.0558	0.8138	1.0323	274.12	<.0001
Scale	0.3327	0.0339	0.2724	0.4062		
Shape	3.0061	0.3065	2.4616	3.6710		
	Type 1 right censored (sample size=60 ,25% censored)					
Intercept	-31.4090	1.3745	-34.1031	-28.7150	522.15	<.0001
temperature	0.9830	0.0424	0.8999	1.0662	536.48	<.0001
Scale	0.2142	0.0227	0.1740	0.2636		
Shape	4.6688	0.4947	3.7932	5.7464		

Table 1: Example of model output with model parameters

4.3 Uncensored Design

This section contain results obtained from fitting an AFT model to different sample sizes in the uncensored design scheme. The procedure involves generating time to events from the Weibull distribution using the population parameter values. The estimates in table 2, refers to mean estimates gotten by averaging over over 1000 different samples of a given dataset size. For example, the second column of table 2 refers to estimates gotten from a fitting the model to 1000 different datasets of sample size 10 and the average of the estimate (temperature), variance, standard error, bias, MSE and coverage are obtained. The adjusted estimates refers to the maximum likelihood estimates which have been corrected by using the RBA discussed above in section 2.5. The standard error and variance are gotten from the output of the fitted model in R. While the bias, MSE and coverage are estimated using the formulas discussed in sections 2.7 and section 2.8 respectively.

Overall, the trend in the table shows increasing coverage probability with increasing sample size. On the other hand, the trend in this scheme shows the population estimate (0.98) is always underestimated but bias reduces as the sample size increases. Also, the MSE, variance and standard error all decrease with increasing sample sizes. Figures 7 provides a graphical representation of the adjusted coverage probability vs the sample size in the uncensored scheme. Figure 8 provides a graphical representation of the coverage probability vs the sample size while taking into account the MSE of estimator, this implies the size of the circle represents the magnitude of the MSE. In figure 7, a steady decrease is seen in the coverage probability with decreasing sample size and with a steeper dip in coverage with a small size less than 20. The curve also becomes relatively flat between a sample size of 40 and 60, thereby illustrating that this interval of sample sizes then to produce about the same coverage probabilities although not quite above 95. Figure 8 points out that although a sample size of 10 attains a pretty good coverage (87.5), its MSE error is the largest as compared to the others. As a result of the large MSE, the results are not precised or accurate as compared to the others.

	Sample Size				
	10	**20**	**30**	**40**	**60**
Estimate	0.9749	0.9755	0.9823	0.9783	0.9793
Variance	0.0207	0.0104	0.0064	0.0049	0.0033
Std error	0.1357	0.0994	0.0787	0.0695	0.0567
Bias	-0.0051	-0.0045	-0.0023	-0.0017	-0.0007
MSE	0.0468	0.0222	0.0136	0.0101	0.0068
Coverage	87.3	91.5	92.2	94.2	94.2
Adj.estimate	0.9504	0.951	0.9538	0.9571	0.9547
Adj.variance	0.0214	0.0107	0.0051	0.0047	0.0034
Adj.Std error	0.138	0.1011	0.0707	0.0681	0.0577
Adj.Bias	-0.0296	-0.029	-0.0262	-0.0229	-0.0253
Adj.MSE	0.0471	0.0228	0.0139	0.0107	0.0074
Adj.coverage	87.5	91.9	92.6	94.4	94.7

Table 2: Estimates For Uncensored AFT Model

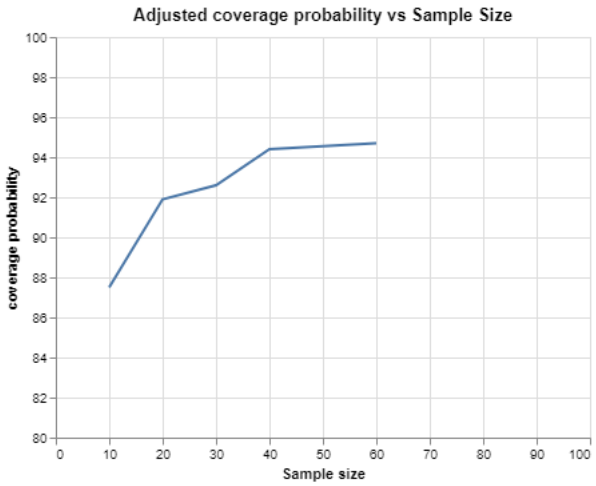

Figure 7: Coverage Uncensored Design

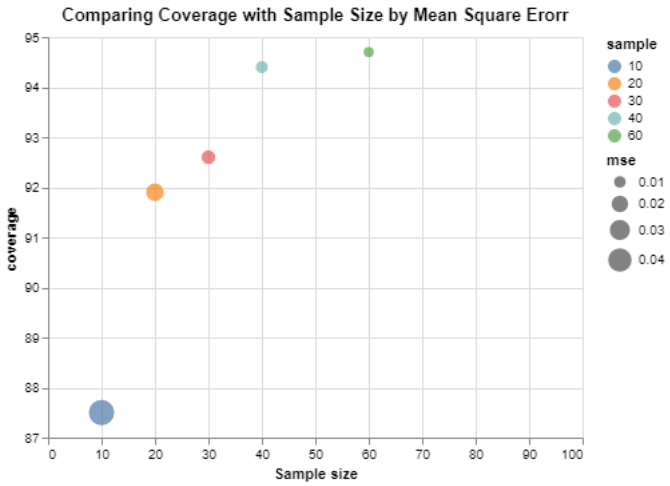

Figure 8: MSE Uncensored Design

4.4 Right Censored Design

This section, provides the results from fitting an AFT model to the datasets generated from various right censored types. Results are generated for sample sizes of $60, 40$ and 20 with different proportions of censoring of $5\%, 10\%, 20\%, 30\%, 50\%$ and 70% for each sample size. This section begins by comparing the coverage, MSE and bias for the various types of right censoring. Finally, an overall comparison is made with respect to the coverage , MSE and bias of each type of right censored design to find out the design with the best coverage, smallest bias and smallest MSE.

4.4.1 Type I Right Censored Design

Type I right censoring refers to a scenario where the censoring time is controlled by the investigator. In this case, different sample sizes with varying proportion of censoring were generated and the model fitted as described in section 3.3.1. The estimate of temperature is then analysed and the values reported as shown in table 3. Table 3 shows the various characteristic of the parameter estimate from various sample sizes with different proportions. Nevertheless, a graphical approach is used to provide a summary of the results. Figure 9 shows the relationship between coverage probability and percentage of censoring in type I. From the plot, a steady decrease in coverage is observed with increasing percentages of censoring but for a sample size of 40 and 60, little difference is observed in the case where the percentage of censoring lies between $0 - 10\%$. A general trend shows a steeper decline in coverage for sample size of 20.

Figure 10 shows the relationship between MSE and sample size in the case of type I design scheme. The blue bar in Figure 10 represents a dataset of $size = 20$ and shows the MSE averaged over all proportions of censoring with $size = 20$. The same is done for sample $size = 40$ and sample $size = 60$. Therefore, overall the MSE is largest for sample size $= 20$ while taking into consideration the various proportions censoring. Figure 11 compares the bias per sample size over various percentages of censoring. The plot shows for sample sizes 20 and 40, the estimate of temperature is increasingly overestimated as percentage of censoring increases. However, estimates from sample size

= 60 show a steady bias over increasing percentage of censoring. Moreover, the bias is lowest with smaller censoring and increases as the percentage of censoring increases.

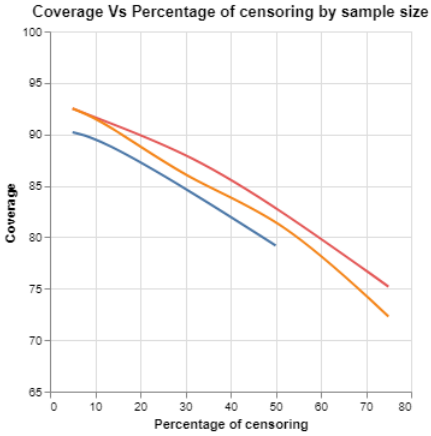

Figure 9: Coverage Type I Design

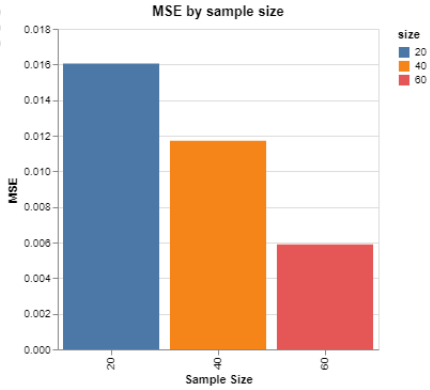

Figure 10: MSE Type I Design

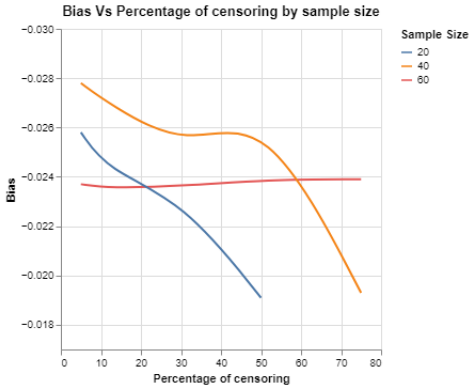

Figure 11: Bias Type I Design

4.4.2 Type II Right Censored Design

For Type II Right Censored Data, a total number of technical devices (n) are monitored until a pre-defined fraction have registered an event. The percentage of censoring in type II refers to the fraction of devices which were marked as censored. For example a percentage of censoring of 5 for a sample size of 60 implies the experiment was stopped after 57 events were observed. The estimate of temperature on the survival times in this design is analysed and the values reported as shown in table 4. Table 4 shows the various properties of the temperature estimate from various sample sizes with different proportions. Nevertheless, a graphical approach is used to provide a summary of the results. Figure 12 shows the relationship between coverage probability and percentage of censoring in type II. Figure 12, surprisingly shows larger coverage for smaller sample sample sizes but at a price of higher MSE. Figure 13 shows the relationship of MSE and sample size. Figure 13 shows smaller sample size of 20 has the highest MSE. Although figure 12 shows a higher coverage for sample size of 20, figure 13 indicates that this coverage is with a higher MSE. As a result, the MSE is on average about 27 times larger in sample size of 20 implying highly un-precise estimates an inaccurate and poor estimation of the temperature effect. Overall in terms of bias, figures 14 shows that with type II design, the sample temperature estimate is always overestimated as compared to the population true value and this bias increases as the percentage of censoring increases.

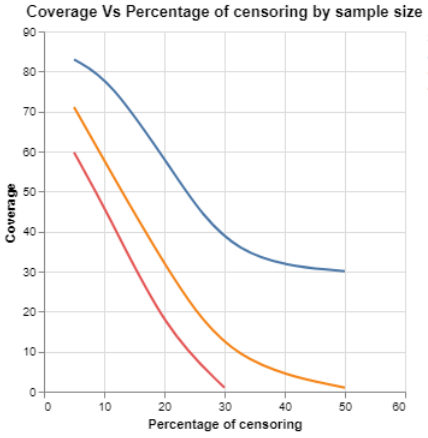

Figure 12: Coverage Type II Design

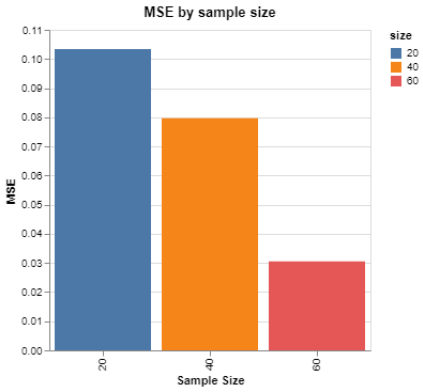

Figure 13: MSE Type II Design

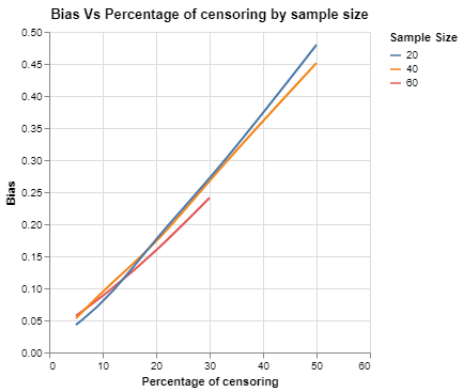

Figure 14: Bias Type II Design

4.4.3 Type III Right Censored Design

For Type III Right Censored Data, censoring occurs in a random manner which is not controlled by the investigator or pre-defined. In this case, different sample sizes with varying proportion of censoring were generated and the model fitted. The estimate of temperature is then analysed and the values reported as shown in table 5. Table 5 shows the various characteristic of the parameter estimate from various sample sizes with different proportions. Nevertheless, a graphical approach is used to provide a summary of the results. Figure 15 shows the coverage vs percentage of censoring per sample size. From the plot, sample sizes of 20 and 40 show a higher coverage than sample sizes 60. However, these figure 16 shows that the estimates gotten from this smaller smaller size (especially sample size of 20) are quite inaccurate and precised. Figure 16 shows a similar amount of MSE for sample sizes 40 and 60. Overall for the type III design, the population estimate is always increasingly underestimated with increasing proportions of censoring.

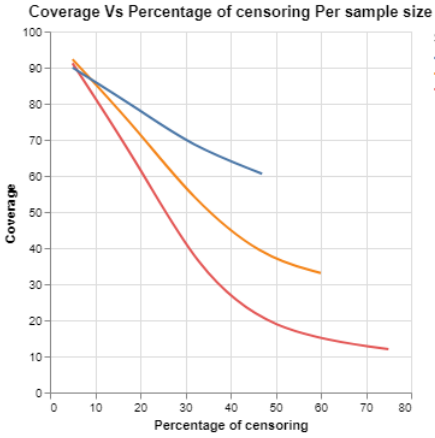

Figure 15: Coverage Type III Design

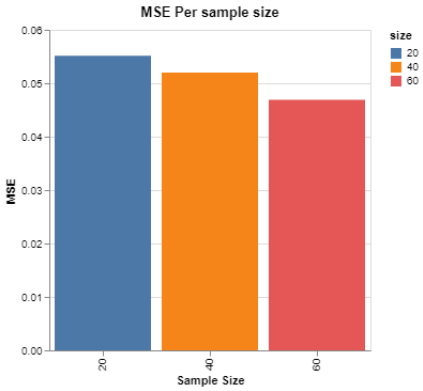

Figure 16: MSE Type III Design

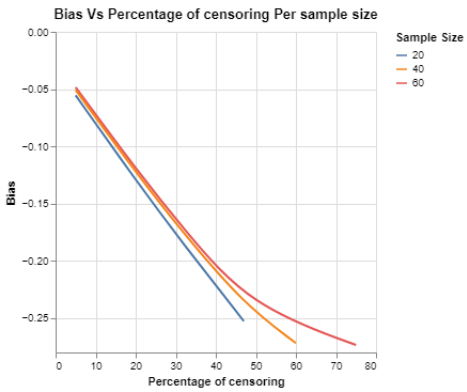

Figure 17: Bias Type III Design

4.4.4 Comparing various right types of censoring

The previous sections above looked at the coverage, MSE and bias for specific types of right censored data. In this subsection comparisons are made between the various type of right censored design scheme in terms of MSE, coverage and bias to illustrate which type of right censoring design provides estimates closest to the true population estimate.

Figure 18 shows the coverage vs percentage of censoring per right censoring type. From the plot, the coverage in type I outperforms all others with increasing censoring. However, with a censoring percentage between $0 - 10\%$ the type I and type III tend to produce similar coverage. Moreover, figure 19 illustrates the MSE from the estimation of the temperature effect in type I are the smallest. The trend in MSE values points out that the type I estimation of the temperature effect is the least precised and provides the least coverage of the true population temperature estimate. but increasing MSE is observed for types II and III with increasing percentage of censoring. Overall, all right censoring design schemes tend to either over-estimate or under-estimate the true population temperature parameter but only the type I censoring provides least bias. The type II is seen to always over-estimate while the III is seen to always under-estimate with increasing proportions of censoring.

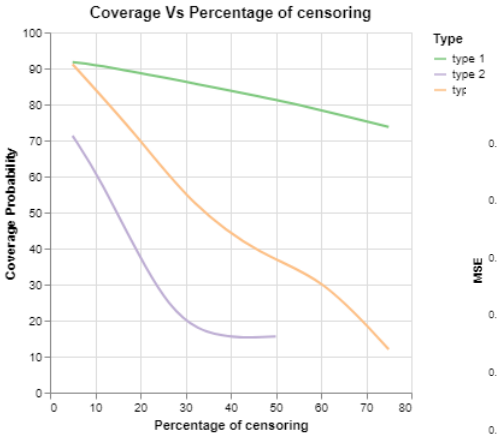

Figure 18: Coverage Right censored

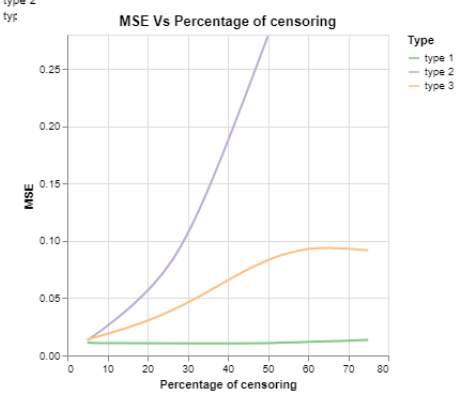

Figure 19: MSE Right censored

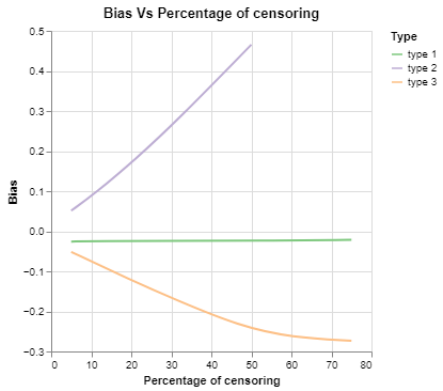

Figure 20: Bias Right censored

4.5 Interval Censored Design

4.5.1 Mix Interval

The mix interval censoring refers to a design where both exact events and interval censored events are recorded. In this design, the censoring is random with percentage of censoring referring to the percentage of interval censored observations in a given dataset of sample size (n). In line with the objective of the thesis, the estimate of temperature when using this design is analysed and compared with the true population value. Table 6 points out the various properties associated with the estimate under this design type. Nevertheless, a graphical approach is used to provide a summary of the results. Figure 21 shows the coverage vs percentage of censoring per sample size. The plot shows decreasing coverage probabilities as the amount of interval observations increases in the dataset however pretty good

coverage ranging from about 94.5% to about 78% is observed across the various sample sizes. Figure 22 shows that the MSE doubles for sample size of 20 as compared to that of 40 and 60. Figure 23 shows the the trend in the bias of the effect of temperature from the samples. The bias begins by being underestimated and as the amount of interval censored observations in the dataset increases the estimate increasingly shifts from becoming underestimated to over-estimated.

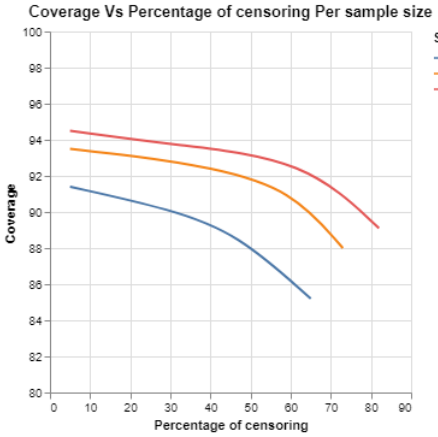

Figure 21: Coverage Mix interval

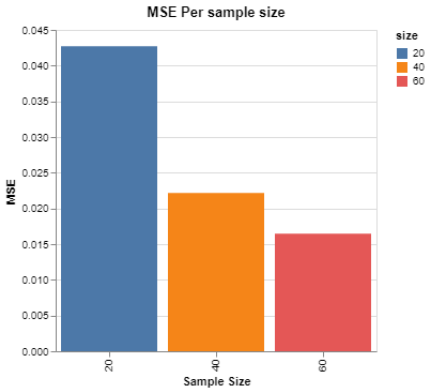

Figure 22: MSE Mix interval

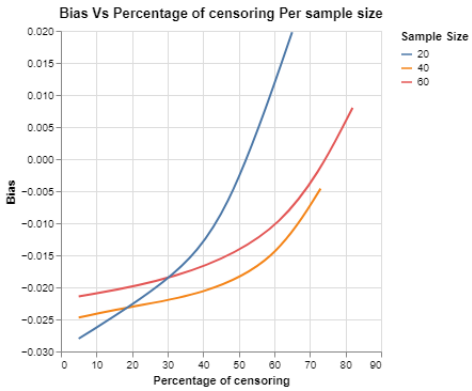

Figure 23: Bias Mix interval

4.5.2 Other Intervals

This section presents results from an interval design scheme where all events occur within an interval (L,R) but the exact time of events are not known. As stated above in section 2.3.2 three types of scenarios are considered. In this project 3 different scenarios are examined but only two are presented in this section. The two scenarios are the fixed lower and fixed right interval and the fixed lower and random right intervals. However parameter estimates for the random right and random left intervals are available in Table 7. The results were left out because of very low coverage and very high MSE values were obtained. Therefore, graphical analysis are done considering only the two mentioned scenarios.

Figure 24, displays the coverage probability vs sample size for the types of interval schemes considered here. For a better comparison a look at figure 25 along side Figure 24 demonstrates that the high coverage of the fixed L and Random R intervals comes with a very high MSE. On the other hand, for a sample size of 60, the MSE for the fixed L and Random R interval is about 25 times larger. This implies the fixed L and R intervals provides an estimate of the temperature effect with a smaller MSE as compared to the others. Also, figure 26 shows for both interval scheme, the the temperature estimate is always under-estimated but more with the fixed L and Random R intervals. Overall, the values of MSE from the estimation of interval censored data is very high, thereby illustrating poor accuracy and precision in the estimation of the temperature effect. Figure 25 shows sample sizes hitting about 100% coverage, which is as a result of very poor estimation of these confidence intervals or mean estimates.

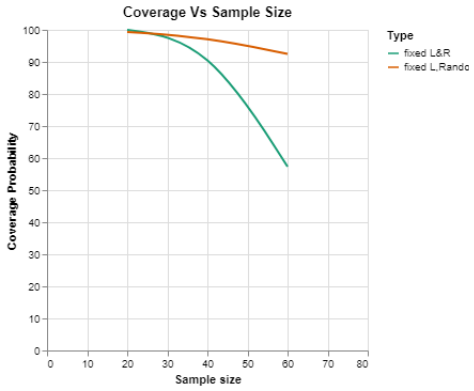

Figure 24: Interval Coverage

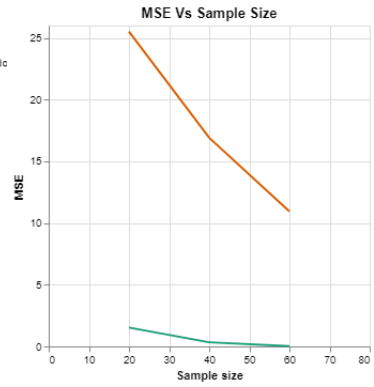

Figure 25: Interval MSE

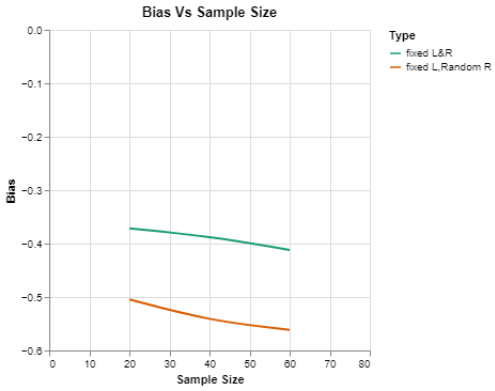

Figure 26: Interval Bias

4.6 Overall Comparison of Designs

So far, various designs schemes have been analysed specifically. Here this section presents an overall comparison of the various designs schemes. The design schemes are compared in terms of coverage probability and MSE to find out which scenario produces the highest coverage while considering the precision (MSE) of the estimation procedure. Figure 27 shows the scenario of fixed L and Random R provides the most coverage, followed by the uncensored scenario and the type I in that order. On the left side, figure 28, the scenario of fixed L and Random R provides the highest MSE which is about 80 times larger than others (table 7). Looking at both plots in terms of coverage and balancing for MSE, the type I demonstrates a reasonable coverage of the estimates with a smaller MSE or higher precision than others. Therefore, in order to draw proper conclusions, another comparison is made without the interval censored data.

Figures 29 & 30 provides comparison of coverage and MSE for mixed interval, type I, II and III designs. The interval designs were left out since the provided and extremely large value for MSE and the uncensored was left out since this design refers to all event times know. However 27 still provides a comparison of all available designs. Figures 29 shows a similar coverage between the mix interval design and the type I but in terms of precision and accuracy of the estimate, figure 30 shows a higher MSE for the mix interval design and the type I produces the least MSE.

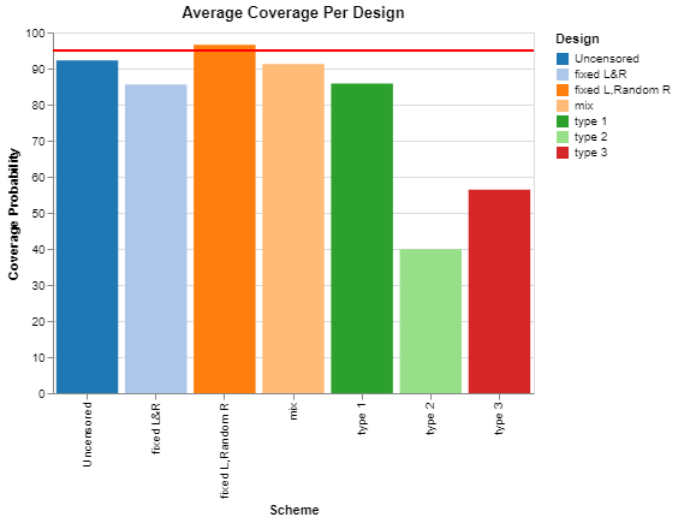

Figure 27: Overall Coverage Comparison

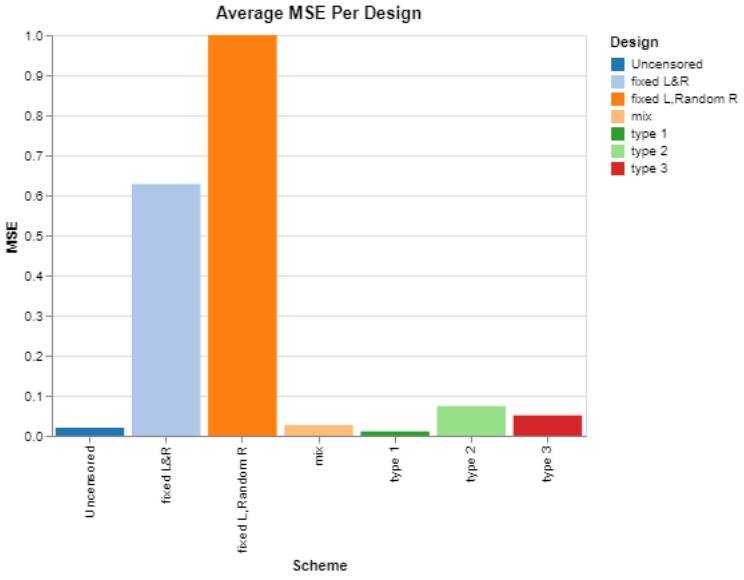

Figure 28: Overall MSE Comparison

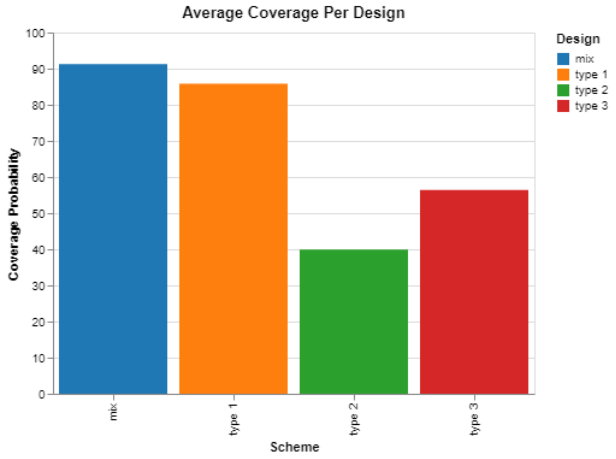

Figure 29: Selected Coverage Comparison

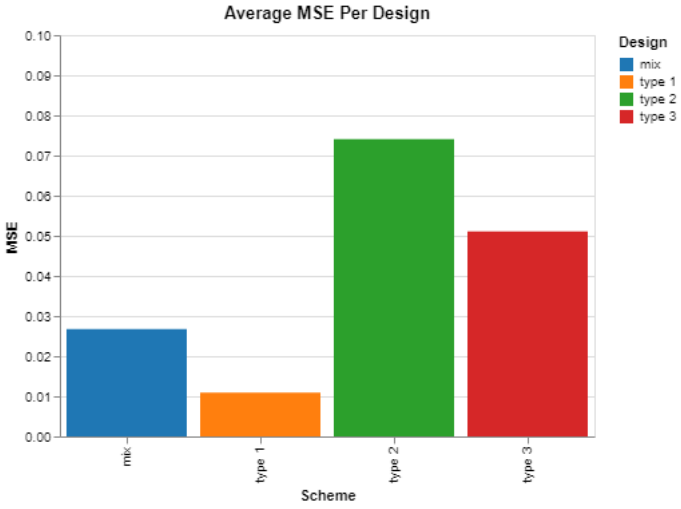

Figure 30: Selected MSE Comparison

5 Discussion and Conclusion

In carrying out experiments to measure the influence of temperature on technical devices, different design scenarios can be used or can occur.As a result, the uncertainty around the influence of temperature under these various designs or scenarios will vary. This paper looks at various possible designs and simulates sample data from these possible designs with different sample sizes (sample sizes ≤ 60), a model is fitted to the various data and the estimates derived are compared with the true population estimate of temperature. The estimate for the temperature effect from the population was known to be 0.98, therefore comparisons are geared around these population value to find out which design provides the closest estimate to the population value. Seven different designs were considered as follows; the uncensored design, the type I right censoring design, the type II right censoring design, type III right censoring design, mix interval, interval censored design where the lower and right intervals are fixed, lower is fixed and right random and finally lower and right are random. Proper description of all these various scenarios can be found in section 2.3 of this paper. The simulation of the time to events for the various technical device is from the Weibull distribution with shape and scale parameters given from the population. The Peck's model is used to describe the relationship between higher temperatures and survival times of this machines. Furthermore, fitting the AFT model provides a descriptive effect in terms of acceleration factor which better describes the effect of temperature on survival times of these devices.

Table 2, 3, 4, 5, 6 and 7, provides the maximum likelihood estimates of the effect of temperature from fitting the model to a simulated sample datasets under the various designs/scenarios and the uncertainty surrounding the estimate of temperature is reported. Under each scenario and for a particular sample size the estimate of temperature is provided along side the bias, MSE, standard error, variance and coverage. Also, these MLE are corrected using the RBA as discussed in section 2.5. In reporting the findings of the estimate in the results section above only the adjusted MLE are used in making comparisons. However, the un-adjusted estimates can be found in the various tables and the values do not differ so much from the adjusted.

In discussing the derived results, the coverage, MSE and bias are mostly looked into. In order to verify if the value of the true population parameters lie within 95% confidence interval given a sample's estimate, the coverage probability is used for over 1000 samples. This ensures intervals produced are independent of a particular sample. Another key aspect in evaluating the temperature effect from this various designs is the precision. High precision (low MSE) provides accurate estimation of the estimate of interest. In order to account for precision during the estimation procedure, the MSE is used which is a composed of bias and variance.

Generally, each design provided a similar trend in terms of MSE and coverage, with increasing MSE and decreasing coverage as the sample size reduces but in case of the interval designs smaller sample sizes rather produced larger coverage and the largest MSE. This behaviour of the estimates can be seen as a consequence of not knowing the exact time of failure of this devices in the interval censored case. This lack of knowledge leads to larger uncertainty around the estimation of the temperature effect which leads to larger confidence intervals and larger MSE. As a result, the engineers have to interpret results in the interval design with lots of caution and also consider the high amount of uncertainty surrounding the estimates in the case of interval censored data. In terms of MSE (figure 28), the interval designs provide the highest, while the type I design provides the least.

Figure 32 illustrates that the type II tends to over-estimate the coefficient of temperature from the population but the other designs generally tend to under-estimate the coefficient of temperature from the population. Figure 27 also shows the overall coverage probability of these various designs. Figure 27 points out that the fixed lower and random right design will produce the most intervals covering the true population value however figure 28 points out that this is done with a very large MSE. Another practical issue with fitting these models is the convergence of the maximum likelihood algorithm and the computational run-time especially in cases of highly censored data with smaller sizes. This convergence difficulties are often encountered since the maximum likelihood method entails the use of the pdf of observed events. Also, convergence difficulties are mostly noticed in smaller sample sizes with higher percentages of censoring especially for interval designs. The interval censored estimation procedure suffered this difficulties since exact event times are not known,

Overall, these various designs provide different estimates with different amounts of variability under the simulated datasets. The results from the various sections compared each design separately in terms of bias, MSE and coverage. However, given these results it is recommended that the engineers should always take into account the amount of variability surrounding the estimates simulated from the various designs. Knowledge of this uncertainty is fundamental in making the appropriate decisions with regards to the effect of temperature under this designs and especially in smaller sample sizes. A large coverage probability for a particular design or sample size implies that the true population value will be highly contained within confidence intervals obtained from that design. Also, a low MSE which is a function of bias and variability illustrates the precision and accuracy of the estimator in estimating this coefficient.

Using both criteria of coverage and MSE, the type I design can be recommended as the designs which provides the most coverage of the true population value with the smallest MSE. Since the type I design often requires engineers to pre-specify a particular censoring time which sometimes is not easily determined. Another practical design which comes in handy is random censoring (type III, with moderately large coverage and not too large MSE) an the mixed interval (with some eventtimes known only within intervals). The mixed interval also produces fairly good coverage (about 92%) with lower MSE. In terms of cost effectiveness, the type II can be adopted although its coverage is a little smaller than that of others, a low value of MSE shows it is quite precised and accurate. An ideal situation involves a case where the investigator observes all the event times (uncensored design) which tends to be rare. However, the uncensored design demonstrates a high coverage and small MSE but not smaller than that of type I design. For the interval design where they are very few observed observations the MSE is very large, an alternative method will be to describe the uncertainty based on prior information, implying the use of Bayesian techniques will be appropriate .

This study focused on Weibull distributed eventtimes modelled through the Peck's model where the humidity is kept constant. Future studies could study the interest of the effect of humidity and temperature together under the various designs and sample sizes. Another issue is various scenarios occur in the industrial setting and as a result not all could be captured through out this thesis. However industrial scenarios could generally be grouped into right censored or interval censored. This implies the results gotten from this study can serve as pointers in cases where other designs come up which have some aspects of right censoring or interval censoring.

This study provided insights into the uncertainty surrounding the measurement of the effect of temperature in different samples under various industrial design settings (types of censoring). The true population effect of temperature was known and as such various factors of the samples such as sample size, and percentage of censoring were varied to find out which design better measures the temperature influence. Therefore, the simulated datasets have provided knowledge on the uncertainty surrounding this temperature estimate when considering the various designs mentioned above. As a of results, quantifying this uncertainty engineers will be able to adopt the most effective designs.

6 Appendix

	Sample Size = 60					Sample Size = 40					Sample Size=20			
Censoring	05%	10%	30%	50%	75%	05%	10%	30%	50%	75%	05%	10%	30%	50%
Estimate	0.9809	0.9812	0.981	0.9807	0.9807	0.9767	0.9774	0.9791	0.9769	0.9854	0.9788	0.9802	0.9813	0.9856
Variance	0.0027	0.0025	0.0019	0.0017	0.0017	0.0041	0.0037	0.0028	0.0024	0.0025	0.0074	0.0067	0.0051	0.0049
Std error	0.052	0.0494	0.0432	0.0401	0.0391	0.0634	0.0601	0.0521	0.0479	0.045	0.0845	0.08	0.0689	0.0634
Bias	0.00009	0.0012	0.001	0.0007	0.0007	-0.0033	-0.0026	-0.0009	-0.0031	0.0054	-0.0012	0.0002	0.0013	0.0056
MSE	0.006	0.0056	0.0049	0.0047	0.0061	0.0091	0.0085	0.0112	0.007	0.0212	0.0168	0.0156	0.0131	0.0176
Coverage	92.3	91.5	87.8	83	74.5	92.3	91.4	85.1	82.2	72	89.8	89.6	84.1	78.4
Adj.estimate	0.9563	0.9565	0.9564	0.9561	0.9561	0.9522	0.9528	0.9546	0.9524	0.9607	0.9542	0.9556	0.9567	0.9609
Adj.variance	0.0028	0.0026	0.002	0.0017	0.0018	0.0042	0.0038	0.0029	0.0025	0.0026	0.0077	0.0069	0.0053	0.005
Adj.Std error	0.0529	0.0502	0.0439	0.0408	0.0398	0.0645	0.0611	0.053	0.0487	0.0457	0.086	0.0814	0.07	0.0645
Adj.Bias	-0.0237	-0.0235	-0.0236	-0.0239	-0.0239	-0.0278	-0.0272	-0.0254	-0.0276	-0.0193	-0.0258	-0.0244	-0.0233	-0.0191
Adj.MSE	0.0065	0.006	0.0053	0.0052	0.0065	0.0097	0.0091	0.0115	0.0076	0.0207	0.0173	0.016	0.0134	0.0175
Adj.coverage	92.5	91.9	88.7	83.2	75.2	92.5	92	85.8	82.8	72.3	90.2	90	84.9	79.2

Table 3: Estimates For Type I Right Censored Data

	Sample Size = 60				Sample Size = 40					Sample Size=20				
Censoring	05%	10%	20%	30%	05%	10%	20%	30%	50%	05%	10%	20%	30%	50%
Estimate	1.064	1.0926	1.1648	1.2527	1.0597	1.1022	1.1747	1.2829	1.4683	1.0495	1.0771	1.1876	1.2772	1.4972
Variance	0.0025	0.0029	0.0038	0.0049	0.0036	0.0044	0.0057	0.0077	0.0236	0.0073	0.0084	0.0123	0.0156	0.0559
Std error	0.0498	0.0535	0.0613	0.0695	0.0595	0.0659	0.075	0.0867	0.1514	0.0844	0.0903	0.1093	0.122	0.2294
Bias	0.084	0.1126	0.1848	0.2727	0.0797	0.1222	0.1947	0.3029	0.4883	0.0695	0.0971	0.2076	0.2972	0.5172
MSE	0.013	0.0191	0.0418	0.0838	0.0151	0.0247	0.0496	0.107	0.2756	0.0223	0.0283	0.068	0.1196	0.3535
Coverage	58.4	45	13.6	0.8	69.9	55.1	26.1	4.5	0.7	82.5	79	54.3	29.7	27.1
Adj.estimate	1.0373	1.0652	1.1356	1.2212	1.0331	1.0745	1.1452	1.2506	1.4315	1.0231	1.0501	1.1578	1.2451	1.4596
Adj.variance	0.0026	0.003	0.0039	0.0051	0.0037	0.0046	0.0059	0.008	0.0244	0.0076	0.0087	0.0127	0.0161	0.0578
Adj.Std error	0.0506	0.0544	0.0624	0.0707	0.0605	0.067	0.0762	0.0882	0.154	0.0858	0.0919	0.1111	0.1241	0.2333
Adj.Bias	0.0573	0.0852	0.1556	0.2412	0.0531	0.0945	0.1652	0.2706	0.4515	0.0431	0.0701	0.1778	0.2651	0.4796
Adj.MSE	0.0091	0.0136	0.0318	0.0676	0.0114	0.0186	0.0389	0.0885	0.2411	0.0191	0.0236	0.0563	0.1013	0.3165
Adj.coverage	59.8	45.8	14	1	71.1	55.6	27.4	4.7047	1	83	79.6	55.6	30.6	30.1

Table 4: Estimates For Type II Right Censored Data

	Sample Size =60					**Sample Size = 40**					**Sample Size = 20**			
Censoring %	5%	20%	31%	47%	75%	5%	20%	31%	47%	60%	05%	20%	31%	47%
Estimate	0.9556	0.8778	0.8245	0.7553	0.7247	0.9532	0.8776	0.8258	0.7577	0.7264	0.9484	0.8718	0.8188	0.746
Variance	0.003	0.0035	0.0042	0.0063	0.0084	0.0045	0.0052	0.0063	0.0095	0.0131	0.008	0.0094	0.0117	0.0194
Std error	0.0542	0.0583	0.0641	0.0781	0.0893	0.0665	0.0712	0.078	0.0946	0.1089	0.0877	0.0943	0.1042	0.1297
Bias	-0.0244	-0.1022	-0.1555	-0.2247	-0.2553	-0.0268	-0.1024	-0.1542	-0.2223	-0.2536	-0.0316	-0.1082	-0.1612	-0.234
MSE	0.0069	0.0177	0.0329	0.0631	0.0821	0.0102	0.0216	0.0376	0.0686	0.0908	0.0189	0.0326	0.052	0.0949
Coverage Prob	90.6	58.7	29.9	13.5	12	91.8	70.4	50.3	33.8	32.2322	89.6	77	66.9	59.0
Adj.Estimate	0.9316	0.8557	0.8038	0.7363	0.7065	0.9292	0.8556	0.805	0.7387	0.7082	0.9246	0.8499	0.7982	0.7273
Adj.Variance	0.0031	0.0036	0.0043	0.0066	0.0087	0.0047	0.0054	0.0065	0.0098	0.0135	0.0083	0.0097	0.0121	0.0201
Adj.Std error	0.0551	0.0593	0.0652	0.0794	0.0908	0.0676	0.0724	0.0793	0.0962	0.1108	0.0892	0.0959	0.1059	0.1319
Adj.Bias	-0.0484	-0.1243	-0.1762	-0.2437	-0.2735	-0.0508	-0.1244	-0.175	-0.2413	-0.2718	-0.0554	-0.1301	-0.1818	-0.2527
Adj.MSE	0.0086	0.0226	0.0397	0.0719	0.0916	0.012	0.0265	0.0442	0.0772	0.1002	0.0208	0.0375	0.0588	0.1036
Adj.Coverage Prob	91.2	60	31.7	14	12	92.3	71.3	52.5	35.5	33.1	90	78	68	60.6

Table 5: Estimates For Type III Right Censored Data

	Sample Size =60				**Sample Size = 40**					**Sample Size = 20**			
Censoring %	5%	31%	65%	82%	5%	20%	40%	61%	73%	5%	31%	47%	65%
Estimate	0.9833	0.9844	0.9915	1.0135	0.9799	0.9812	0.9823	0.9876	1.0005	0.9765	0.9841	0.9927	1.0256
Variance	0.0032	0.0045	0.0091	0.0197	0.0049	0.0054	0.008	0.0126	0.0212	0.0088	0.0126	0.0173	0.0404
Std error	0.0566	0.0662	0.0934	0.1321	0.0694	0.0727	0.0877	0.108	0.1322	0.092	0.1087	0.126	0.1672
Bias	0.0033	0.0044	0.0115	0.0335	-1e-04	0.0012	0.0023	0.0076	0.0205	-0.0035	0.0041	0.0127	0.0456
MSE	0.0067	0.0094	0.0198	0.0453	0.0102	0.0113	0.0167	0.0272	0.0451	0.0189	0.0282	0.0408	0.0853
Coverage Prob	93.8	93.2	92.9	88.6	93.2	92.8	92.3	91.2	91	90.8	89.9	88.8	85
Adj.Estimate	0.9586	0.9597	0.9666	0.988	0.9553	0.9565	0.9577	0.9628	0.9754	0.952	0.9594	0.9678	0.9998
Adj.Variance	0.0034	0.0046	0.0095	0.0204	0.0051	0.0056	0.0083	0.013	0.0219	0.0091	0.013	0.0179	0.0418
Adj.Std error	0.0576	0.0673	0.095	0.1344	0.0706	0.0739	0.0892	0.1099	0.1344	0.0936	0.1106	0.1281	0.1701
Adj.Bias	-0.0214	-0.0203	-0.0134	0.008	-0.0247	-0.0235	-0.0223	-0.0172	-0.0046	-0.028	-0.0206	-0.0122	0.0198
Adj.MSE	0.0071	0.0097	0.0196	0.0437	0.0107	0.0118	0.017	0.0272	0.0442	0.0195	0.0283	0.0402	0.0829
Adj.Coverage Prob	94.5	93.9	93.3	89.1	93.5	93.3	92.8	91.6	91.4	91.4	90.3	89.1	85.2

Table 6: Estimates for Mixed Interval Data

	Fixed Intervals			**Fixed Lower, Random Upper**			**Random Lower, Random Upper**		
	Sample Size			Sample Size			Sample Size		
	20	40	60	20	40	60	20	40	60
Estimate	0.6241	0.6099	0.5827	1.0271	0.946	0.9335	0.6988	0.5836	0.546
Variance	1.476×10^{80}	3.246×10^{79}	2.144×10^{78}	0.3838	0.1393	0.0879	6.5537	0.0339	0.0074
Std error	8.115×10^{38}	3.795×10^{38}	6.296×10^{37}	0.5199	0.3668	0.2914	0.2429	0.0305	0.0164
Bias	-0.3559	-0.3701	-0.3973	0.0471	-0.034	-0.0465	-0.2812	-0.3964	-0.434
MSE	1.476×10^{80}	3.246×10^{79}	2.144×10^{78}	0.4887	0.282	0.2526	6.7552	0.229	0.2117
Coverage	100	99.5	57	98.2	93.2	76.1	0.2	0.3	0.5
Adj.estimate	0.6085	0.5946	0.5681	1.0013	0.9222	0.9101	0.6813	0.5689	0.9538
Adj.variance	1.526×10^{78}	3.358×10^{79}	2.218×10^{79}	0.397	0.1441	0.0909	6.7792	0.0351	0.0051
Adj.Std error	8.253×10^{38}	3.859×10^{38}	6.403×10^{37}	0.5288	0.3731	0.2964	0.247	0.0311	0.0166
Adj.Bias	-0.3715	-0.3854	-0.4119	0.0213	-0.0578	-0.0699	-0.2987	-0.4111	-0.4477
Adj.MSE	1.526×10^{80}	3.358×10^{79}	2.218×10^{78}	0.495	0.282	0.2503	6.9848	0.2401	0.2227
Adj.coverage	100	99.5	57	98.1	97	76.1	7	1	0

Table 7: Estimates for Interval Censored Data

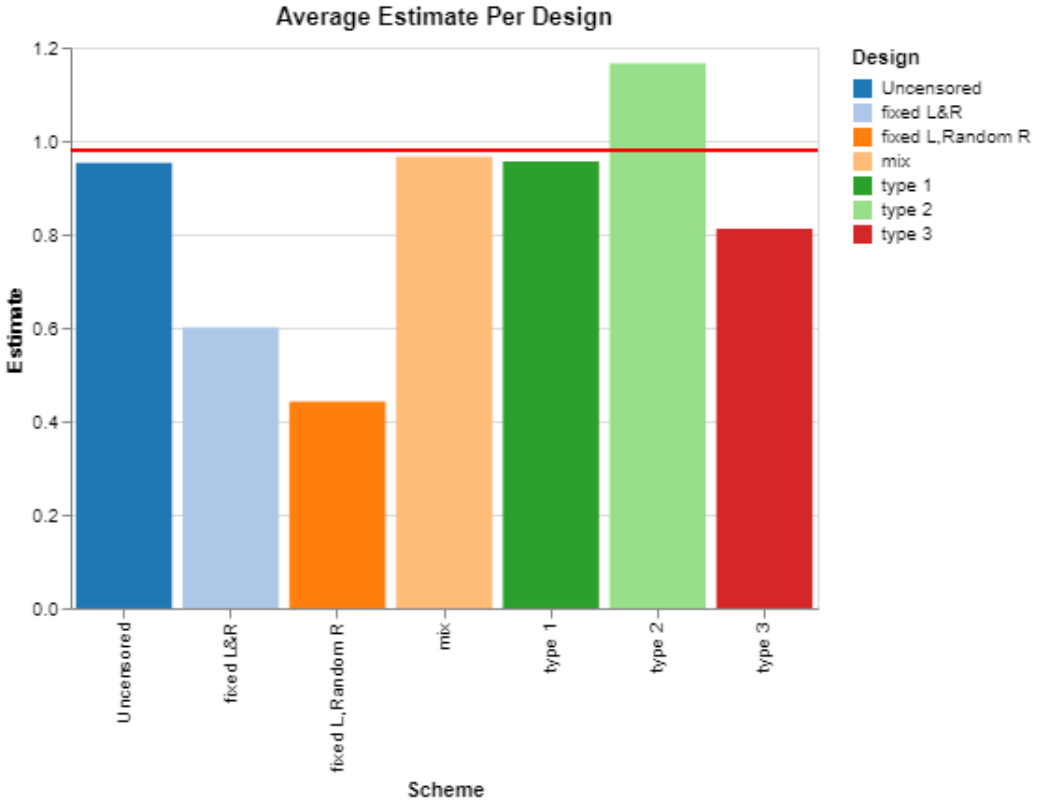

Figure 31: Overall Estimate Comparison With Population Estimate

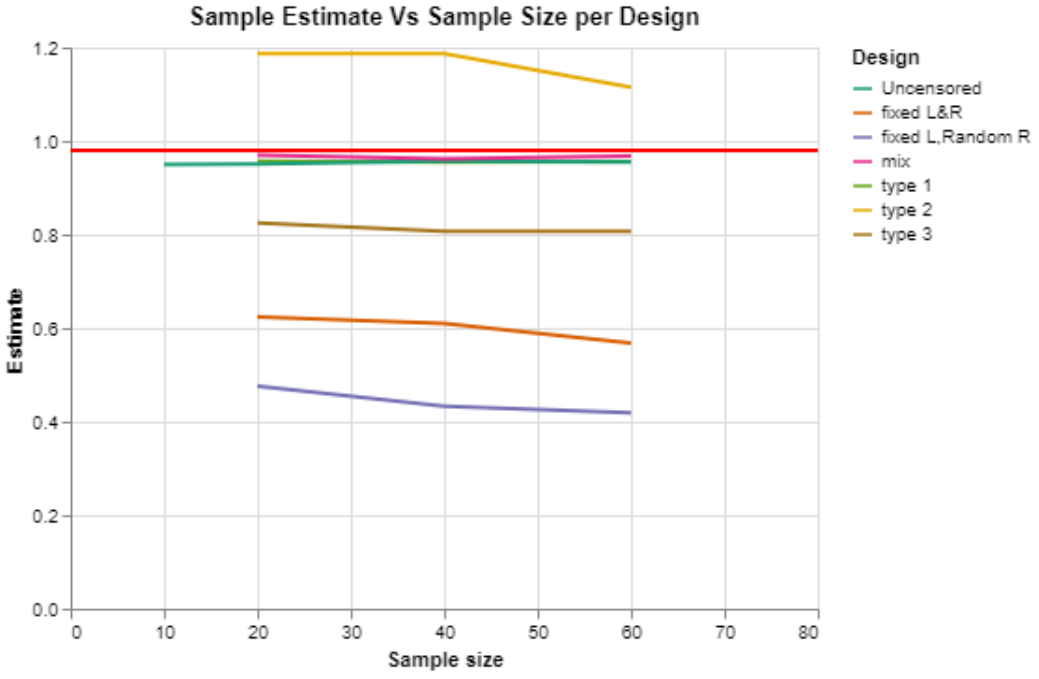

Figure 32: Overall Estimate vs Sample size per Design Type

References

[1] Bijma, Fetsje, Marianne Jonker, and Aad van der Vaart. *An Introduction to Mathematical Statistics.* Amsterdam University Press, Amsterdam, 2017.

[2] B. Qi, Y. Sun, W. Hu and X. Ding, *"A multi-stress Accelerated Life Tests method for Smart Electricity Meter based upon the Life-Stress Model,"* The Proceedings of 2011 9th International Conference on Reliability, Maintainability and Safety, 2011, pp. 1136-1140, doi: 10.1109/ICRMS.2011.5979441.

[3] Chan, P.S., Ng, H.K.T. Su, F *Exact likelihood inference for the two-parameter exponential distribution under Type-II progressively hybrid censoring.* Metrika 78, 747–770 (2015). https://doi.org/10.1007/s00184-014-0525-5

[4] Doganaksoy, Necip, and Josef Schmee. *"Practical Aspects of Corrected Likelihood Ratio Confidence Intervals: A Discussion of Jeng-Meeker and Wong-Wu."* Technometrics, vol. 42, no. 2, 2000, pp. 156–159. JSTOR, www.jstor.org/stable/1271447. Accessed 25 May 2021.

[5] Escobar, L., Meeker, W. (2006). *A Review of Accelerated Test Models.* Statistical Science, 21(4), 552-577. Retrieved May 25, 2021, from http://www.jstor.org/stable/27645794

[6] Fritz Scholz, *Inference for the Weibull Distribution*, Stat 498B Industrial Statistics, May 22, 2008, http://faculty.washington.edu/fscholz/DATAFILES498B2008/WeibullBounds.pdf

[7] Greg Caswell. *Temperature and Humidity Acceleration Factors on MLV Lifetime With and Without DC Bias*, DfR Solutions. https://www.dfrsolutions.com. 9000 Virginia Manor Rd Ste 290, Beltsville MD 20705

[8] Genschel,Ulrike and Meeker,William Q.(2010) *A Comparison of Maximum Likelihood and Median Rank Regression for Weibull Estimation*(vol 22) ,Journal of Quality engineering. Department of Statistics, Iowa State University, Ames, IA 50011. isbn=0898-2112,236-255(2010)

[9] Hilary Term. *lecture5:Confidence intervals.*Descriptive Statistics for Research.Institute for the Advancement of University Learning Department of Statistics,2002.

[10] Jeng, S. L. and Meeker W.Q. (2000). Comparisons of Weibull Distribution Approximate Confidence Intervals Procedures for Type I Censored Data. Technometrics 42, 135-148.

[11] Kleinbaum, David G., and Mitchel Klein. *Survival Analysis: A Self-Learning Text.* vol. 2005: 3, Springer, New York, 2005;2006;

[12] Monika Hebeisen, *Estimation from Interval-censored Time-to-event Data: Method Comparison by Simulation based on GALLIUM Study for Follicular Lymphoma*, University of Zurich, October 2014. http//:https://www.math.uzh.ch/li/index.php?filekey1=30882.

[13] Moore, Dirk F. *Applied Survival Analysis using R.* vol. 2016: 2, Springer International Publishing, Cham, 2016.

[14] Nelson, W.B., *Applied life data analysis*, New York: John Wiley Sons, 1985.isbn=9780471094586;0471094587;

[15] Ng H.K.T.,Wang Z. *Statistical estimation for the parameters of Weibull distribution based on progressively type-I interval censored sample*, Journal of Statistical Computation and Simulation, 79-2 (2009) 145-159.

[16] Pham, Hoang. Springer Handbook of Engineering Statistics. Springer, 2006.

[17] Nora E. Rosenberg, Leah Sirkus, *Survival Analysis Using SAS: A Practical Guide*. Second Edition By Paul D. Allison, American Journal of Epidemiology, Volume 174, Issue 4, 15 August 2011, Pages 503–504, https://doi.org/10.1093/aje/kwr202

[18] Winterbottom, Alan. Journal of the Royal Statistical Society. Series A (Statistics in Society), vol. 160, no. 2, 1997, pp. 367–368. JSTOR, www.jstor.org/stable/2983228. Accessed 25 May 2021.

[19] Xiang Jia, Dong Wang, Ping Jiang, Bo Guo, *Inference on the reliability of Weibull distribution with multiply Type-I censored data,Reliability Engineering System Safety*, Volume 150, 2016, Pages 171-181, ISSN 0951-8320, https://doi.org/10.1016/j.ress.2016.01.025.

```
##-------------------UNCENSORED-----------------------------#
##--------------------------------------------------------#
##This code generates 1000 uncensored datasets of size n and aft model is fitted.
by varying nt75,nt85,nt95, the total number of observations
in a dataset changes
nt75=20;nt85=20;nt95=20; n= nt75 + nt85 + nt95
#### population parameters
Kb <- 8.617333262 * 10^-5;alpha = 3.02 ;alpha0 = -31.39;alpha1 = 0.98
## predictor
x <-1/(Kb*(273.15+c(rep(75,nt75),rep(85,nt85),rep(95,nt95))))
######generating 1000 uncensored datasets
set.seed(1000)
uncens2 <- lapply(1:1000,
               function(ign) data.frame(eventtime <- rweibull(n,scale=exp(-31.39+0.98*x),shape=3.02),
                                        status = rep(1,n),
                                        predictor = x))
## rename some variables
uncens <- lapply(uncens2,
function(x) {names(x)[names(x)== 'eventtime....rweibull.n..scale...exp..31.39...0.98...x...
shape...3.02.'] <- 'eventtime'; x})
### fitting the model to the list of datasets
uncens_mod <- lapply(uncens, function(un) survreg(Surv(eventtime,status) ~ predictor, data = un,
dist = "weibull",control = list(iter.max=500)))
### extract the coefficients
coef <- purrr::map(uncens_mod, function(x){purrr::pluck(x, 'coefficients')[[2]]})
coef <- unlist(coef)
### extracting variance of coefficients
new_dat <- purrr::map(uncens_mod, function(x) {purrr::pluck(x, 'var')[[5]]})
var <- unlist(new_dat)
### standard errors
std_err <-  purrr:: map(new_dat, sqrt)
std_err <- unlist(std_err)
## Bias of estiamte
bias2 <- purrr::map(coef, function(x) {x - 0.98})
bias <- unlist(bias2)
## MSE = variance + squared bias
mse <- purrr::map2(new_dat,bias2, function(x,y) {x + y^2})
mse <- unlist(mse)
############ REDUCED BIAS ADJUSTMENT
###### here the variance is multiplied by the correction factor(RBA)
### From the weibull handbook C_4 for 60 events = (0.995772)^6 = 0.974898
```

```
## adjusted coefficients
est_adj <- purrr::map(coef, function(x) {x * 0.974898})
est_adj <- unlist(est_adj)
### adjusted unbiased estimate of variance
RBA_sig <- (sqrt(60/(60-1))/0.974898)
sigma_adj <-  purrr:: map(new_dat, function(x){var_adj <- x * RBA_sig})
sigma_adj <- unlist(sigma_adj)
### adjusted standard error of coefficients
stder_adj <-  purrr:: map(sigma_adj, function(x){sqrt(x)})
stderr_adj <- unlist(stder_adj)
### adjusted bias
bias2_adj <- purrr::map(est_adj, function(x) {x - 0.98})
bias_adj <- unlist(bias2_adj)
### adjusted mean square error
## MSE = adjusted variance + adjusted squared bias
mse2_adj <- purrr::map2(sigma_adj,bias2_adj, function(x,y) {x + y^2})
mse_adj <- unlist(mse2_adj)
#### creating a dataset for all interested parameters
type <- data.frame(coef,var,std_err,bias,mse,est_adj,sigma_adj,stderr_adj,bias_adj,mse_adj)
####### confidence intervals
lcl <-  purrr::map2(type$coef,type$std_err, function(x,y) {x + qnorm(0.025) * y})
ucl <-  purrr::map2(type$coef,type$std_err, function(x,y) {x + qnorm(0.975) * y})
#####
lcl <- unlist(lcl,use.names = FALSE)
ucl <- unlist(ucl, use.names = FALSE)
int <- data.frame(lcl,ucl)
#### coverage
countx <- function(x){(x$lcl < 0.98) & (0.98 < x$ucl)}
rcens <- as.list(table(countx(int)))
cov_rcens <- (rcens[['TRUE']]/nrow(type))* 100
####### adjusted intervals
lcl_adj <- purrr::map2(type$coef,type$stderr_adj, function(x,y) {x - 1.96 * y})
ucl_adj <- purrr::map2(type$coef,type$stderr_adj, function(x,y) {x + 1.96 * y})
lcl_adj <- unlist(lcl_adj,use.names = FALSE)
ucl_adj <- unlist(ucl_adj, use.names = FALSE)
int_adj <- data.frame(lcl_adj,ucl_adj)
####adjusted coverage
countx_adj <- function(x){(x$lcl_adj < 0.98) & (0.98 < x$ucl_adj)}
rcens_adj <- as.list(table(countx_adj(int_adj)))
cov_rcens_adj <- (rcens_adj[['TRUE']]/nrow(type))* 100
#########
```

```
print(paste0("sample size: ",n))
print(paste0("Estimate: ", round(mean(type$coef),digits=4)))
print(paste0("variance: ",round(mean(type$var),digits=4)))
print(paste0("standard error: ", round(mean(type$std_err),digits=4)))
print(paste0("Bias: ", round(mean(type$bias),digits=4)))
print(paste0("Mean squrare erorr: ",round(mean(type$mse),digits=4)))
print(paste0("coverage probability: ", round(cov_rcens,digits = 4)))
print(paste0("adjusted estimate: ", round(mean(type$est_adj),digits=4)))
print(paste0("adjusted variance: ", round(mean(type$sigma_adj),digits=4)))
print(paste0("adjusted standard error: ", round(mean(type$stderr_adj),digits=4)))
print(paste0("Adjusted Bias: ", round(mean(type$bias_adj),digits=4)))
print(paste0("Adjusted Mean squrare erorr: ", round(mean(type$mse_adj),digits=4)))
print(paste0("adjusted coverage probabilities: ", round(cov_rcens_adj,digits = 4)))
##------------------TYPE 1 RIGHT CENSORED--------------------#
##-----------------------------------------------------------#
##This code generates 1000 Type 1 right censored datasets of size n and aft model is fitted.
by varying nt75,nt85,nt95, the total number of observations
in a dataset changes
nt75=20;nt85=20;nt95=20; n= nt75 + nt85 + nt95
#### population parameters
Kb <- 8.617333262 * 10^-5;alpha = 3.02 ;alpha0 = -31.39;alpha1 = 0.98
## predictor
x <-1/(Kb*(273.15+c(rep(75,nt75),rep(85,nt85),rep(95,nt95))))
################generating 1000 type 1 right censored datasets##########################################
set.seed(1000)
###
typ1 <- lapply(1:1000,
               function(ign) data.frame(eventtime <- rweibull(n,scale=exp(-31.39+0.98*x),shape=3.02),
                                        t_c <- qweibull(rate,  scale=exp(-31.39+0.98*x),shape=3.02),
                                        status <- if_else(t_c < eventtime,1,0),
                                        predictor))
## rename variable list names
type1 <- lapply(typ1,
                function(x) {names(x)[names(x) == 'eventtime....rweibull.n..scale...exp..
                31.39...0.98...x...shape...3.02.'] <-'eventtime'; x
                names(x)[names(x) == 'status....if_else.t_c...eventtime..1..0.'] <- 'status'; x})
#
type1_mod <- lapply(type1, function(ty1) survreg(Surv(eventtime,status) ~ predictor, data = ty1, dist = "weibull",
control = list(iter.max=500)))
### extract the coefficients
coef1 <- purrr::map(type1_mod, function(x){purrr::pluck(x, 'coefficients')[[2]]})
```

```
coef1 <- unlist(coef1)
### extracting variance of coefficients
new_dat1 <- purrr::map(type1_mod, function(x) {purrr::pluck(x, 'var')[[5]]})
var1 <- unlist(new_dat1)
#### stamdard error
std_err1 <-  purrr:: map(new_dat1, sqrt)
std_err1 <- unlist(std_err1)
## Bias
bias21 <- purrr::map(coef1, function(x) {x - 0.98})
bias1 <- unlist(bias21)
## MSE = variance + squared bias
mse1 <- purrr::map2(new_dat1,bias21, function(x,y) {x + y^2})
mse1 <- unlist(mse1)
######## REDUCED BIAS ADJUSTMENT
####### here the variance is multiplied by the correction factor(RBA)
### From the weibull handbook C_4 for 60 events = (0.995772)^6 = 0.974898
## overall adjusted mean
est_adj1 <- purrr::map(coef1, function(x) {x * 0.974898})
est_adj1 <- unlist(est_adj1)
### adjusted unbiased estimate of variance
RBA_sig <- (sqrt(60/(60-1))/0.974898)
sigma_adj1 <-  purrr:: map(new_dat1, function(x){var_adj <- x * RBA_sig})
sigma_adj1 <- unlist(sigma_adj1)
### adjusted standard error of coefficients
stder_adj1 <-  purrr:: map(sigma_adj1, function(x){sqrt(x)})
stderr_adj1 <- unlist(stder_adj1)
### adjusted bias
bias2_adj1 <- purrr::map(est_adj1, function(x) {x - 0.98})
bias_adj1 <- unlist(bias2_adj1)
### adjusted mean square error
## MSE = adjusted variance + adjusted squared bias
mse2_adj1 <- purrr::map2(sigma_adj1,bias2_adj1, function(x,y) {x + y^2})
mse_adj1 <- unlist(mse2_adj1)
#### creating a dataset fro all interested parameters
type1 <- data.frame(coef1,var1,std_err1,bias1,mse1,est_adj1,sigma_adj1,stderr_adj1,bias_adj1,mse_adj1)
type1 <- na.omit(type1)
## selecting models that converged
type1 <- subset(type1, (coef1<5 & coef1>0))
######## confidence intervals
lcl <-  purrr::map2(type1$coef1,type1$std_err1, function(x,y) {x + qnorm(0.025) * y})
ucl <-  purrr::map2(type1$coef1,type1$std_err1, function(x,y) {x + qnorm(0.975) * y})
```

```
#####
lcl <- unlist(lcl,use.names = FALSE)
ucl <- unlist(ucl, use.names = FALSE)
int <- data.frame(lcl,ucl)
###### coverage
countx <- function(x){(x$lcl < 0.98) & (0.98 < x$ucl)}
rcens <- as.list(table(countx(int)))
cov_rcens <- (rcens[['TRUE']]/nrow(type1))* 100
####### adjusted intervals
lcl_adj <-  purrr::map2(type1$coef1,type1$stderr_adj1, function(x,y) {x - 1.96 * y})
ucl_adj <-  purrr::map2(type1$coef1,type1$stderr_adj1, function(x,y) {x + 1.96 * y})
####
lcl_adj <- unlist(lcl_adj,use.names = FALSE)
ucl_adj <- unlist(ucl_adj, use.names = FALSE)
int_adj <- data.frame(lcl_adj,ucl_adj)
#### adjusted coverage
countx_adj <- function(x){(x$lcl_adj < 0.98) & (0.98 < x$ucl_adj)}
rcens_adj <- as.list(table(countx_adj(int_adj)))
cov_rcens_adj <- (rcens_adj[['TRUE']]/nrow(type1))* 100
##########################################
print(paste0("sample size: ",n))
print(paste0("Percentage of censoring: ",rate*100))
print(paste0("Estimate: ", round(mean(type1$coef1),digits=4)))
print(paste0("variance: ",round(mean(type1$var1),digits=4)))
print(paste0("standard error: ", round(mean(type1$std_err1),digits=4)))
print(paste0("Bias: ", round(mean(type1$bias1),digits=4)))
print(paste0("Mean squrare erorr: ",round(mean(type1$mse1),digits=4)))
print(paste0("coverage probability: ", round(cov_rcens,digits = 4)))
print(paste0("adjusted estimate: ", round(mean(type1$est_adj1),digits=4)))
print(paste0("adjusted variance: ", round(mean(type1$sigma_adj1),digits=4)))
print(paste0("adjusted standard error: ", round(mean(type1$stderr_adj1),digits=4)))
print(paste0("Adjusted Bias: ", round(mean(type1$bias_adj1),digits=4)))
print(paste0("Adjusted Mean squrare erorr: ", round(mean(type1$mse_adj1),digits=4)))
print(paste0("adjusted coverage probabilities: ", round(cov_rcens_adj,digits = 4)))
##-------------------TYPE 2 RIGHT CENSORED--------------------#
##------------------------------------------------------------#
#################generating 1000 type 2 right censored datasets###########################################
n=nt75 + nt85 + nt95
cens_n=11 ### number of censored observations
r= n-cens_n ### required sample
```

```
set.seed(1000)
###
typ2 <- lapply(1:1000,
               function(ign) data.frame(z <- rweibull(n,scale=exp(-31.39+0.98*x),shape=3.02),
                                        eventtime <- sort(z),
                                        status <- c(rep(1,n-cens_n),rep(0,cens_n)),
                                        predictor <- sort(x)))

##rename variable names in list
type2 <- lapply(typ2,
                function(x) {names(x)[names(x) == 'eventtime....sort.z.'] <- 'eventtime'; x
                names(x)[names(x) == 'status....c.rep.1..n...cens_n...rep.0..cens_n..'] <- 'status'; x
                names(x)[names(x) == 'predictor....sort.x.'] <- 'predictor'; x})
### fit the model to the list
type2_mod <- lapply(type2, function(ty2) survreg(Surv(eventtime,status) ~ predictor, data = ty2, dist = "weibull"))
### extract the coefficients
coef2 <- purrr::map(type2_mod, function(x){purrr::pluck(x, 'coefficients')[[2]]})
coef2 <- unlist(coef2)
### extracting variance
new_dat2 <- purrr::map(type2_mod, function(x) {purrr::pluck(x, 'var')[[5]]})
var2 <- unlist(new_dat2)
#### standard errors
std_err2 <-  purrr:: map(new_dat2, sqrt)
std_err2 <- unlist(std_err2)
## Bias
bias22 <- purrr::map(coef2, function(x) {x - 0.98})
bias2 <- unlist(bias22)
## MSE = variance + squared bias
mse2 <- purrr::map2(new_dat2,bias22, function(x,y) {x + y^2})
mse2 <- unlist(mse2)
###################### REDUCED BIAS ADJUSTMENT
###### here the variance is multiplied by the correction factor(RBA)
### From the weibull handbook C_4 for 60 events = (0.995772)^6 = 0.974898
## overall adjusted mean
est_adj2 <- purrr::map(coef2, function(x) {x * 0.974898})
est_adj2 <- unlist(est_adj2)
### adjusted unbiased estimate of variance
RBA_sig <- (sqrt(60/(60-1))/0.974898)
sigma_adj2 <-  purrr:: map(new_dat2, function(x){var_adj <- x * RBA_sig})
sigma_adj2 <- unlist(sigma_adj2)
### adjusted standard error of coefficients
```

```
stder_adj2 <-  purrr:: map(sigma_adj2, function(x){sqrt(x)})
stderr_adj2 <- unlist(stder_adj2)
### adjusted bias
bias2_adj2 <- purrr::map(est_adj2, function(x) {x - 0.98})
bias_adj2 <- unlist(bias2_adj2)
### adjusted mean square error
## MSE = adjusted variance + adjusted squared bias
mse2_adj2 <- purrr::map2(sigma_adj2,bias2_adj2, function(x,y) {x + y^2})
mse_adj2 <- unlist(mse2_adj2)
#### creating a dataset fro all interested parameters
type2 <- data.frame(coef2,var2,std_err2,bias2,mse2,est_adj2,sigma_adj2,stderr_adj2,bias_adj2,mse_adj2)
type2 <- na.omit(type2)
type2 <- subset(type2, (coef2<5 & coef2>0))
######## confidence intervals
lcl <-  purrr::map2(type2$coef2,type2$std_err2, function(x,y) {x + qnorm(0.025) * y})
ucl <-  purrr::map2(type2$coef2,type2$std_err2, function(x,y) {x + qnorm(0.975) * y})
####
lcl <- unlist(lcl,use.names = FALSE)
ucl <- unlist(ucl, use.names = FALSE)
int <- data.frame(lcl,ucl)
##### coverage
countx <- function(x){(x$lcl < 0.98) & (0.98 < x$ucl)}
rcens <- as.list(table(countx(int)))
cov_rcens <- (rcens[['TRUE']]/nrow(type2))* 100
######## adjusted intervals
lcl_adj <-  purrr::map2(type2$coef2,type2$stderr_adj2, function(x,y) {x - 1.96 * y})
ucl_adj <-  purrr::map2(type2$coef2,type2$stderr_adj2, function(x,y) {x + 1.96 * y})
#####
lcl_adj <- unlist(lcl_adj,use.names = FALSE)
ucl_adj <- unlist(ucl_adj, use.names = FALSE)
int_adj <- data.frame(lcl_adj,ucl_adj)
##### adjusted coverage
countx_adj <- function(x){(x$lcl_adj < 0.98) & (0.98 < x$ucl_adj)}
rcens_adj <- as.list(table(countx_adj(int_adj)))
cov_rcens_adj <- (rcens_adj[['TRUE']]/nrow(type2))* 100
#####
print(paste0("sample size: ",n))
print(paste0("Percentage of censoring: ",(cens_n/n)*100))
print(paste0("Estimate: ", round(mean(type2$coef2),digits=4)))
print(paste0("variance: ",round(mean(type2$var2),digits=4)))
print(paste0("standard error: ", round(mean(type2$std_err2),digits=4)))
```

```
print(paste0("Bias: ", round(mean(type2$bias2),digits=4)))
print(paste0("Mean squrare erorr: ",round(mean(type2$mse2),digits=4)))
print(paste0("coverage probability: ", round(cov_rcens,digits = 4)))
print(paste0("adjusted estimate: ", round(mean(type2$est_adj2),digits=4)))
print(paste0("adjusted variance: ", round(mean(type2$sigma_adj2),digits=4)))
print(paste0("adjusted standard error: ", round(mean(type2$stderr_adj2),digits=4)))
print(paste0("Adjusted Bias: ", round(mean(type2$bias_adj2),digits=4)))
print(paste0("Adjusted Mean squrare erorr: ", round(mean(type2$mse_adj2),digits=4)))
print(paste0("adjusted coverage probabilities: ", round(cov_rcens_adj,digits = 4)))
##------------------TYPE 3 RIGHT CENSORED--------------------#
##-----------------------------------------------------------#
set.seed(1000)
typ3 <- lapply(1:1000,
               function(ign) data.frame(eventtime <- rweibull(n, scale=exp(alpha0 + alpha1*x), shape=alpha),
                                        cens_time <- runif(n,min = 0.1,max = 3),
                                        status <- if_else(cens_time < eventtime,1,0),
                                        predictor <- 1/(Kb*(273.15+c(rep(75,nt75),rep(85,nt85),rep(95,nt95))))
                                        ))
## rename variables
type3 <- lapply(typ3,
                function(x) {
names(x)[names(x) == 'eventtime....rweibull.n..scale...exp.alpha0...alpha1...x...shape...alpha.'] <-'eventtime'; x
names(x)[names(x) == 'status....if_else.cens_time...eventtime..1..0.'] <- 'status'; x
names(x)[names(x) == 'predictor....1..Kb....273.15...c.rep.75..nt75...rep.85..nt85...'] <- 'predictor';x})
#### status to calculate percentage of censoring
sta <- purrr::map(type3 , function(x){
  purrr::pluck(x, 'status')})
a <- unlist(sta)
b <- table(a)
rate <- (b["0"])/(n*1000) * 100
### fit the model to the list
type3_mod <- lapply(type3, function(ty3) survreg(Surv(eventtime,status) ~ predictor, data = ty3, dist = "weibull",
control = list(iter.max=50000,rel.tolerance=1e-09),init = c(-8.5,0.15,-0.11)))
### extract the coefficients
coef3 <- purrr::map(type3_mod, function(x){purrr::pluck(x, 'coefficients')[[2]]})
coef3 <- unlist(coef3)
### extracting variance of coefficients
new_dat3 <- purrr::map(type3_mod, function(x) {purrr::pluck(x, 'var')[[5]]})
var3 <- unlist(new_dat3)
#### standard errors
std_err3 <-  purrr:: map(new_dat3, sqrt)
```

```
std_err3 <- unlist(std_err3)
## Bias
bias23 <- purrr::map(coef3, function(x) {x - 0.98})
bias3 <- unlist(bias23)
## mean square error
## MSE = variance + squared bias
mse3 <- purrr::map2(new_dat3,bias23, function(x,y) {x + y^2})
mse3 <- unlist(mse3)
######## REDUCED BIAS ADJUSTMENT
###### here the variance is multiplied by the correction factor(RBA)
### From the weibull handbook C_4 for 60 events = (0.995772)^6 = 0.974898
## overall adjusted mean
est_adj3 <- purrr::map(coef3, function(x) {x * 0.974898})
est_adj3 <- unlist(est_adj3)
### adjusted unbiased estimate of variance
RBA_sig <- (sqrt(60/(60-1))/0.974898)
sigma_adj3 <-  purrr:: map(new_dat3, function(x){var_adj <- x * RBA_sig})
sigma_adj3 <- unlist(sigma_adj3)
### adjusted standard error of coefficients
stder_adj3 <-  purrr:: map(sigma_adj3, function(x){sqrt(x)})
stderr_adj3 <- unlist(stder_adj3)
### adjusted bias
bias2_adj3 <- purrr::map(est_adj3, function(x) {x - 0.98})
bias_adj3 <- unlist(bias2_adj3)
### adjusted mean square error
## MSE = adjusted variance + adjusted squared bias
mse2_adj3 <- purrr::map2(sigma_adj3,bias2_adj3, function(x,y) {x + y^2})
mse_adj3 <- unlist(mse2_adj3)
#### creating a dataset for all interested parameters
type3 <- data.frame(coef3,var3,std_err3,bias3,mse3,est_adj3,sigma_adj3,stderr_adj3,bias_adj3,mse_adj3)
type3 <- na.omit(type3)
type3 <- subset(type3, (coef3<5 & coef3>0)& (var3>0) & (var3<1.058094e+01))
####### intervals
lcl <-  purrr::map2(type3$coef3,type3$std_err3, function(x,y) {x + qnorm(0.025) * y})
ucl <-  purrr::map2(type3$coef3,type3$std_err3, function(x,y) {x + qnorm(0.975) * y})
######
lcl <- unlist(lcl,use.names = FALSE)
ucl <- unlist(ucl, use.names = FALSE)
int <- data.frame(lcl,ucl)
#######
countx <- function(x){(x$lcl < 0.98) & (0.98 < x$ucl)}
```

```
rcens <- as.list(table(countx(int)))
cov_rcens <- (rcens[['TRUE']]/nrow(type3))* 100
####### adjusted intervals
lcl_adj <-  purrr::map2(type3$coef3,type3$stderr_adj3, function(x,y) {x - 1.96 * y})
ucl_adj <-  purrr::map2(type3$coef3,type3$stderr_adj3, function(x,y) {x + 1.96 * y})
######
lcl_adj <- unlist(lcl_adj,use.names = FALSE)
ucl_adj <- unlist(ucl_adj, use.names = FALSE)
int_adj <- data.frame(lcl_adj,ucl_adj)
###### asjusted coverage
countx_adj <- function(x){(x$lcl_adj < 0.98) & (0.98 < x$ucl_adj)}
rcens_adj <- as.list(table(countx_adj(int_adj)))
cov_rcens_adj <- (rcens_adj[['TRUE']]/nrow(type3))* 100
##############################################
print(paste0("sample size: ",n))
print(paste0("Percentage of censoring: ",rate))
print(paste0("Estimate: ", round(mean(type3$coef3),digits=4)))
print(paste0("variance: ",round(mean(type3$var3),digits=4)))
print(paste0("standard error: ", round(mean(type3$std_err3),digits=4)))
print(paste0("Bias: ", round(mean(type3$bias3),digits=4)))
print(paste0("Mean squrare erorr: ",round(mean(type3$mse3),digits=4)))
print(paste0("coverage probability: ", round(cov_rcens,digits = 4)))
print(paste0("adjusted estimate: ", round(mean(type3$est_adj3),digits=4)))
print(paste0("adjusted variance: ", round(mean(type3$sigma_adj3),digits=4)))
print(paste0("adjusted standard error: ", round(mean(type3$stderr_adj3),digits=4)))
print(paste0("Adjusted Bias: ", round(mean(type3$bias_adj3),digits=4)))
print(paste0("Adjusted Mean squrare erorr: ", round(mean(type3$mse_adj3),digits=4)))
print(paste0("adjusted coverage probabilities: ", round(cov_rcens_adj,digits = 4)))

##------------------MIXED INTERVAL --------------------#
##-------------------------------------------------------------#
set.seed(1000)
cens8 <- lapply(1:1000,
        function(ign) data.frame(eventtime <- rweibull(n,scale=exp(-31.39+0.98*x),shape=3.02),
        t_c <- runif(n,min = 0.1,max = 9),
        status <- if_else(t_c < eventtime,1,0),
        L <- ifelse(status == 1,eventtime,0.1),
        R <- ifelse(status == 0,t_c,eventtime),
        predictor <- 1/(Kb*(273.15+c(rep(75,nt75),rep(85,nt85),rep(95,nt95))))))
## rename elements
cens_int <- lapply(cens8,
```

```
              function(x) {names(x)[names(x) == 'L....ifelse.status....1..eventtime..0.1.'] <- 'L'; x
    names(x)[names(x) == 'R....ifelse.status....0..t_c..eventtime.'] <- 'R'; x
    names(x)[names(x) == 'status....if_else.t_c...eventtime..1..0.'] <- 'status'; x
    names(x)[names(x) == 'predictor....1..Kb....273.15...c.rep.75..nt75...rep.85..nt85...'] <- 'predictor'; x})
#### calculate perentage of censoring
sta <- purrr::map(cens_int , function(x){purrr::pluck(x, 'status')})
a <- unlist(sta)
b <- table(a)
rate <- (b["0"])/(n*1000) * 100
### fit model
control = list(iter.max=50,rel.tolerance=1e-09),init=c(-10,0.5,-0.3)
test8_model <- lapply(cens_int, function(aft8){
  survreg( Surv(time=L,time2=R,type = "interval2") ~ predictor,data = aft8,dist = "weibull")})
### extract the coefficients
coef8 <- purrr::map(test8_model,function(x){purrr::pluck(x,'coefficients')[[2]]})
coef8 <- unlist(coef8)
### extracting variance of coefficients
new_dat8 <- purrr::map(test8_model, function(x) {purrr::pluck(x,'var')[[5]]})
var8 <- unlist(new_dat8)
#### standard errors
std_err8 <-  purrr:: map(new_dat8, sqrt)
std_err8 <- unlist(std_err8)
## Bias
bias28 <- purrr::map(coef8, function(x) {x - 0.98})
bias8 <- unlist(bias28)
## MSE = variance + squared bias
mse8 <- purrr::map2(new_dat8,bias28, function(x,y) {x + y^2})
mse8 <- unlist(mse8)
###################### REDUCED BIAS ADJUSTMENT
###### here the variance is multiplied by the correction factor(RBA)
### From the weibull handbook C_4 for 60 events = (0.995772)^6 = 0.974898
## overall adjusted mean
est_adj8 <- purrr::map(coef8, function(x) {x * 0.974898})
est_adj8 <- unlist(est_adj8)
### adjusted unbiased estimate of variance
RBA_sig <- (sqrt(60/(60-1))/0.974898)
sigma_adj8 <-  purrr:: map(new_dat8, function(x){var_adj <- x * RBA_sig})
sigma_adj8 <- unlist(sigma_adj8)
### adjusted standard error of coefficients
stder_adj8 <-  purrr:: map(sigma_adj8, function(x){sqrt(x)})
stderr_adj8 <- unlist(stder_adj8)
```

```
### adjusted bias
bias2_adj8 <- purrr::map(est_adj8, function(x) {x - 0.98})
bias_adj8 <- unlist(bias2_adj8)
### adjusted mean square error
mse2_adj8 <- purrr::map2(sigma_adj8,bias2_adj8, function(x,y) {x + y^2})
mse_adj8 <- unlist(mse2_adj8)
#### creating a dataset fro all interested parameters
type8 <- data.frame(coef8,var8,std_err8,bias8,mse8,est_adj8,sigma_adj8,stderr_adj8,bias_adj8,mse_adj8)
type8 <- na.omit(type8)
type8 <- subset(type8,(coef8<5 & coef8>0) & (var8>0) & (var8<1.058094e+01))
####### intervals
lcl <-  purrr::map2(type8$coef8,type8$std_err8, function(x,y) {x + qnorm(0.025) * y})
ucl <-  purrr::map2(type8$coef8,type8$std_err8, function(x,y) {x + qnorm(0.975) * y})
###
lcl <- unlist(lcl,use.names = FALSE)
ucl <- unlist(ucl, use.names = FALSE)
int <- data.frame(lcl,ucl)
### coverage
countx <- function(x){(x$lcl < 0.98) & (0.98 < x$ucl)}
rcens <- as.list(table(countx(int)))
cov_rcens <- (rcens[['TRUE']]/nrow(type8))* 100
####### adjusted intervals
lcl_adj <-  purrr::map2(type8$coef8,type8$stderr_adj8, function(x,y){x - 1.96 * y})
ucl_adj <-  purrr::map2(type8$coef8,type8$stderr_adj8, function(x,y){x + 1.96 * y})
###
lcl_adj <- unlist(lcl_adj,use.names = FALSE)
ucl_adj <- unlist(ucl_adj, use.names = FALSE)
int_adj <- data.frame(lcl_adj,ucl_adj)
## adjusted coverage
countx_adj <- function(x){
  (x$lcl_adj < 0.98) & (0.98 < x$ucl_adj)}
rcens_adj <- as.list(table(countx_adj(int_adj)))
cov_rcens_adj <- (rcens_adj[['TRUE']]/nrow(type8))* 100
##########################################
print(paste0("sample size: ",n))
print(paste0("Percentage of censoring: ",rate))
print(paste0("Estimate: ", round(mean(type8$coef8),digits=4)))
print(paste0("variance: ",round(mean(type8$var8),digits=4)))
print(paste0("standard error: ", round(mean(type8$std_err8),digits=4)))
print(paste0("Bias: ", round(mean(type8$bias8),digits=4)))
print(paste0("Mean squrare erorr: ",round(mean(type8$mse8),digits=4)))
```

```
print(paste0("coverage probability: ", round(cov_rcens,digits = 4)))
print(paste0("adjusted estimate: ", round(mean(type8$est_adj8),digits=4)))
print(paste0("adjusted variance: ", round(mean(type8$sigma_adj8),digits=4)))
print(paste0("adjusted standard error: ", round(mean(type8$stderr_adj8),digits=4)))
print(paste0("Adjusted Bias: ", round(mean(type8$bias_adj8),digits=4)))
print(paste0("Adjusted Mean squrare erorr: ", round(mean(type8$mse_adj8),digits=4)))
print(paste0("adjusted coverage probabilities: ", round(cov_rcens_adj,digits = 4)))
##------------------FIXED LOWER FIXED RIGHT -----------------#
##-------------------------------------------------------------#
set.seed(1000)
######
flfr2 <- lapply(1:1000,
               function(inter) data.frame(
    grid75_l <- rep(0.1,n75),## temperature  75 will have short failure times so the grid is short
    grid75_u <- rep(6,n75),
    ## generate event times where temperature is 75
    event75 <- rweibull(n75, scale=exp(alpha0 + alpha1*((1)/(Kb*348.15))), shape=alpha),
    L75 <- if_else(grid75_l < event75 ,grid75_l,event75),
    R75 <- if_else(event75 < grid75_u ,grid75_u,event75),
    ## temperature  85 will have shorter failure times so the grid is shorter
    grid85_l <- rep(0.1,n85),
    grid85_u <- rep(2,n85),
    ## generate event times where temperature is 85
    event85 <- rweibull(n85, scale=exp(alpha0 + alpha1*((1)/(Kb*358.15))), shape=alpha),
    L85 <- if_else(grid85_l < event85 ,grid85_l,event85),
    R85 <- if_else(event85 < grid85_u ,grid85_u,event85),
    ## temperature  95 will have shortest failure times so the gird is the shortest
    grid95_l <- rep(0.1,n95),
    grid95_u <- rep(1.2,n95),
    ## generate event times where temperature is 85
    event95 <- rweibull(n95, scale=exp(alpha0 + alpha1*((1)/(Kb*368.15))), shape=alpha),
    L95 <- if_else(grid95_l < event95 ,grid95_l,event95),
    R95 <- if_else(event95 < grid95_u ,grid95_u,event95),
    ### combining various grids
    L <- c(L75,L85,L95),
    R <- c(R75,R85,R95),
    ####indicators
    status <- rep(3, each=n),
    ### predictor
    predictor <- 1/(Kb*(273.15+c(rep(75,n75),rep(85,n95),rep(95,n95))))))
## rename elements
```

```
flfr <- lapply(flfr2,
            function(x) {names(x)[names(x) == 'L....c.L75..L85..L95.'] <- 'L'; x
    names(x)[names(x) == 'R....c.R75..R85..R95.'] <- 'R'; x
    names(x)[names(x) == 'status....rep.3..each...n.'] <- 'status'; x
    names(x)[names(x) == 'predictor....1..Kb....273.15...c.rep.75..n75...rep.85..n95...'] <- 'predictor'; x})
## fit model
flfr_model <- lapply(flfr, function(aft3){
  survreg( Surv(time=L,time2=R,event=status,type = "interval") ~ predictor,data = aft3,dist = "weibull",
           control = list(iter.max=50,rel.tolerance=1e-09),init = c(-10,0.5,-0.3))})
### extract the coefficients
coef4 <- purrr::map(flfr_model, function(x){purrr::pluck(x, 'coefficients')[[2]]})
coef4 <- unlist(coef4)
### extracting variance of coefficients
new_dat4 <- purrr::map(flfr_model, function(x) {purrr::pluck(x, 'var')[[5]]})
var4 <- unlist(new_dat4)
#### standard error
std_err4 <-  purrr:: map(new_dat4, sqrt)
std_err4 <- unlist(std_err4)
## Bias
bias24 <- purrr::map(coef4, function(x) {x - 0.98})
bias4 <- unlist(bias24)
## mean square error
mse4 <- purrr::map2(new_dat4,bias24, function(x,y) {x + y^2})
mse4 <- unlist(mse4)
###################### REDUCED BIAS ADJUSTMENT
###### here the variance is multiplied by the correction factor(RBA)
### From the weibull handbook C_4 for 60 events = (0.995772)^6 = 0.974898
## overall adjusted mean
est_adj4 <- purrr::map(coef4, function(x) {x * 0.974898})
est_adj4 <- unlist(est_adj4)
### adjusted unbiased estimate of variance
RBA_sig <- (sqrt(60/(60-1))/0.974898)
sigma_adj4 <-  purrr:: map(new_dat4, function(x){var_adj <- x * RBA_sig})
sigma_adj4 <- unlist(sigma_adj4)
### adjusted standard error of coefficients
stder_adj4 <-  purrr:: map(sigma_adj4, function(x){sqrt(x)})
stderr_adj4 <- unlist(stder_adj4)
### adjusted bias
bias2_adj4 <- purrr::map(est_adj4, function(x) {x - 0.98})
bias_adj4 <- unlist(bias2_adj4)
### adjusted mean square error
```

```
## MSE = adjusted variance + adjusted squared bias
mse2_adj4 <- purrr::map2(sigma_adj4,bias2_adj4, function(x,y) {x + y^2})
mse_adj4 <- unlist(mse2_adj4)
#### creating a dataset fro all interested parameters
type4 <- data.frame(coef4,var4,std_err4,bias4,mse4,est_adj4,sigma_adj4,stderr_adj4,bias_adj4,mse_adj4)
type4 <- na.omit(type4)
####### confidence intervals
lcl <-  purrr::map2(type4$coef4,type4$std_err4, function(x,y) {
  x + qnorm(0.025) * y})
ucl <-  purrr::map2(type4$coef4,type4$std_err4, function(x,y) {
  x + qnorm(0.975) * y})
#####
lcl <- unlist(lcl,use.names = FALSE)
ucl <- unlist(ucl, use.names = FALSE)
int <- data.frame(lcl,ucl)
######coverage
countx <- function(x){(x$lcl < 0.98) & (0.98 < x$ucl)}
rcens <- as.list(table(countx(int)))
cov_rcens <- (rcens[['TRUE']]/nrow(type4))* 100
####### adjusted intervals
lcl_adj <-  purrr::map2(type4$coef4,type4$stderr_adj4, function(x,y) {x - 1.96 * y})
ucl_adj <-  purrr::map2(type4$coef4,type4$stderr_adj4, function(x,y) {x + 1.96 * y})
######
lcl_adj <- unlist(lcl_adj,use.names = FALSE)
ucl_adj <- unlist(ucl_adj, use.names = FALSE)
int_adj <- data.frame(lcl_adj,ucl_adj)
###### adjusted coverage
countx_adj <- function(x){
  (x$lcl_adj < 0.98) & (0.98 < x$ucl_adj)}
###
obs <- nrow(type4)
rcens_adj <- as.list(table(countx_adj(int_adj)))
cov_rcens_adj <- (rcens_adj[['TRUE']]/obs)* 100
##########################################
print(paste0("sample size: ",n))
print(paste0("Estimate: ", round(mean(type4$coef4),digits=4)))
print(paste0("variance: ",round(mean(type4$var4),digits=4)))
print(paste0("standard error: ", round(mean(type4$std_err4),digits=4)))
print(paste0("Bias: ", round(mean(type4$bias4),digits=4)))
print(paste0("Mean squrare erorr: ",round(mean(type4$mse4),digits=4)))
print(paste0("coverage probability: ", round(cov_rcens,digits = 4)))
```

```
print(paste0("adjusted estimate: ", round(mean(type4$est_adj4),digits=4)))
print(paste0("adjusted variance: ", round(mean(type4$sigma_adj4),digits=4)))
print(paste0("adjusted standard error: ", round(mean(type4$stderr_adj4),digits=4)))
print(paste0("Adjusted Bias: ", round(mean(type4$bias_adj4),digits=4)))
print(paste0("Adjusted Mean squrare erorr: ", round(mean(type4$mse_adj4),digits=4)))
print(paste0("adjusted coverage probabilities: ", round(cov_rcens_adj,digits = 4)))
##------------------FIXED LOWER Random RIGHT -----------------#
##----------------------------------------------------------#
set.seed(6000)
##
flrr2 <- lapply(1:1000,
               function(inter) data.frame(grid75_l <- rep(0.1,n75),
grid75_u <- runif(n75,5.5,6),
## generate event times where temperature is 75
event75 <- rweibull(n75, scale=exp(alpha0 + alpha1*((1)/(Kb*348.15))), shape=alpha),

 L75 <- if_else(grid75_l < event75 ,grid75_l,event75),
R75 <- if_else(event75 < grid75_u ,grid75_u,event75),
## temperature  85 will have shorter failure times so
 grid85_l <- rep(0.1,n85),
 grid85_u <- runif(n85,1.8,2),
 ## generate event times where temperature is 85
 event85 <- rweibull(n85, scale=exp(alpha0 + alpha1*((1)/(Kb*358.15))), shape=alpha),
 L85 <- if_else(grid85_l < event85 ,grid85_l,event85),
 R85 <- if_else(event85 < grid85_u ,grid85_u,event85),
 ## temperature  95 will have shortest failure times so
 grid95_l <- rep(0.1,n95),
  grid95_u <- runif(n95,0.9,1),
 ## generate event times where temperature is 85
  event95 <- rweibull(n95, scale=exp(alpha0 + alpha1*((1)/(Kb*368.15))), shape=alpha),
  L95 <- if_else(grid95_l < event95 ,grid95_l,event95),
  R95 <- if_else(event95 < grid95_u ,grid95_u,event95),
   ### combining various grids
  L <- c(L75,L85,L95),
  R <- c(R75,R85,R95),
  predictor <- 1/(Kb*(273.15+c(rep(75,n75),rep(85,n85),rep(95,n95))),
  status <- rep(3, each=n)))
## rename elements
flrr <- lapply(flrr2,
               function(x) {names(x)[names(x) == 'L....c.L75..L85..L95.'] <- 'L'; x
names(x)[names(x) == 'R....c.R75..R85..R95.'] <- 'R'; x
```

```
names(x)[names(x) == 'status....rep.3..each...n.'] <- 'status'; x
names(x)[names(x) == 'predictor....1..Kb....273.15...c.rep.75..n75...rep.85..n85...'] <- 'predictor'; x})
###fit model
flrr_model <- lapply(flrr, function(aft5){
  survreg( Surv(time=L,time2=R,event = status,type = "interval")~ predictor,data = aft5,dist = "weibull",
control = list(iter.max=500,rel.tolerance=1e-09),init = c(-8.5,0.15,-0.11))})
### extract the coefficients
coef5 <- purrr::map(flrr_model, function(x){purrr::pluck(x, 'coefficients')[[2]]})
coef5 <- unlist(coef5)
### extracting standard errors of coefficients
new_dat5 <- purrr::map(flrr_model, function(x) {
  purrr::pluck(x, 'var')[[5]]})
var5 <- unlist(new_dat5)
#### standard error
std_err5 <-  purrr:: map(new_dat5, sqrt)
std_err5 <- unlist(std_err5)
## Bias
bias25 <- purrr::map(coef5, function(x) {x - 0.98})
bias5 <- unlist(bias25)
## mean square error
## MSE = variance + squared bias
mse5 <- purrr::map2(new_dat5,bias25, function(x,y) {x + y^2})
mse5 <- unlist(mse5)
###################### REDUCED BIAS ADJUSTMENT
###### here the variance is multiplied by the correction factor(RBA)
### From the weibull handbook C_4 for 60 events = (0.995772)^6 = 0.974898
## adjusted estimate
est_adj5 <- purrr::map(coef5, function(x) {x * 0.974898})
est_adj5 <- unlist(est_adj5)
### adjusted unbiased estimate of variance
RBA_sig <- (sqrt(60/(60-1))/0.974898)
sigma_adj5 <-  purrr:: map(new_dat5, function(x){var_adj <- x * RBA_sig})
sigma_adj5 <- unlist(sigma_adj5)
### adjusted standard error of coefficients
stder_adj5 <-  purrr:: map(sigma_adj5, function(x){sqrt(x)})
stderr_adj5 <- unlist(stder_adj5)
### adjusted bias
bias2_adj5 <- purrr::map(est_adj5, function(x) {x - 0.98})
bias_adj5 <- unlist(bias2_adj5)
### adjusted mean square error
mse2_adj5 <- purrr::map2(sigma_adj5,bias2_adj5, function(x,y) {x + y^2})
```

```
mse_adj5 <- unlist(mse2_adj5)
#### creating a dataset fro all interested parameters
type5 <- data.frame(coef5,var5,std_err5,bias5,mse5,est_adj5,sigma_adj5,stderr_adj5,bias_adj5,mse_adj5)
type5 <- na.omit(type5)
####### intervals
lcl <-  purrr::map2(type5$coef5,type5$std_err5, function(x,y) {x + qnorm(0.025) * y})
ucl <-  purrr::map2(type5$coef5,type5$std_err5, function(x,y) {x + qnorm(0.975) * y})
####
lcl <- unlist(lcl,use.names = FALSE)
ucl <- unlist(ucl, use.names = FALSE)
int <- data.frame(lcl,ucl)
##### coverage
countx <- function(x){(x$lcl < 0.98) & (0.98 < x$ucl)}
rcens <- as.list(table(countx(int)))
cov_rcens <- (rcens[['TRUE']]/nrow(type5))* 100
####### adjusted intervals
lcl_adj <-  purrr::map2(type5$coef5,type5$stderr_adj5, function(x,y) {x - 1.96 * y})
ucl_adj <-  purrr::map2(type5$coef5,type5$stderr_adj5, function(x,y) {x + 1.96 * y})
####
lcl_adj <- unlist(lcl_adj,use.names = FALSE)
ucl_adj <- unlist(ucl_adj, use.names = FALSE)
int_adj <- data.frame(lcl_adj,ucl_adj)
##### adjusted coverage
countx_adj <- function(x){(x$lcl_adj < 0.98) & (0.98 < x$ucl_adj)}
rcens_adj <- as.list(table(countx_adj(int_adj)))
cov_rcens_adj <- (rcens_adj[['TRUE']]/nrow(type5))* 100
##########################################
print(paste0("sample size: ",n))
print(paste0("Estimate: ", round(mean(type5$coef5),digits=4)))
print(paste0("variance: ",round(mean(type5$var5),digits=4)))
print(paste0("standard error: ", round(mean(type5$std_err5),digits=4)))
print(paste0("Bias: ", round(mean(type5$bias5),digits=4)))
print(paste0("Mean squrare erorr: ",round(mean(type5$mse5),digits=4)))
print(paste0("coverage probability: ", round(cov_rcens,digits = 4)))
print(paste0("adjusted estimate: ", round(mean(type5$est_adj5),digits=4)))
print(paste0("adjusted variance: ", round(mean(type5$sigma_adj5),digits=4)))
print(paste0("adjusted standard error: ", round(mean(type5$stderr_adj5),digits=4)))
print(paste0("Adjusted Bias: ", round(mean(type5$bias_adj5),digits=4)))
print(paste0("Adjusted Mean squrare erorr: ", round(mean(type5$mse_adj5),digits=4)))
print(paste0("adjusted coverage probabilities: ", round(cov_rcens_adj,digits = 4)))
```

```
##------------------Random LOWER Random RIGHT ----------------#
##-------------------------------------------------------------#
set.seed(6000)
rlrr2 <- lapply(1:1000,
                function(inter) data.frame(#grid75_l = runif(20,0.1,0.3),
## temperature  75 will have longer failure times so
grid75_l <- runif(n75,0.8,1),
grid75_u <- runif(n75,5,5.9),
## generate event times where temperature is 75
event75 <- rweibull(n75, scale=exp(alpha0 + alpha1*((1)/(Kb*348.15))), shape=alpha),
L75 <- if_else(grid75_l < event75 ,grid75_l,event75),
R75 <- if_else(event75 < grid75_u ,grid75_u,event75),
## temperature  85 will have longer failure times so
grid85_l <- runif(n85,0.5,0.8),
grid85_u <- runif(n85,1.8,2.2),
## generate event times where temperature is 85
event85 <- rweibull(n85, scale=exp(alpha0 + alpha1*((1)/(Kb*358.15))), shape=alpha),
L85 <- if_else(grid85_l < event85 ,grid85_l,event85),
R85 <- if_else(event85 < grid85_u ,grid85_u,event85),
## temperature  95 will have longer failure times so
grid95_l <- runif(n95,0.1,0.3),
grid95_u <- runif(n95,0.9,1),
## generate event times where temperature is 85
event95 <- rweibull(n95, scale=exp(alpha0 + alpha1*((1)/(Kb*368.15))), shape=alpha),
L95 <- if_else(grid95_l < event95 ,grid95_l,event95),
R95 <- if_else(event95 < grid95_u ,grid95_u,event95),
### combining various grids
L <- c(L75,L85,L95),
R <- c(R75,R85,R95),
predictor <- 1/(Kb*(273.15+c(rep(75,n75),rep(85,n85),rep(95,n95)))),
status <- rep(3, each=n)))
## rename elements
rlrr <- lapply(rlrr2,
               function(x) {names(x)[names(x) == 'L....c.L75..L85..L95.'] <- 'L'; x
names(x)[names(x) == 'R....c.R75..R85..R95.'] <- 'R'; x
names(x)[names(x) == 'status....rep.3..each...n.'] <- 'status'; x
names(x)[names(x) == 'predictor....1..Kb....273.15...c.rep.75..n75...rep.85..n85...'] <- 'predictor'; x})
###fit model
rlrr_model <- lapply(rlrr, function(aft6){
  survreg( Surv(time=L,time2=R,event = status,type = "interval")~ predictor,data = aft6,dist = "weibull",
           control = list(iter.max=500,rel.tolerance=1e-09),init = c(-21,0.5,-0.5))})
```

```
### extract the coefficients
coef6 <- purrr::map(rlrr_model, function(x){purrr::pluck(x, 'coefficients')[[2]]})
coef6 <- unlist(coef6)
### extracting variance of coefficients
new_dat6 <- purrr::map(rlrr_model, function(x) {purrr::pluck(x, 'var')[[5]]})
var6 <- unlist(new_dat6)
#### standard errors
std_err6 <-  purrr:: map(new_dat6, sqrt)
std_err6 <- unlist(std_err6)
##Bias
bias26 <- purrr::map(coef6, function(x) {x - 0.98})
bias6 <- unlist(bias26)
## MSE = variance + squared bias
mse6 <- purrr::map2(new_dat6,bias26, function(x,y) {x + y^2})
mse6 <- unlist(mse6)
###################### REDUCED BIAS ADJUSTMENT
###### here the variance is multiplied by the correction factor(RBA)
### From the weibull handbook C_4 for 60 events = (0.995772)^6 = 0.974898
## overall adjusted mean
est_adj6 <- purrr::map(coef6, function(x) {x * 0.974898})
est_adj6 <- unlist(est_adj6)
### adjusted unbiased estimate of variance
RBA_sig <- (sqrt(60/(60-1))/0.974898)
sigma_adj6 <-  purrr:: map(new_dat6, function(x){var_adj <- x * RBA_sig})
sigma_adj6 <- unlist(sigma_adj6)
### adjusted standard error of coefficients
stder_adj6 <-  purrr:: map(sigma_adj6, function(x){sqrt(x)})
stderr_adj6 <- unlist(stder_adj6)
### adjusted bias
bias2_adj6 <- purrr::map(est_adj6, function(x) {x - 0.98})
bias_adj6 <- unlist(bias2_adj6)
### adjusted mean square error
mse2_adj6 <- purrr::map2(sigma_adj6,bias2_adj6, function(x,y) {x + y^2})
mse_adj6 <- unlist(mse2_adj6)
#### creating a dataset fro all interested parameters
type6 <- data.frame(coef6,var6,std_err6,bias6,mse6,est_adj6,sigma_adj6,stderr_adj6,bias_adj6,mse_adj6)
type6 <- na.omit(type6)
######## confidence intervals
lcl <-  purrr::map2(type6$coef6,type6$std_err6, function(x,y) {x + qnorm(0.025) * y})
ucl <-  purrr::map2(type6$coef6,type6$std_err6, function(x,y) {x + qnorm(0.975) * y})
###############
```

```
lcl <- unlist(lcl,use.names = FALSE)
ucl <- unlist(ucl, use.names = FALSE)
int <- data.frame(lcl,ucl)
######coverage
countx <- function(x){(x$lcl < 0.98) & (0.98 < x$ucl)}
rcens <- as.list(table(countx(int)))
cov_rcens <- (rcens[['TRUE']]/nrow(type6))* 100
####### adjusted intervals
lcl_adj <-  purrr::map2(type6$coef6,type6$stderr_adj6, function(x,y){x - 1.96 * y})
ucl_adj <-  purrr::map2(type6$coef6,type6$stderr_adj6, function(x,y) {x + 1.96 * y})
#######
lcl_adj <- unlist(lcl_adj,use.names = FALSE)
ucl_adj <- unlist(ucl_adj, use.names = FALSE)
int_adj <- data.frame(lcl_adj,ucl_adj)
######## adjusted coverage
countx_adj <- function(x){
  (x$lcl_adj < 0.98) & (0.98 < x$ucl_adj)}
rcens_adj <- as.list(table(countx_adj(int_adj)))
cov_rcens_adj <- (rcens_adj[['TRUE']]/nrow(type6))* 100
##########################################
print(paste0("sample size: ",n))
print(paste0("Estimate: ", round(mean(type6$coef6),digits=4)))
print(paste0("variance: ",round(mean(type6$var6),digits=4)))
print(paste0("standard error: ", round(mean(type6$std_err6),digits=4)))
print(paste0("Bias: ", round(mean(type6$bias6),digits=4)))
print(paste0("Mean squrare erorr: ",round(mean(type6$mse6),digits=4)))
print(paste0("coverage probability: ", round(cov_rcens,digits = 4)))
print(paste0("adjusted estimate: ", round(mean(type6$est_adj6),digits=4)))
print(paste0("adjusted variance: ", round(mean(type6$sigma_adj6),digits=4)))
print(paste0("adjusted standard error: ", round(mean(type6$stderr_adj6),digits=4)))
print(paste0("Adjusted Bias: ", round(mean(type6$bias_adj6),digits=4)))
print(paste0("Adjusted Mean squrare erorr: ", round(mean(type6$mse_adj6),digits=4)))
print(paste0("adjusted coverage probabilities: ", round(cov_rcens_adj)))
##------------------Plots-------------------------------#
##------------------------------------------------------------#
reticulate::conda_create("r-reticulate")
install.packages("altair")
library("altair")
###
###### type 1 plots
type1 <- subset(right, scheme=="type 1")
```

```
###coverage type 1
alt$Chart(type1)$mark_line(interpolate = "bundle")$encode(
    y = alt$Y("coverage:Q",title='Coverage',scale=alt$Scale(domain =list(65,100))),
    x = alt$X("censoring:Q",title='Percentage of censoring',scale=alt$Scale(domain =list(0,80))),
    color = alt$Color("size:N",title = "Sample Size"))$
  properties(title = "Coverage Vs Percentage of censoring by sample size",width=300,height=300)
##Bias type 1
alt$Chart(type1)$mark_line(interpolate = "bundle")$encode(
    x = alt$X("censoring:Q",title='Percentage of censoring',scale=alt$Scale(domain =list(0,80))),
    y = alt$Y("bias:Q",title='Bias',scale=alt$Scale(domain=list(-0.0178,-0.0291))),
    color = alt$Color("size:N",title = "Sample Size"))$
  properties(title = "Bias Vs Percentage of censoring by sample size",
             width=300,
             height=300)
###type 1 bias
alt$Chart(data = type1)$mark_bar()$encode(
    x = alt$X("size:N",title='Sample Size'),
    y = alt$Y("mean(mse):Q",title='MSE'),
    color = "size:N") $
  properties(title='MSE by sample size',width=300,height=300)
###### type 2 plots
type2 <- subset(right, scheme=="type 2")
###coverage
alt$Chart(type2)$mark_line(interpolate = "bundle")$encode(
    y = alt$Y("coverage:Q",title='Coverage',scale=alt$Scale(domain =list(0,85))),
    x = alt$X("censoring:Q",title='Percentage of censoring',scale=alt$Scale(domain =list(0,60))),
    color = alt$Color("size:N",title = "Sample Size"))$
  properties(title = "Coverage Vs Percentage of censoring by sample size", width=300,height=300)
##Bias
alt$Chart(type2)$mark_line(interpolate = "bundle")$encode(
    x = alt$X("censoring:Q",title='Percentage of censoring',scale=alt$Scale(domain =list(0,60))),
    y = alt$Y("bias:Q",title='Bias'),
    color = alt$Color("size:N",title = "Sample Size"))$
  properties(title = "Bias Vs Percentage of censoring by sample size",width=300,height=300)
###bias
alt$Chart(data = type2)$mark_bar()$encode(
    x = alt$X("size:N",title='Sample Size'),
    y = alt$Y("mean(mse):Q",title='MSE'),
    color = "size:N"
  ) $properties(title='MSE by sample size',width=300,height=300)
###### type 3 plots
```

```
type3 <- subset(join, scheme=="type 3")
###coverage
alt$Chart(type3)$mark_line(interpolate = "bundle")$encode(
    y = alt$Y("coverage:Q",title='Coverage',scale=alt$Scale(domain =list(0,95))),
    x = alt$X("censoring:Q",title='Percentage of censoring',scale=alt$Scale(domain =list(0,80))),
    color = alt$Color("size:N",title = "Sample Size"))$
  properties(title = "Coverage Vs Percentage of censoring Per sample size",width=300,height=300)
##Bias
alt$Chart(type3)$mark_line(interpolate = "bundle")$encode(
    x = alt$X("censoring:Q",title='Percentage of censoring',scale=alt$Scale(domain =list(0,80))),
    y = alt$Y("bias:Q",title='Bias'),
    color = alt$Color("size:N",title = "Sample Size"))$
    properties(title = "Bias Vs Percentage of censoring Per sample size", width=300,height=300)
### mse
alt$Chart(data = type3)$mark_bar()$encode(
    x = alt$X("size:N",title='Sample Size'),
    y = alt$Y("mean(mse):Q",title='MSE'),
    color = "size:N") $
    properties(title='MSE Per sample size',width=300,height=300)
###### mix interval plots
mix <- subset(join, scheme=="mix")
###coverage
alt$Chart(mix)$mark_line(interpolate = "bundle")$encode(
    y = alt$Y("coverage:Q",title='Coverage',scale=alt$Scale(domain =list(80,100))),
    x = alt$X("censoring:Q",title='Percentage of censoring',scale=alt$Scale(domain =list(0,90))),
    color = alt$Color("size:N",title = "Sample Size"))$
  properties(title = "Coverage Vs Percentage of censoring Per sample size",width=300, height=300)
##Bias
alt$Chart(mix)$
  mark_line(interpolate = "bundle")$encode(
    x = alt$X("censoring:Q",title='Percentage of censoring',scale=alt$Scale(domain =list(0,90))),
    y = alt$Y("bias:Q",title='Bias'),
    color = alt$Color("size:N",title = "Sample Size"))$
  properties(title = "Bias Vs Percentage of censoring Per sample size",width=300,height=300)
### mse
alt$Chart(data = mix)$mark_bar()$encode(
    x = alt$X("size:N",title='Sample Size'),
    y = alt$Y("mean(mse):Q",title='MSE'),
    color = "size:N")$
  properties(title='MSE Per sample size',width=300,height=300)
###### overall plots
```

```
### coverage
right$cov <- rep(95,nrow(right))
###
alt$Chart(right)$mark_line(interpolate = "bundle")$encode(
    x = alt$X("censoring:Q",title='Percentage of censoring',scale=alt$Scale(domain =list(0,80))),
    y = alt$Y("mean(coverage):Q",title='Coverage Probability'),
    color = alt$Color("scheme:N",title = "Type",scale=alt$Scale(scheme ='accent'))$
  properties(title = "Coverage Vs Percentage of censoring ",
             width=300,
             height=300)
###mse
alt$Chart(right)$
  mark_line(interpolate = "bundle")$
  encode(
    x = alt$X("censoring:Q",title='Percentage of censoring',scale=alt$Scale(domain =list(0,80))),
    y = alt$Y("mean(mse):Q",title='MSE'),
    color = alt$Color("scheme:N",title = "Type",scale=alt$Scale(scheme ='accent'))
  )$
  properties(title = "MSE Vs Percentage of censoring",
             width=300,
             height=300)
###bias
alt$Chart(right)$mark_line(interpolate = "bundle")$encode(
    x = alt$X("censoring:Q",title='Percentage of censoring',scale=alt$Scale(domain =list(0,80))),
    y = alt$Y("mean(bias):Q",title='Bias'),
    color = alt$Color("scheme:N",title = "Type",scale=alt$Scale(scheme ='accent')))$
    properties(title = "Bias Vs Percentage of censoring",
             width=300,
             height=300)
######################################## INTERVAL################
select <-subset(interval,(scheme=="fixed L&R"|scheme=="fixed L,Random R"))
###coverage
alt$Chart(select)$mark_line(interpolate = "bundle")$encode(
    x = alt$X("size:Q",title='Sample size',scale=alt$Scale(domain =list(0,80))),
    y = alt$Y("mean(coverage):Q",title='Coverage Probability'),
    color = alt$Color("scheme:N",title = "Type",scale=alt$Scale(scheme ='dark2')))$
  properties(title = "Coverage Vs Sample Size ",width=300,height=300)
###mse
alt$Chart(select)$mark_line()$encode(
    x = alt$X("size:Q",title='Sample size',scale=alt$Scale(domain =list(0,80))),
    y = alt$Y("mean(mse2):Q",title='MSE'),
```

```
    color = alt$Color("scheme:N",title = "Type",scale=alt$Scale(scheme ='dark2'))$
  properties(title = "MSE Vs Sample Size",width=300,height=300)
###bias
alt$Chart(select)$mark_line(interpolate = "bundle")$encode(
    x = alt$X("size:Q",title='Sample Size',scale=alt$Scale(domain =list(0,80))),
    y = alt$Y("mean(bias):Q",title='Bias'),
    color = alt$Color("scheme:N",title = "Type",scale=alt$Scale(scheme ='dark2'))$
  properties(title = "Bias Vs Sample Size",width=300,height=300)
#######comparing all the schemes
### Coverage
right$cov <- rep(95,nrow(right))
#
c <- alt$Chart(join)$mark_bar(interpolate = "bundle")$encode(
    x = alt$X("scheme:N",title='Scheme'),
    y = alt$Y("mean(coverage):Q",title='Coverage Probability'),
    color = alt$Color("scheme:N",title = "Design",scale=alt$Scale(scheme ='category20'))$
    properties(title = "Average Coverage Per Design",width=400,height=300)
rule <- alt$Chart(right)$mark_rule(color = "red")$encode(y = alt$Y("cov:Q"))
c + rule
### mse
alt$Chart(join)$mark_bar(interpolate = "bundle")$encode(
    x = alt$X("scheme:N",title='Scheme'),
    y = alt$Y("mean(mse):Q",title='MSE',scale=alt$Scale(domain = list(0,1),clamp = TRUE)),
    color = alt$Color("scheme:N",title = "Design",scale=alt$Scale(scheme ='category20'))$
properties(title = "Average MSE Per Design",width=400,height=300)
### estimate
join$est <- rep(0.98,nrow(join))
es <- alt$Chart(join)$
  mark_bar()$encode(
    x = alt$X("scheme:N",title='Scheme'),
    y = alt$Y("mean(estimate):Q",title='Estimate'),
    color = alt$Color("scheme:N",title = "Design",scale=alt$Scale(scheme ='category20'))
  )$properties(title = "Average Estimate Per Design",width=400,height=300)
rule <-
  alt$Chart(join)$mark_rule(color = "red")$encode(y = alt$Y("est:Q"))
es + rule
### coverage, sample size, type
alt$Chart(join)$
  mark_line()$
  encode(
    x = alt$X("size:Q",title='Sample size',scale=alt$Scale(domain =list(0,80))),
```

```
    y = alt$Y("mean(coverage):Q",title='Coverage'),
    color = alt$Color("scheme:N",title = "Design",scale=alt$Scale(scheme ='dark2')))$properties(title = "Coverage Vs Sa
#########estimae,sample size
es <- alt$Chart(join)$
 mark_line()$encode(
    x = alt$X("size:Q",title='Sample size',scale=alt$Scale(domain =list(0,80))),
    y = alt$Y("mean(estimate):Q",title='Estimate'),
    color = alt$Color("scheme:N",title = "Design",scale=alt$Scale(scheme ='dark2')))$
  properties(title = "Sample Estimate Vs Sample Size per Design",width=400,height=300)
rule <- alt$Chart(join)$mark_rule(color = "red")$encode(
    y = alt$Y("est:Q"))
es + rule
######## selecting all except the fixed and random intervals
top <-subset(join,(scheme=="type 1"|scheme=="type 2"|scheme=="type 3"|scheme=="uncensored"|scheme=="mix"))
############comparing
### Coverage
top$cov <- rep(95,nrow(top))
#
c <- alt$Chart(top)$mark_bar(interpolate = "bundle")$encode(
    x = alt$X("scheme:N",title='Scheme'),
    y = alt$Y("mean(coverage):Q",title='Coverage Probability'),
    color = alt$Color("scheme:N",title = "Design",scale=alt$Scale(scheme ='category10')))$
    properties(title = "Average Coverage Per Design",width=400,height=300)
rule <-alt$Chart(top)$mark_rule(color = "red")$encode(y = alt$Y("cov:Q"))
c + rule
### mse
alt$Chart(top)$mark_bar(interpolate = "bundle")$encode(
    x = alt$X("scheme:N",title='Scheme'),
    y = alt$Y("mean(mse):Q",title='MSE',scale=alt$Scale(domain = list(0,0.1),clamp = TRUE)),
    color = alt$Color("scheme:N",title = "Design",scale=alt$Scale(scheme ='category10')))$
  properties(title = "Average MSE Per Design",width=400,height=300)
```