February 2020

Bayesian Reliability Analysis of The Power Law Process and Statistical Modeling of Computer and Network Vulnerabilities with Cybersecurity Application

Freeh N. Alenezi
University of South Florida

Bayesian Reliability Analysis of The Power Law Process and Statistical Modeling of Computer and Network Vulnerabilities with Cybersecurity Application

by

Freeh N. Alenezi

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
Department of Mathematics & Statistics
College of Arts and Sciences
University of South Florida

Major Professor: Chris P. Tsokos, Ph.D.
Kandethody M. Ramachandran, Ph.D.
Lu Lu, Ph.D.
Zhuo Lu, Ph.D.

Date of Approval:
March 3, 2020

Keywords: Ordinary Bayes, Non-Homogeneous Poisson Process, Loss function, operating systems, Cyber

Dedication

To my beloved family, especially
To my late father:
Naif Alenezi.
To my mother:
Moneerah Alenezi.

Acknowledgments

Above all, I offer my genuine and heartfelt gratitude to my major advisor Professor Chris P. Tsokos, who inspired me with his endless support, profound advice, and invaluable guidance. Without his patience and encouragement I could not have developed such a depth of new knowledge and made application of it in this dissertation. He has given me a breadth and depth of knowledge that will allow me to continue my work.

I am also truly grateful to the members of my dissertation committee: Professors Kanadethody M. Ramachandran, Lu Lu, and Zhuo Lu for their time and kind assistance during my dissertation preparation. Specifically, Dr. Ramachandran who is one of the most well-prepared, accurate, and fair classroom professors I have met and whose method of teaching research principles and ideas added positively to my educational experience, Dr. Lu Lu whose fundamental application courses taught me that science is more than theory, and Dr. Zhuo Lu who imparted data security and cryptography knowledge that has been instrumental in developing a portion of my thinking for not only part of this dissertation, but also for future work. I would like to express my sincere thanks to Professor Getachew A. Dagne for chairing the session.

I express my gratitude to the University of South Florida, especially the Mathematics and Statistics Department for all the help and resources made available to me.

Thanks to the Academic Success Center, its administrators Zoraya Betancourt, Dr. Ana Torres-Ayala, Monica Quintero, Dr. Kristen Gay, Katie Sawyer and my fellow Graduate Assistants, Matt Allman, Nicole Snyder, Johnny Johnson, Celina Major, Elizabeth Wood, Jaeden Ayala, Damir Khussainov, Aisha Schadrac for their contribution to my professional development. Thanks also to the tutors and ambassadors at The Center. I am indebted to the Academic Success Center and Majmaah University for financial support.

I am also indebted to all my former and current graduate student friends in the special topics course for their constructive suggestions and critiques, particularly Dr. Nawa Raj Pokhrel, Dr. Mahdi

Goudarzi, and Dr. A. K. M. Bashar. Finally, I would like to express my thanks to my mother, brothers, and sisters for their many manifold blessings to me, and my friends Maria Varela Suarez and Jim Alcorn for support and encouragement. I would not have been able to complete the pursuit of my doctoral studies without them.

Table of Contents

List of Tables

List of Figures

Abstract

As most of mankind now lives in an era of high dependence on multiple technologies and complex systems to store and manage sensitive information, researchers are constantly urged to obtain and improve measurements and methodologies that have the ability to evaluate systems reliability and security. The objectives of the present dissertation are to improve the Bayesian reliability estimation of a software package where the Power Law Process, also known as Non-Homogeneous Poisson Process, is the underlying failure model and to develop a set of statistical models evaluating computer operating systems vulnerabilities. Furthermore, we develop a reliability function of a computer network system using the Common Vulnerability Scoring System framework.

In the context of software reliability, we propose a Bayesian Reliability analysis approach of the Power Law Process under the Higgins-Tsokos loss function for modeling software failure times. We demonstrate, using real data, that the shape parameter of the Power Law Process behaves as a random variable. Based on Monte Carlo simulations and using real data, we show that the Bayesian estimate of the shape parameter and the proposed estimate of the scale parameter perform better compared to approximate maximum likelihood estimates, while they are sensitive to a prior selection. Using this result, we obtained a Bayesian reliability estimate of the Power Law Process. The results of this study have the potential to contribute not only to the reliability analysis field but also to other fields that employ the Power Law Process.

We further illustrate the robustness of the Higgins-Tsokos loss function verses the commonly used squared-error loss function in Bayesian Reliability analysis of the Power Law Process. Based on extensive Monte Carlo simulations and using real data, the Bayesian estimate of the shape parameter and the proposed estimate of the scale parameter were not only as robust as the Bayesian estimates under the squared-error loss function, but also performed better. The reliability function of the Power Law Process is a function of the intensity function; therefore the relative efficiency is used to compare the intensity function estimates. The intensity function using the Bayesian

estimate of the shape parameter under the Higgins-Tsokos loss function and its influence on the scale parameter estimate is more efficient than using the Bayesian estimate under the squared-error loss function. Moreover, using Monte Carlo simulations for different sample sizes, we show the efficiency and best performance of Bayesian reliability analysis under the Higgins-Tsokos loss function, recognizing that it is sensitive to selections of its parameters values and the shape parameter's prior density function. An interactive user interface application was developed to simply, without any prior coding knowledge required of the user, compute and visualize the Bayesian and maximum likelihood estimates of the intensity and reliability functions of the Power Law Process for a given dataset.

In addition, we propose a new approach using Copula theory to obtain Bayesian estimates of the Power Law Process intensity function parameters. We first demonstrate, using real data, the random behaviors of the shape and scale parameters of the Power Law Process. We then show the applicability of Copula theory in capturing the dependency structure of the subject parameters and develop a bivariate probability distribution that best characterizes their bivariate probabilistic behaviors. Copula-based Bayesian analysis, under the squared-error loss function and the developed bivariate probability distribution, was studied, where Copula-based Bayesian estimates of the shape and scale parameters of the Power Law Process are obtained simultaneously, considering both parameters unknown and random quantities. Monte Carlo simulations and using real data found superiority of the simultaneous Copula-based Bayesian estimates of the shape and scale parameters of the Power Law Process compared to their corresponding approximate maximum likelihood and Jeffreys Bayesian estimates, under the Higgins-Tsokos loss function where only the shape parameter is considered an unknown and random quantity. A Copula-based Bayesian reliability estimate of the Power Law Process is then obtained using the obtained Copula-based Bayesian estimates of the subject parameters. This result is expected to be widely used in different fields that employ the Power Law Process.

In the context of cybersecurity, we demonstrate the vulnerability scores of the commonly used computer operating systems from 1999 to 2019. We then show the difficulty of performing parametric analysis and proceed to develop non-parametric analytical probability and cumulative density functions estimates of the vulnerability scores of Microsoft, Apple, and Linux computer operating

systems, combined and individually. We also obtain and compare probabilities and expected values of their low, medium, and high vulnerability scores. The results showed that the vulnerability scores of Microsoft computer operating systems have higher expected and median values of vulnerabilities compared to Linux and Apple computer operating systems. The developed estimates of the probability and cumulative density functions of computer operating systems vulnerability scores, along with their graphical figures, should help IT mangers better understand their behavior probabilistically and serve as an important marketing tool.

Moreover, we propose analytic classification and prediction models to classify and predict the vulnerability severity level (low (0-3.9), medium (4-6.9), and high (7-10)) and score of a given computer operating system vulnerability, respectively, based on the Random Forest method subject to 13 risk factors. Using the National Vulnerability Database, we evaluated the most commonly used classification methods to develop analytical models to classify computer operating systems vulnerabilities. Evaluation of the Random Forest method showed it to have the best performance, on the subject vulnerabilities, compared to the other methods based on multiple evaluation metrics. We also propose a Random Forest classification method to develop an analytical model subject to 13 risk factors to classify whether a given computer operating system vulnerability will allow attackers to cause Denial of Service to the subject system. Furthermore, we develop a ranking process of computer operating systems vulnerabilities risk factors using the Random Forest regression method which should help prioritize the remediation process. The developed analytical models will assist not only vendors and information technology specialists, but also end-users, in managing and understanding the impact of the unfixed and newly discovered vulnerabilities of their computer operating systems.

Finally, we expand upon previous efforts and proceed to utilize two structured statistical models to simulate expected path length and the minimum number of steps data necessary to hack a computer network system with a very high probability. We illustrate a process of identifying probability distribution functions, parametrically and non-parametrically, that characterize the probabilistic behaviors of the expected path length and the minimum number of steps data. A mixture of Gamma and LogNormal probability distributions was found to be a good fit to the data. Also, we develop non-parametric analytical density estimates of the expected path length and the minimum

number of steps to hack the computer network system with a very high probability. As well, we perform a parametric reliability analysis, and additionally a non-parametric reliability analysis of the computer network system for maintenance purposes and other administrative management. The analytical procedure and methodology presented in this study can be applied to a larger computer network system.

Chapter 1
Introduction

The present study is a series of research problems in the fields of reliability estimation and cybersecurity. It consists of eight chapters where the first chapter is an overview of software reliability and vulnerability of computer operating systems. The second, third, and fourth chapters are concerned with improvements in Bayesian reliability estimation methodologies under the Power Law Process with application to software reliability. The methods we developed are not only applicable to software reliability domain but can be used in survival analysis, finance, and transportation, among other domains. The fifth, sixth, and seventh chapters are dedicated to present a set of statistical models to be applied in the field of cybersecurity, with application to software vulnerability. The last chapter presents the future research.

This chapter presents a review of software reliability and vulnerability of computer operating systems that are related to the present studies. Firstly, an overview of software reliability theory is presented, along with ordinary Bayesian methods, Bayesian point estimation, and the power law process. Secondly, an overview of quantitative software vulnerability is reviewed, along with the vulnerability database. Finally, we summarize the structure of the problems that we investigate in each study.

1.1 Software Reliability

1.1.1 Overview

In the early 1950s, the increase in system complexity has led scientists to the exploration of parametric life testing field. A series of research papers enriched the theory and practice of reliability estimations, where exponential time-to-failure distribution was the assumed failure distribution. Shortly after that, other probability distributions were considered as failure distributions such as gamma, Weibull, and Poisson probability distributions, among others. Now, we have several classical probability distributions where most of them considered as failure distributions [94, 95].

Reliability analysis of a software under development is a key to assess whether a desired level of a quality product is achieved. Specially, when a software package is considered, and is tested after each failure detection, and then corrected until a new failure is observed.

There are some commonly used statistical concepts in reliability analysis such as reliability function, hazard rate, and time to failure. The data collected while software reliability testing are considered realizations of a random variables. Moreover, time to failure can be defined as a random variable X with a probability density function $f(x)$. The probability that the software will fail by the time x is calculated using the cumulative probability density function, which is given by:

$$F(x) = P(X \leq x) = \int_0^x f(t)dt, \quad t \geq 0\,.$$

Software reliability is the probability that it will perform (i.e. will not have any failure) according to predetermined standards over a period of time [14, 22]. Based on this definition, the software reliability can be written mathematically as:

$$R(x) = P(X > x) = 1 - F(x) = \int_x^\infty f(t)dt, \quad t \geq 0,$$

where $R(x)$ is the reliability function. It is also called survival function in medical applications and denoted by $S(x)$.

Software developers might be also interested in the instantaneous failure rate at which the software has a failure in certain time, given no previous failures are detected. This is known as the hazard function and defined as follows:

$$h(x) = \lim_{\Delta x \to 0} \frac{P(x < X \leq x + \Delta x | X > x)}{\Delta x} = \frac{f(x)}{R(x)}\,.$$

The hazard function $h(x)$ represents the probability that the software will have a failure in a small period of time; from x to $x + \Delta x$, given that it has survived up to time x.

1.1.2 Parametric Analysis

Parametric analysis in a branch of statistics where the underlying probability distribution of a given data (i.e. time to failure of a software package) is well-defined. Then, inferential statistical analysis is used to study the properties of the underlying probability distribution.

One aspect of such analysis is parameters estimation of the well-defined probability distribution.

The parameters are assumed unknown and constant quantities. The method of maximum likelihood estimation is commonly used due to its robustness compared to other methods that are based on moments and quantiles. The maximum likelihood estimation is based on maximizing likelihood function so that the given data are more probable under the assumed well-defined distribution.

1.1.3 Ordinary Bayesian Method

Estimating software reliability from a probabilistic or statistical viewpoint would require an engineer to specify or identify a probability distribution of the software failures. In classical statistical analysis, the parameters of the failure model are considered unknown and constant quantities. However, Bayesian analysis rests on the assumption that at least one of the parameters within the failure model is unknown and random, and thus, an engineer can incorporate their prior knowledge. Bayesian analysis relies on utilizing prior knowledge, along with Bayes theorem.

The ordinary Bayesian method in reliability analysis is conducting a classical statistical analysis that involves parameters estimations. One of the parameters (i.e. β) is considered a random variable in such statistical analysis. With a proper justification, probability density function of the parameter of interest must be well-defined.

The underlying failure model of a given data could have two parameters that are considered unknown and random. If the behaviors of the two parameters are assumed to be independent, a probability density function, as a prior knowledge, is required to be well-defined for each parameter. In some cases, the the behaviors of the two parameters are observed to be dependent, which requires an engineer to identify or structure a bivariate probability density function that best characterizes their probabilistic behaviors.

In addition, a loss function with respect to the posterior probability distribution of β after collecting the study's data ($\pi(\beta|\mathbf{x})$) must be considered. The posterior distribution is the base of all inference whereas in classical statistical approach additional techniques are needed such as confidence intervals [11].

1.1.4 Bayesian Point Estimation

Consider the underlying probability density function of a given data, $f(\mathbf{x};\beta)$, has one parameter β. The goal of Bayesian point estimation is to estimate the unknown parameter of interest β based on observing realizations $\mathbf{x} = (x_1, \ldots, x_n)$ of independent random variables $X_1, X_2,..,X_n$ with a common PDF conditional on β, $f(x_1, \ldots, x_n\ |\beta)$. The likelihood function of the observed data ($\mathbf{x}$)

conditional on β is defined as follows:

$$L(\mathbf{x}|\beta) = P(\mathbf{x}|\beta) = \prod_{n=1}^{n} P(\mathbf{x}_i)\,.$$

In contrast to classical statistical approach, β is considered a random variable characterized by a probability distribution $\pi(\beta)$ (prior information of β) in the ordinary Bayesian approach. In association with Bayes' theorem, this prior distribution can be updated to the posterior distribution $\pi(\beta|\mathbf{x})$ of β conditional on the observed data ($\mathbf{x}$). The posterior distribution of β can be defined as follows:

$$\pi(\beta|\mathbf{x}) = P(\beta|\mathbf{x}) = \frac{P(\beta \cap \mathbf{x})}{P(\mathbf{x})} = \frac{P(\mathbf{x}|\beta) \cdot \pi(\beta)}{P(\mathbf{x})},$$

where:

- $P(\mathbf{x}|\beta)$ is the likelihood function of the data given the parameter β.
- $P(\mathbf{x})$ is the evidence (sometimes called marginal likelihood)

By specifying a loss function, we can find the point in the posterior distribution ($\pi(\beta|\mathbf{x})$) that we shall use as our estimate. The squared-error loss function is commonly used as a loss function in Bayesian analysis because it offers analytical tractability. It is given by:

$$L(\hat{\beta}, \beta) = (\hat{\beta} - \beta)^2\,.$$

The Bayes estimator of the realization β is the value of $\hat{\beta}$ that minimizes the expected loss (also called the risk). The expected loss is given by:

$$E[L(\hat{\beta}, \beta)] = \int_{-\infty}^{\infty} [(\hat{\beta} - \beta)^2] \pi(\beta|\mathbf{x}) d\beta\,.$$

By differentiating $E[L(\hat{\beta}, \beta)]$ with respect to β and setting it equal to zero we solve for $\hat{\beta}$, therefore the Bayesian estimate of β with respect to the squared-error loss function and prior probability distribution $\pi(\beta)$ is given by:

$$\hat{\beta}_{B.SE} = \int_{-\infty}^{\infty} \beta \cdot \pi(\beta|\mathbf{x}) d\beta,$$

where the posterior PDF of β given data ($\mathbf{x}$), $\pi(\beta|\mathbf{x})$, using the Bayes' theorem, is given by:

$$\pi(\beta|\mathbf{x}) = \frac{L(\mathbf{x}|\beta)\pi(\beta)}{\int_{-\infty}^{\infty} L(\mathbf{x}|\beta)\pi(\beta)d\beta}.$$

Note that in this case, the mean of the posterior density is the estimate that minimizes the expected loss.

1.1.5 Power Law Process

Repairable software is tested until a failure is detected, then fixed, and tested again until a new failure is detected. When number of failures are collected, let $0 < T_1 < T_2 < \cdots < T_n$ denotes the time to failure of the software under development where times are measures in global time [44, 82, 65].

Over the past few decades, the reliability analysis of a software package has been studied, where graphical and numerical metrics have been introduced. One of the earliest, Duane (1964) [29], who introduced a graph to assess the reliability of a software package over time using its failure times. The graph has the cumulative failure rate a software package and the time on the y-axis and x-axis, respectively. In this graph, one can conclude a software reliability improvement if a negative curve is observed whereas a positive curve means the software reliability is deteriorating. On other hand, a horizontal line indicates that the software reliability is stable. The failure numbers $N(t)$ in time interval $(0, t]$ is considered a Poisson counting process after satisfying the following conditions:

1. $N(t = 0) = 0$.

2. Independent increment (counts of disjoint time intervals are independent).

3. It has an intensity function:

$$V(t) = \lim_{\Delta t \to 0} \frac{P(N(t, t + \Delta t) = 1)}{\Delta t}.$$

4. Simultaneous failures do not exist:

$$\lim_{\Delta t \to 0} \frac{P(N(t, t + \Delta t) = 2)}{\Delta t} = 0.$$

The probability of random value $N(t) = n$ is given by:

$$P(N(t) = n) = \frac{\exp\left\{-\int_0^t V(t)dt\right\}\left\{\int_0^t V(t)dt\right\}^n}{n!},\ t > 0, \tag{1.1.5.1}$$

which is the definition of a Non-Homogeneous Poisson process for a nonconstant intensity function $V(t)$. It implies that for a $N(t)$ in the time interval $(a, b]$, with any $a < b$, follows a Poisson probability distribution with parameter $\int_a^b V(t)dt$, [26, 65, 4].

Crow (1974) [25] proposed a Non-Homogeneous Poisson Process (NHPP) , which is a Poisson Process with a time varying intensity function, given by:

$$V(t) = V(t; \beta, \theta) = \frac{\beta}{\theta} \left(\frac{t}{\theta}\right)^{\beta-1}, \ t > 0, \ \beta > 0, \ \theta > 0, \tag{1.1.5.2}$$

with β and θ are the shape and scale parameters, respectively. This Non-Homogeneous Poisson Process is also known as the Power Law Process (PLP).

The joint probability density function (PDF) of the ordered failure times T_1, T_2, ..., T_n from a NHPP with intensity function $V(t; \beta, \theta)$ is given by:

$$f\left(t_1, .., t_n\right) = \prod_{i=1}^{n} V\left(t_i; \beta, \theta\right) exp\left\{- \int_0^w V\left(t; \beta, \theta\right) dt\right\}, \tag{1.1.5.3}$$

where w is the so-called stopping time; w = t_n for the failure truncated case. Considering the failure truncation case, the conditional reliability function of the failure time T_n given $T_1 = t_1$, $T_2 = t_2$, $T_3 = t_3$,..., $T_{n-2} = t_{n-2}$, $T_{n-1} = t_{n-1}$ is a function of $V(t; \beta, \theta)$.

As a numerical assessment, the estimate of the key parameter β in the $V(t; \beta, \theta)$ has an important role in evaluating the reliability of a software package. When the estimates of β are less and larger than 1, they indicate that the software reliability is improving and decreasing, respectively. The PLP is reduced to a homogeneous Poisson process when the estimate of β equals to 1.

The subject model has been shown to be effective and useful not only in software reliability assessment [26, 100, 39, 21, 9, 40, 97, 82, 83, 84, 5], but also in cybersecurity; the attack detection in cloud systems [61, 67], breast and skin cancer treatments' effectiveness, [98, 93, 45], respectively, finance; modeling of financial markets at the ultra-high frequency level [60], trnasportation; modeling passengers' arrivals [56, 86, 77, 63, 102], and in the formulation of a software cost model [58].

1.1.6 Analytical Power Law Process

The probability of achieving n failures of a given system in the time interval $(0, t]$ can be written as [5]:

$$P(x=n;t)=\frac{\exp\left\{-\int_0^t V(x)dx\right\}\left\{\int_0^t V(x)dx\right\}^n}{n!},\ t>0,$$

where $V(t)$ is the intensity function given in (1.1.5.2). The reduced expression is given by:

$$P(x=n;t)=\frac{1}{n!}\exp\left\{-\frac{t}{\theta}^{\beta}\right\}\frac{t}{\theta}^{n\beta}.$$

The PLP that is commonly known as Weibull or Non- Homogeneous Poisson Process.

When the PLP is the underlying failure model of the ordered failure times t_1, t_2, t_3,... , t_{n-1}, and t_n, the conditional reliability function of t_n given t_1, t_2, t_3,... , t_{n-1} can be written mathematically as a function of the intensity function, given by:

$$R(t_n|t_1,t_2,...,t_{n-1})=\exp\left\{\int_{t_{n-1}}^{t_n}-V(t;\beta,\theta)dt\right\},\ tn>t_{n-1}>0, \tag{1.1.6.1}$$

since it is independent of t_1, t_2, t_3, ... , t_{n-2}.

Note that the improvement in estimating the key parameter β in the $R(t_n|t_1,t_2,...,t_{n-1})$ of the PLP, equation (1.1.6.1), will improve the reliability estimation.

The maximum likelihood estimation (MLE) of β is a function of the largest failure time and the MLE of θ is also a function of the MLE of β. Let T_1, T_2, ..., T_n denote the first n failure times of the PLP, where $T_l<T_2<...<T_n$ are measured in global time; that is, the times are recorded since the initial startup of the system. Thus, the truncated conditional probability distribution function, $f_i(t|t_1,...,t_{i-1})$, in the Weibull process is given by

$$f_i(t|t_1,...,t_{i-1})=\frac{\beta}{\theta}\left(\frac{t}{\theta}\right)^{\beta-1}\exp\left\{-\frac{t}{\theta}^{\beta}+\frac{t_{i-1}}{\theta}^{\beta}\right\},\ t_{i-1}<t.$$

With $t=(t_1,t_2,...,t_n)$, the Likelihood function for the first n failure times of the PLP $T_1=t_1,T_2=t_2,...,T_n=t_n$ can be written as:

$$L(t,\beta)=\exp\left(-\left(\frac{t_n}{\theta}\right)^{\beta}\right)\left(\frac{\beta}{\theta}\right)^n\prod_{i=1}^{n}\left(\frac{t_i}{\theta}\right)^{\beta-1}.$$

The MLE for the shape parameter is given by

$$\hat{\beta}_n=\frac{n}{\sum_{i=1}^{n}\log\left(\frac{t_n}{t_i}\right)},$$

and for the scale parameter is

$$\hat{\theta}_n = \frac{t_n}{n^{1/\hat{\beta}_n}}.$$

Note that the MLE of θ depends on the MLE of β.

1.2 Software Vulnerability

1.2.1 Overview

Information security is a major concern of not only software developers but also end-users. Software reliability and security are important components to be evaluated during the development process, and monitored as long as the software is publicly available to users. Our high dependency on software packages to running simple daily routines, such as storing our personal and financial information, put us at risk of trusting such software. Small and large businesses use software packages to operate their businesses and store their data as well.

Software security breaches mostly happen when hackers identify and exploit the given software vulnerabilities. The National Institute of Standards and Technology (NIST), under the U.S. Department of Commerce, defines system vulnerability as, "**Weakness in an information system, system security procedures, internal controls, or implementation that could be exploited or triggered by a threat source**". Clearly, when a software package has several vulnerabilities or one sever vulnerability, it makes it less reliable and vice versa.

There has been efforts to evaluate the vulnerabilities of software packages to help establishing and maintaining security policies. One of which is developing a scoring system to classify each software vulnerability. Despite scoring systems developed by software vendors, the Common Vulnerability Scoring System (CVSS) is the commonly used scoring system. CVSS is under the custodial care of the Forum of Incident Response and Security Teams (FIRST) [62]. CVSS provides software developers and end-users with a framework to communicate publicly published software vulnerabilities and assist in prioritizing their efforts related to the most significant impacts [87]. There are three versions of the CVSS, namely, versions 1.0, 2.0, and 3.0., referred as CVSS v1, v2, and v3.

1.2.2 Common Vulnerability Scoring System

The CVSS is a framework to evaluate software vulnerabilities was developed by CVSS Special Interest Group (CVSS-SIG) under the FIRST [33]. The CVSS has three versions [32]; CVSS v1,

v2, and v3, and they were published in 2004, 2007, and 2015, respectively. Technology community including organizations and individual specialists are using CVSS to assess software vulnerabilities and prioritize their remediation processes [78]. The vulnerability score (VS) based on CVSS is a number between 0 and 10, where the vulnerability's severity increases as the number approaches to 10. There are three metrics used to compute the VS; namely, base, temporal, and environmental metrics. The VS using the base metrics is the emphasis of this research. It has 6 metrics; congregated into exploitability metrics (Access Vector (AV), Access Complexity (AC), and Authentication (AU)) and impact metrics (Availability Impact (AI), Confidentiality Impact (CI), and Integrity Impact (II)). The schematic diagram given by Figure 1.1 shows the computation of the vulnerability base score based on CVSS v2.

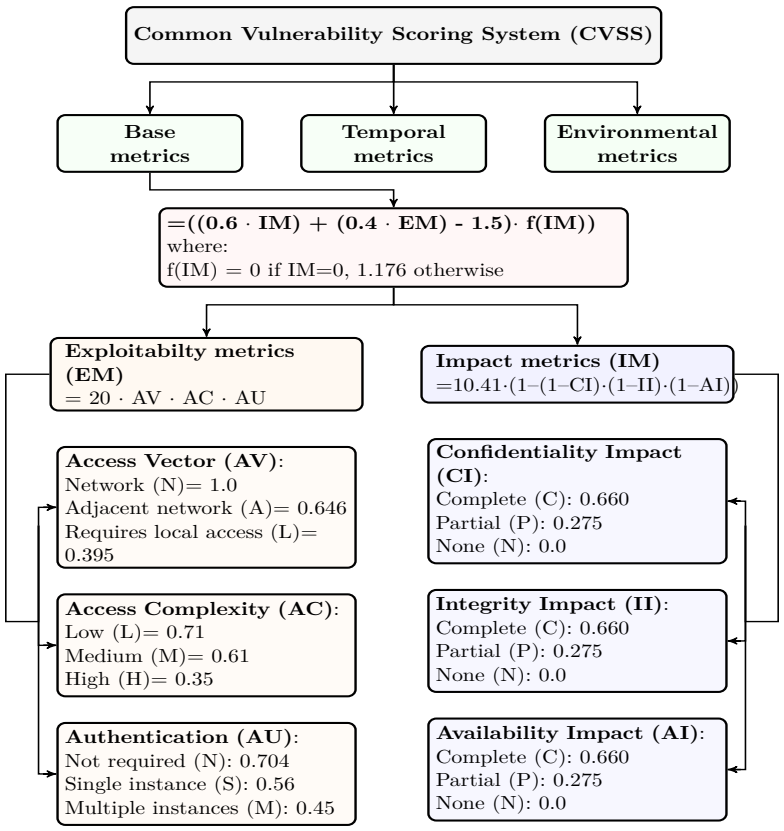


Figure 1.1.: The calculation of the vulnerability base score under the CVSS v2 framework where it is eventually rounded to one decimal place

According to FIRST, the CVSS-SIG and others have identified major issues with the CVSS v1. One of main issue is that it did not go through mass peer review across multiple organizations and industries, which led the team to develop and publish the CVSS v2 [99].

1.2.3 Vulnerability Database

The National Institute of Standards and Technology (NIST), under the U.S. Department of Commerce, has several initiatives such as Common Vulnerabilities and Exposures (CVE), National Vulnerability Database (NVD), and Common Vulnerability Scoring System (CVSS).

The CVE was established in 1999, and can be seen as a vulnerabilities' dictionary, where each vulnerability is assigned a unique identification (ID) by the CVE Numbering Authorities (CNAs). CNAs are international organizations that are authorized to communicate and assign new vulnerabilities that are related to their products with a unique ID. The vulnerabilities' IDs are therefore made publicly available. The standards followed for assigning vulnerabilities' IDs and the process of participating as a member of the CNAs can be found on the CVE web-page [24]. CVE is currently maintained and managed by MITRE [23]. The vulnerability ID consists of three parts separated by dashes, namely, the abbreviation CVE, year, and four digis serial number. For example, CVE-2019-0772 is assigned to one of the Windows 10 vulnerabilities that was published on April 8, 2019.

The NVD [72] is a publicly and freely available comprehensive database of products' vulnerabilities. It is a product of the Computer Security Division, NIST, and sponsored by the National Cyber Security Division, Department of Homeland Security. NVD was established in 2000 under the name Internet –Categorization of Attacks Toolkit and abbreviation (ICAT). The NVD staff analyze the published CVEs, and collect the associated description, references, and any other related public data.

1.3 Summary of the Present Studies

1.3.1 Bayesian Reliability Approach to the Power Law Process Under Higgins-Tsokos Loss Function

The primary purpose of this study is to develop the analytical Bayesian form of the key parameter β, under the Higgins-Tsokos loss function, in the intensity function. The underlying failure distribution is the Power Law Process. The Power Law Process, also known as Non-Homogeneous Poisson Process, has been used in various aspects, one of which is the software reliability assessment. Specifically, by using its intensity function to compute the rate of change of a software reliability as time-varying function. Justification of Bayesian analysis applicability to the Power Law Process was shown using real data. The probability distribution that best characterizes the behavior of

the key parameter of the intensity function was first identified, then the likelihood-based Bayesian reliability estimate of the Power Law Process under the Higgins-Tsokos loss function was obtained.

As a result of a simulation study and using real data, the Bayesian estimate shows an outstanding performance compared to the approximate maximum likelihood estimate using different sample sizes. In addition, a sensitivity analysis was performed, resulting in the Bayesian estimate being sensitive to the prior selection; whether parametric or non-parametric.

1.3.2 The Effectiveness of The squared-error And Higgins-Tsokos Loss Functions On The Bayesian Reliability Analysis Of Software Failure Times Under The Power Law Process

The primary purpose of this study is to investigate the effectiveness of the squared-error and Higgins-Tsokos loss functions on the Bayesian reliability analysis of software failure times under the Power Law Process. The choice of a loss function is an important entity of the Bayesian settings. The analytical estimate of likelihood-based Bayesian reliability estimates of the Power Law Process under the squared-error and Higgins-Tsokos loss functions were obtained for different prior knowledge of its key parameter.

As a result of a simulation analysis and using real data, the Bayesian reliability estimate under the Higgins-Tsokos loss function not only is robust as the Bayesian reliability estimate under the squared-error loss function but also performed better, where both are superior to the maximum likelihood reliability estimate. A sensitivity analysis resulted in the Bayesian estimate of the reliability function being sensitive to the prior, whether parametric or non-parametric, and to the loss function.

An interactive user interface application was additionally developed using Wolfram language to compute and visualize the Bayesian and maximum likelihood estimates of the intensity and reliability functions of the Power Law Process for a given data.

1.3.3 Copula-Based Bayesian Reliability Analysis of The Power Law Process

The primary purpose of this study is to develop a bivariate probability distribution function of the Power Law Process parameters β and θ. In addition, we develop analytical Bayesian forms, under the squared-error loss function, of the subject parameters simultaneously, and use them to obtain a Bayesian Reliability function of the Power Law Process. We demonstrate Copula theory applicability using real data, and show that the parameters behave as dependent random

variables. With a sequence of estimates of the parameters we proceed to develop a bivariate probability distribution that characterizes their joint behaviors. We then obtain a Bayesian estimate of the reliability function using squared-error loss function and the developed bivariate probability distribution as a bivariate prior.

Based on extensive simulations, the simultaneous Bayesian estimates of parameters β and θ are superior to the maximum likelihood estimates, across different sample sizes.

1.3.4 Probabilistic Comparisons of Commonly Used Computer Operating Systems' Vulnerabilities

The primary purpose of this study is to demonstrate the vulnerability scores of the commonly used operating systems and identify a probability distribution that best characterises their probabilistic behavior. In this study, We collected the vulnerability data from Common Vulnerabilities and Exposures and National Vulnerability Database databases. We then employ the kernel density approach to characterize the probabilistic behavior of each of the Microsoft, Linux, and Apple operating systems vulnerability scores. In addition, we obtain a probabilistic comparisons of the vulnerability scores of the subject operating systems.

The results showed that there is not a well-defined probability distribution that fits the vulnerability scores of Microsoft, Linux, or Apple operating systems. In the kernel density approach, the optimal bandwidth (h^*) and kernel function were chosen so as to minimize the asymptotic mean integrated squared error (AMISE). Based on Epanechnikov kernel function, we obtained kernel density estimates of the probability and cumulative density functions of Microsoft, Linux, and Apple computer operating systems vulnerability scores combined and individually. Also, we found that the Microsoft computer operating systems have higher expected and median values of vulnerability scores comparing to other operating systems.

1.3.5 Effective Classification and Prediction Methods of Computer Operating Systems Vulnerabilities Subject to Risk Factors

The primary purpose of this study is to develop statistical models that contribute to the field of cybersecurity. We also show the effectiveness of common machine learning algorithms in classifying and predicting the operating systems vulnerability levels (low (0-3.9), medium (4-6.9), and high (7-10)) and scores, respectively, subject to 13 risk factors. We use published vulnerability data in National Vulnerability Database. The Random Forest algorithm has the best performance,

compared to other algorithms, in predicting the operating system vulnerability level based on precision, recall, and F-measure evaluation metrics. Considering the regression version of the Random Forest algorithm, the explained variance, R squared, adjusted R squared, and root mean squared-error confirmed the excellent performance of the model in predicting the vulnerability score of a given vulnerability description for an operating system. In addition, a classification model, subject to the 13 risk factors, was developed to classify whether a newly discovered OS vulnerability would allow attackers to cause denial of service to the system.

1.3.6 Parametric and Non-parametric Modeling of a Computer Network Reliability

The primary purpose of this study is to develop a set of statistical methodologies that contribute to the field of cybersecurity; computer network security. In this study, we extend a previous study by Kaluarachchi and Tsokos [54]. The authors utilized the National Vulnerability Database, using a model computer network example with a host centric attack graph. They developed a statistical models to estimate the expected path length of hacking the computer network as a function of systems vulnerability scores and the minimum number of steps to compromise the network example with a probability of one using the Markovian iteration process.

We model, parametrically and non-parametrically, the expected path lengths for a hacker to compromise the computer network and the minimum number of steps that an attacker needs to hack the network with a probability of one with different vulnerability information. Moreover, We use the statistical models provided by the Kaluarachchi and Tsokos [54], and utilize the National Vulnerability Database (NVD) to calculate the expected path length and minimum number of steps that an attacker will need to hack the network with a probability of one. Then proceed to identify the probability distribution functions that characterize the probabilistic behaviors of those estimates, respectively. Furthermore, we develop parametric and non-parametric reliability functions of the network example.

When a company uses its own computer network, this parametric and non-parametric modeling will assist their cybersecurity specialists not only in monitoring the security status but also in the reliability status of the network, especially in estimating the probability that a randomly selected set of computers in the network example will have an expected path length that falls between two desired time units. Similar usefulness applies to the minimum number of steps that an attacker needs to hack the network example with a probability of one. More importantly, estimating the reliability and hacking rate of the computer network.

Chapter 2

Bayesian Reliability Approach to the Power Law Process Under Higgins-Tsokos Loss Function

In this chapter, we investigate the effectiveness of Bayesian analysis using the Higgins-Tsokos (H-T) loss function for modeling software failure times. To accomplish this, we use the Power Law Process as the underlying failure distribution, we utilize the H-T loss function to perform sensitive analysis of prior selections employing parametric and non-parametric priors. One article is published [2] based on findings of this Chapter.

The chapter is organized as follows: Section 2 describes the theory and development of the Bayesian reliability model; Section 3 presents the results and discussion; Section 4 details the contributions.

2.1 Introduction

Software reliability growth is often tested during the software development process to insure a good quality product. Repairable software is tested until a failure is detected, then fixed, and tested again until a new failure is detected. This reliability improvement of software has been studied for decades. Duane (1964) [29] introduced the "learning curve approach", which is a plot of the failure rate (or the intensity function) of a system as a function of time. It is used to assess software reliability improvement over time. For example, software reliability has improved when we observe a negative curve, whereas a positive curve means that reliability is deteriorating. Stability in software reliability is achieved when the graph is a horizontal line. The number of failures in the interval $(0, t]$, $N(t)$, is considered a Poisson counting process after satisfying the following conditions, [27]:

1. $N(t = 0)$=0.

2. Independent increment (counts of disjoint time intervals are independent).

3. It has an intensity function $V(t) = \lim_{\Delta t \to 0} \frac{P(N(t,t+\Delta t)=1)}{\Delta t}$.

4. Simultaneous failures do not exist ($\lim_{\Delta t \to 0} . \frac{P(N(t,t+\Delta t)=2)}{\Delta t}=0$).

The probability of a random value $N(t)$=n is given by:

$$P(N(t) = n) = \frac{\exp\left\{-\int_0^t V(t)dt\right\}\left\{\int_0^t V(t)dt\right\}^n}{n!},\ t > 0. \tag{2.1.1}$$

Crow (1974) proposed a non-homogeneous Poisson process (NHPP) [25], which is a Poisson process with a time-varying intensity function, given by:

$$V(t) = V(t;\beta,\theta) = \frac{\beta}{\theta}\left(\frac{t}{\theta}\right)^{\beta-1},\ t > 0,\ \beta > 0,\ \theta > 0, \tag{2.1.2}$$

with β and θ as the shape and scale parameters, respectively. This NHPP is also known as the power law process (PLP).

The joint probability density function (PDF) of the ordered failure times T_1, T_2, ..., T_n from an NHPP with intensity function $V(t;\beta,\theta)$ is given by:

$$f\left(t_1,..,t_n\right) = \prod_{i=1}^{n} V\left(t_i;\beta,\theta\right) exp\left\{-\int_0^w V\left(t;\beta,\theta\right)dt\right\}, \tag{2.1.3}$$

where w is the so-called stopping time. Considering the failure truncation case (w = t_n), the conditional reliability function of the failure time T_n given $T_1 = t_1$, $T_2 = t_2$, $T_3 = t_3$,..., $T_{n-2} = t_{n-2}$, $T_{n-1} = t_{n-1}$ is a function of $V(t;\beta,\theta)$.

To monitor software reliability growth over time, an engineer can use the estimate of the β value, the key parameter in the intensity function, since it plays a significant role during the testing process. For $\beta > 1$, the number of failures would increase because the intensity function is increasing. On the other hand, if the intensity function is decreasing, $\beta < 1$ means that the number of failures would decrease, indicating improved software reliability. Note that in the case of a homogeneous Poisson process pertains when $\beta = 1$, in which case the intensity function will be $\frac{1}{\theta}$ and whatever changes have been made have had no effect on the outcome.

The NHPP has been used for analyzing software failure times, and for predicting the next failure event. Several publications show the effectiveness and usefulness of this model in assessing reliability growth [26, 100, 40]. In addition, NHPP has been used to study drug effectiveness in breast cancer treatment [98] and in the formulation of a software cost model [58].

Since the intensity function is driving the NHPP, improving the existing methods to estimate

the key parameter β will certainly improve the accuracy of reliability growth assessment and help the structuring of maintenance strategies. Molinares and Tsokos [65], obtained a Bayesian estimate of the parameter β and compared it with its approximate maximum likelihood estimate (MLE). The authors derived the Bayesian estimates with respect to squared-error loss function, using Burr, Jeffreys, and inverted gamma probability distributions as the prior PDFs for β.

In performing Bayesian analysis on a real world problem, we need some sort of justification for pursuing this particular type of analysis. Once we have identified the probability distribution that characterized the probabilistic behavior of the failure time, we need to identify the prior PDF of β and a loss function. The squared-error loss function is the most popular loss function used in Bayesian analysis because of its analytical tractability. It places a small weight on the estimates around the true value, but proportionally more weight on estimates far from the true value. Higgins and Tsokos [47] proposed a new loss function that places exponential weight on extreme deviations from the true value, while remaining mathematically tractable.

In the present study, we investigate the effectiveness of Bayesian analysis in using the Higgins-Tsokos (H-T) loss function (that puts the loss at the end of the process) for modeling software failure times. To accomplish this, we use the NHPP as the underlying failure distribution subject to using the Burr PDF as a prior of β. In addition, we utilize the H-T loss function to perform sensitive analysis of prior selections. We employ parametric and non-parametric priors, namely Burr, inverted gamma, Jeffery, and two kernel PDFs. Therefore, the primary objective of the study is to answer the following questions within Bayesian framework:

1. What is the performance of the Bayesian estimate of β under the H-T loss function compared to its MLE when modeling software failure times using PLP?

2. Is the Bayesian estimate of β, using the H-T loss function in the PLP, sensitive to the selection of the prior PDF, both parametric and non-parametric?

2.2 Theory and Bayesian Estimates

2.2.1 Review of the Analytical Power Law Process

The probability of achieving n failures of a given system in the time interval $(0, t]$ can be written as

$$P(x=n;t)=\frac{\exp\left\{-\int_0^t V(t;\beta,\theta)dt\right\}\left\{\int_0^t V(t;\beta,\theta)dt\right\}^n}{n!},\ t>0, \tag{2.2.1.1}$$

where $V(t;\beta,\theta)$ is the intensity function given by (3.1.0.2). The reduced expression

$$P(x=n;t)=\frac{1}{n!}\exp\left\{-\frac{t}{\theta}^{\beta}\right\}\frac{t}{\theta}^{n\beta}, \tag{2.2.1.2}$$

is the PLP that is commonly known as the Weibull or NHPP.

If the PLP is the underlying failure model of the failure times t_1, t_2, t_3,... , t_{n-1}, and t_n, the conditional reliability function of t_n given t_1, t_2, t_3,... , t_{n-1} can be written as:

$$R(t_n|t_1,t_2,...,t_{n-1})=\exp\left\{\int_{t_{n-1}}^{t_n}-V(t;\beta,\theta)dt\right\},\ tn>t_{n-1}>0, \tag{2.2.1.3}$$

since it is independent of t_1, t_2, t_3, ... , t_{n-2}.
Since the reliability function, equation (2.2.1.3), is written mathematically as a function of the intensity function, estimating the parameter β in the $V(t;\beta,\theta)$ leads to estimation of the reliability function.
The (MLE) of β is a function of the largest failure time and the MLE of θ is also a function of the MLE of β. Let T_1, T_2, ..., T_n denote the first n failure times of the PLP, where $T_l<T_2<...<T_n$ are total times since the initial startup of the system. Thus, the truncated conditional PDF, $f_i(t|t_1,...,t_{i-1})$, in the Weibull process and is given by

$$f_i(t|t_1,...,t_{i-1})=\frac{\beta}{\theta}\left(\frac{t}{\theta}\right)^{\beta-1}\exp\left\{-\frac{t}{\theta}^{\beta}+\frac{t_{i-1}}{\theta}^{\beta}\right\},\ t_{i-1}<t. \tag{2.2.1.4}$$

With $t=(t_1,t_2,...,t_n)$, the likelihood function for the first n failure times of the PLP $T_1=t_1,T_2=t_2,...,T_n=t_n$ can be written as:

$$L(t;\beta)=\exp\left(-\left(\frac{t_n}{\theta}\right)^{\beta}\right)\left(\frac{\beta}{\theta}\right)^n\prod_{i=1}^{n}\left(\frac{t_i}{\theta}\right)^{\beta-1}. \tag{2.2.1.5}$$

The MLE for the shape parameter is given by:

$$\hat{\beta}_n = \frac{n}{\sum_{i=1}^{n} \log\left(\frac{t_n}{t_i}\right)}, \tag{2.2.1.6}$$

and for the scale parameter is:

$$\hat{\theta}_n = \frac{t_n}{n^{1/\hat{\beta}_n}}. \tag{2.2.1.7}$$

Note that the MLE of θ depends on the MLE of β using the largest (last) observed failure time.

2.2.2 Development of the Bayesian Estimates

Crow [26, 25] failure data from a system undergoing developmental testing was used, by Molinares & Tsokos [65], to show how β varied depending on the last failure time (largest time), thus they proposed a Bayesian approach to the PLP. The authors also found that the MLE of β follows a four-parameter Burr probability distribution, $g(\beta; \alpha, \gamma, \delta, \kappa)$, known as the four-parameter Burr type XII probability distribution, with a PDF given by:

$$g_B(\beta) = g(\beta; \alpha, \gamma, \delta, \kappa) = \begin{cases} \frac{\alpha\kappa\left(\frac{\beta-\gamma}{\delta}\right)^{\alpha-1}}{\delta\left(1+\left(\frac{\beta-\gamma}{\delta}\right)^{\alpha}\right)^{\kappa+1}}, & \gamma \leq \beta < \infty \\ 0 & , otherwise \end{cases}, \tag{2.2.2.1}$$

where the hyperparameters α, γ, δ and κ are being estimated using MLE in the goodness of fit (GOF) test applied to the β estimates. The Crow successive failure data for his system is given in Table 2.1.

Table 2.1: Crow's failure times of a system under development.

Failure times						
0.7	3.7	13.2	17.6	54.5	99.2	112.2
120.9	151	163	174.5	191.6	282.8	355.2
486.3	490.5	513.3	558.4	678.1	688	785.9
887	1010.7	1029.1	1034.4	1136.1	1178.9	1259.7
1297.9	1419.7	1571.7	1629.8	1702.4	1928.9	2072.3
2525.2	2928.5	3016.4	3181	3256.3	–	–

According to the reliability growth failure data, the system failed for the first time at 0.7 units of time, $t_1 = 0.7$, and it failed the 40^{th} time at 3256.3 units of time, $t_{40} = 3256.3$. The MLE of the parameter β for $n = 40$ is

$$\hat{\beta}_{40} = \frac{40}{\sum_{i=1}^{40} \log\left(\frac{3256.3}{t_i}\right)} \simeq 0.49. \tag{2.2.2.2}$$

If β were treated in a non-Bayesian setting, its MLE would be given by equation (2.2.2.2).

In an experimental process, the largest time to failure could occur at any point in the series of failures for a given system. Therefore, consider the case where the largest failure is $t_{39} = 3181$. In such a case, the estimate of β_{39} is 0.48.

The largest failure time always affects the MLE of β. Thus, it is recommended that β not to be thought of as an unknown constant [65], but rather as an unknown random variable. This recommendation provides the opportunity to study Bayesian analysis in the PLP with respect to various selections of loss functions and priors. The Bayesian estimates of β will be derived using H-T loss functions.

2.2.2.1 Bayesian Estimates Using the Higgins-Tsokos Loss Function

The H-T loss function (1976) is given by

$$L(\hat{\xi}, \xi) = \frac{f_1 \exp\left\{f_2(\hat{\xi} - \xi)\right\} + f_2 \exp\left\{-f_1(\hat{\xi} - \xi)\right\}}{f_1 + f_2} - 1,\ f_1, f_2 > 0. \tag{2.2.2.1}$$

Higgins and Tsokos [47] showed that it places more weight on the extreme underestimation and overestimation of the true value when $f_1 > f_2$ and $f_1 < f_2$, respectively. The risk using the H-T loss function, where $\xi = \beta$ represents the estimate of $\hat{\xi} = \hat{\beta}$, is given by:

$$E[L(\hat{\beta}, \beta)] = \int_{-\infty}^{\infty} [\frac{f_1 \exp\left\{f_2(\hat{\beta} - \beta)\right\} + f_2 \exp\left\{-f_1(\hat{\beta} - \beta)\right\}}{f_1 + f_2} - 1] h(\beta|t) d\beta. \tag{2.2.2.2}$$

By differentiating $E[L(\hat{\beta}, \beta)]$ with respect to β and setting it equal to zero we solve for $\hat{\beta}$, the Bayesian estimate of β with respect to the H-T loss function, is given by:

$$\hat{\beta}_{B.TH} = \frac{1}{f_1 + f_2} \ln[\frac{\int_{-\infty}^{\infty} \exp\{f_1\beta\} h(\beta|t) d\beta}{\int_{-\infty}^{\infty} \exp\{-f_2\beta\} h(\beta|t) d\beta}]. \tag{2.2.2.3}$$

The Bayesian estimate of β with respect to the H-T loss function and Burr probability distribution, as the prior, has $h(\beta|t)$ given by

$$h(\beta|t) = \frac{(\frac{\beta}{\theta})^n \exp\left\{-(\frac{t_n}{\theta})^\beta\right\} \prod_{i=1}^n \left(\frac{t_i}{\theta}\right)^{\beta-1} \frac{(\frac{\beta-\gamma}{\delta})^{\alpha-1}}{(1+(\frac{\beta-\gamma}{\delta})^\alpha)^{\kappa+1}} d\beta}{\int_\gamma^\infty (\frac{\beta}{\theta})^n \exp\left\{-(\frac{t_n}{\theta})^\beta\right\} \prod_{i=1}^n \left(\frac{t_i}{\theta}\right)^{\beta-1} \frac{(\frac{\beta-\gamma}{\delta})^{\alpha-1}}{(1+(\frac{\beta-\gamma}{\delta})^\alpha)^{\kappa+1}} d\beta}. \tag{2.2.2.4}$$

With the use of equation (2.2.1.3), the conditional reliability of t_i, the analytical structure of the conditional Bayesian reliability estimate for the PLP that is subject to the above information, is given by:

$$\hat{R}_B(t_i|t_1, t_2, ..., t_{i-1}) = \exp\left\{-\int_{t_{i-1}}^{t_i} \hat{V}'_B(t; \beta, \theta)dt\right\},\ t_i > t_{i-1} > 0, \tag{2.2.2.5}$$

where

$$\hat{V}'_B(t; \beta_{B.TH}, \theta) = \frac{\hat{\beta}_{B.TH}}{\theta}\left(\frac{t}{\theta}\right)^{\hat{\beta}_{B.TH}-1},\ \theta > 0, t > 0, \tag{2.2.2.6}$$

where $\hat{\beta}_{B.TH}$ is the Bayesian estimate of β using the H-T loss function. We are also interested in comparing the Bayesian estimate, using the H-T loss function, with MLE of the subject parameter for different parametric and non-parametric priors, assuming β has a random behavior and θ is known; and also comparing equation (2.2.1.7) with an adjusted MLE considered as a function of β.

2.2.3 Sensitivity Analysis: Prior Selection

In this section, we seek the answer to the following question: Is the Bayesian estimate of β, using the H-T loss function in the PLP, sensitive to the selection of the prior, with parametric or non-parametric priors? Assuming β is a random variable, using simulated data, sensitivity analysis was done for the following parametric and non-parametric priors:

1. Jeffreys' prior [51]:

   Jeffreys' prior is proportional to the square root of the determinant of the Fisher information matrix ($I(\beta)$). It is a non-informative prior, where the Jeffreys' prior for the PLP, considering that β, $I(\beta)$ is scalar in this case, is given by:

$$g^J(\beta) \propto \sqrt{I(\beta)} = \sqrt{-E(\frac{\partial^2 LogL(t;\beta)}{\partial\beta^2})} \propto \frac{1}{\beta},\ \beta > 0. \tag{2.2.3.1}$$

2. The inverted gamma:

   The PLP and inverted gamma probability distributions belong to the exponential family of probability distributions, which makes the latter a logical choice for an informative parametric prior for β. The inverted gamma probability distribution is given by:

$$g^{IG}(\beta) \propto \left(\frac{\mu}{\beta}\right)^{v+1} \frac{1}{\mu\Gamma(v)} \exp\left\{\frac{-\mu}{\beta}\right\},\ \beta > 0, \mu > 0, v > 0, \tag{2.2.3.2}$$

   where v and μ are the shape and scale parameters.

3. Kernel prior:

   The kernel probability density estimation is a non-parametric method to approximately estimate the PDF of β using a finite data set. It is given by:

$$g^{K}(\beta) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{\beta - \beta_i}{h}\right), \tag{2.2.3.3}$$

where K is the kernel function and h is a positive number called the bandwidth.

2.2.3.1 The Jeffreys' Prior:

Assuming Jeffreys' PDF, equation (3.2.3.1), as the prior of β and using the likelihood function (2.2.1.5), the posterior density of β is given by:

$$h_J(\bar{t}|\beta) = \frac{\exp\left\{\left(\frac{t_n}{\theta}\right)^{\beta}\right\} \frac{\beta^{n-1}}{\theta^{n\beta}} \prod_{i=1}^{n} (t_i)^{\beta-1}}{\int_0^{\infty} \exp\left\{\left(\frac{t_n}{\theta}\right)^{\beta}\right\} \frac{\beta^{n-1}}{\theta^{n\beta}} \prod_{i=1}^{n} (t_i)^{\beta-1} d\beta}. \tag{2.2.3.1.1}$$

Thus, the Jeffreys' Bayesian estimate of β in $V(t;\beta,\theta)$ under the H-T loss function, using equation (2.2.2.3), is given by:

$$\hat{\beta}_{J.HT} = \frac{1}{f_1 + f_2} \ln\left[\frac{\int_{\gamma}^{\infty} \exp\{f_1\beta\}\, h_J(\bar{t}|\beta) d\beta}{\int_{\gamma}^{\infty} \exp\{-f_2\beta\}\, h_J(\bar{t}|\beta) d\beta}\right]. \tag{2.2.3.1.2}$$

We cannot obtain a closed analytical form of the Bayesian estimate, $\hat{\beta}_{J.HT}$, thus we must utilize numerical method to obtain the subject estimate. Also note that the estimate depends on knowing or being able to estimate the scale parameter θ.

2.2.3.2 The Inverted Gamma Prior:

We proceed with our study with the prior probability density of β given by the inverted gamma distribution equation (2.2.3.2). Using the likelihood equation (2.2.1.5), the posterior density of β is given by:

$$h_{IG}(t|\beta) = \frac{\frac{\beta^{n-v-1}}{\theta^{n\beta}} \exp\left\{-\left(\frac{t_n}{\theta}\right)^{\beta} - \frac{\mu}{\beta}\right\} \prod_{i=1}^{n}(t_i)^{\beta-1}}{\int_0^{\infty} \frac{\beta^{n-v-1}}{\theta^{n\beta}} \exp\left\{-\left(\frac{t_n}{\theta}\right)^{\beta} - \frac{\mu}{\beta}\right\} \prod_{i=1}^{n}(t_i)^{\beta-1} d\beta}. \tag{2.2.3.2.1}$$

Thus, the Bayesian estimate of β under the inverted gamma distribution with respect to the H-T loss function, using equation (2.2.2.3) and equation (2.2.3.2.1), is given by:

$$\hat{\beta}_{IG.HT} = \frac{1}{f_1 + f_2} \ln[\frac{\int_{\gamma}^{\infty} \exp\{f_1\beta\} h_{IG}(t|\beta)d\beta}{\int_{\gamma}^{\infty} \exp\{-f_2\beta\} h_{IG}(t|\beta)d\beta}]. \tag{2.2.3.2.2}$$

Here as well, we must rely on a numerical estimation of $\hat{\beta}_{IG.HT}$ because we cannot obtain a closed form of the above equation. Also note that the estimate depends on knowing or being able to estimate the scale parameter θ.

2.2.3.3 The Kernel Prior:

Here, we shall assume the non-parametric kernel probability density equation (2.2.3.3) as the prior PDF of β; using the likelihood equation (2.2.1.5), the posterior density of β is given by:

$$h_k(\bar{t}|\beta) = \frac{\exp\left\{\left(\frac{t_n}{\theta}\right)^{\beta}\right\} \frac{\beta^n}{\theta^{n\beta}} \prod_{i=1}^{n}(t_i)^{\beta-1} \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{\beta-\beta_i}{h}\right)}{\int_0^{\infty} \exp\left\{\left(\frac{t_n}{\theta}\right)^{\beta}\right\} \frac{\beta^n}{\theta^{n\beta}} \prod_{i=1}^{n}(t_i)^{\beta-1} \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{\beta-\beta_i}{h}\right) d\beta}. \tag{2.2.3.3.1}$$

Thus, the kernel Bayesian estimate of the key parameter β in $V(t;\beta,\theta)$ under the H-T loss function, using equation (2.2.2.3) and equation (2.2.3.1.1), is given by:

$$\hat{\beta}_{K.HT} = \frac{1}{f_1 + f_2} \ln[\frac{\int_{\gamma}^{\infty} \exp\{f_1\beta\} h_k(\bar{t}|\beta)d\beta}{\int_{\gamma}^{\infty} \exp\{-f_2\beta\} h_k(\bar{t}|\beta)d\beta}]. \tag{2.2.3.3.2}$$

We must rely on a numerical estimation because we cannot obtain a closed form solution for $\hat{\beta}_{K.HT}$. In addition, the kernel function, $K(u)$, and bandwidth, h, will be chosen to minimize the asymptotic mean integrated squared error (AMISE) given by:

$$AMISE\left(\hat{f}(\beta)\right) = \int E\left[\left(\hat{f}(\beta) - f(\beta)\right)^2\right] d\beta, \tag{2.2.3.3.3}$$

where $\hat{f}(\beta)$ and $f(\beta)$ are the estimated probability density of β and the true probability density of β, respectively. Below are details of the analysis we conducted using Monte Carlo simulation to generate data governed by a PLP, followed by using actual data.

2.3 Results and Discussion

2.3.1 Numerical Simulation

A Monte Carlo simulation was used to compare the Bayesian (under H-T loss functions) and the MLE approaches. The parameter β of the intensity function for the PLP was calculated using numerical integration techniques in conjunction with a Monte Carlo simulation to obtain its Bayesian estimate. Substituting these estimates in the intensity function, we obtained the Bayesian intensity function estimates, from which the reliability function can be estimated.

For a given value of the parameter θ, a stochastic value for the parameter β was generated from the Burr PDF. For each pair of values of θ and β, 400 samples of 40 failure times that followed a PLP were generated. This procedure was repeated 200 times for three distinct values of θ. The procedure is summarized in the algorithm given by Figure 2.1.

For each sample of size 40, the Bayesian estimates and MLEs of the parameter were calculated when $\theta \in \{0.5, 1.7441, 4\}$. The comparison is based on the mean squared error (MSE) averaged over the $100,000$ repetitions. The results are given in Table 2.2.

Table 2.2: MSE for Bayesian estimates under the H-T loss function and MLE of β, for each assumed θ value.

θ	MSE of $\hat{\beta}$	MSE of $\hat{\beta}_{B.HT}$
0.5	0.0112436	0.000507356
1.7441	0.0110573	0.000516057
4	0.010961	0.000518632

It is observed that $\hat{\beta}_{B.HT}$ maintains a good accuracy, and is superior to $\hat{\beta}$ in estimating β for the different values of θ. For various sample sizes, the Bayesian estimate under the H-T loss function and the MLE of the parameter β were calculated and averaged over $10,000$ repetitions. Table 2.3 displays the simulated result of comparing a true value of β with respect to its MLE and Bayesian estimates for $n = 20, 30, \ldots, 160$.

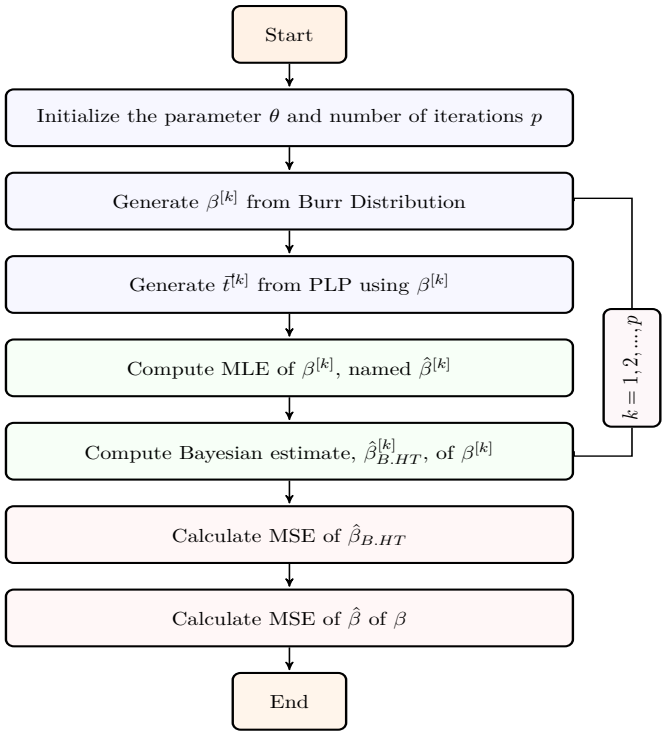


Figure 2.1.: Simulation to analyze Bayesian estimates of β for a given θ.

Table 2.3: Bayesian estimates, under H-T loss function, and MLEs for the parameter β= 0.7054 averaged over 10,000 repetitions

n	β_{Fixed}	$\hat{\beta}$	$\hat{\beta}_{B.HT}$
20	0.7054	0.784026	0.675263
30	0.7054	0.756617	0.690189
40	0.7054	0.743982	0.696467
50	0.7054	0.73531	0.699158
60	0.7054	0.729563	0.700642
70	0.7054	0.725977	0.70169
80	0.7054	0.723338	0.702382
100	0.7054	0.719117	0.703165
120	0.7054	0.716315	0.703585
140	0.7054	0.714821	0.70398
160	0.7054	0.713641	0.704244

Again, the Bayesian estimate is uniformly closer to the true value of β than its MLE, even for a very small sample size of $n = 20$. A graphical comparison of the true value of β along with the Bayesian and MLE estimates as functions of sample size is given by Figure 2.2.

Figure 3.2 shows the MLE of β tends to overestimate whereas the Bayesian estimate tends to underestimate the true value of β, particularly when considering small sample sizes. The MSEs of

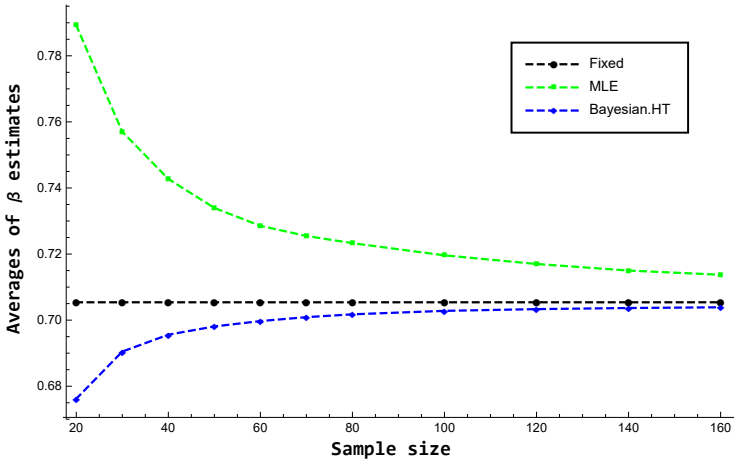


Figure 2.2.: β estimates versus sample size.

the MLE and Bayesian estimates of β is given below by Figure 2.3.

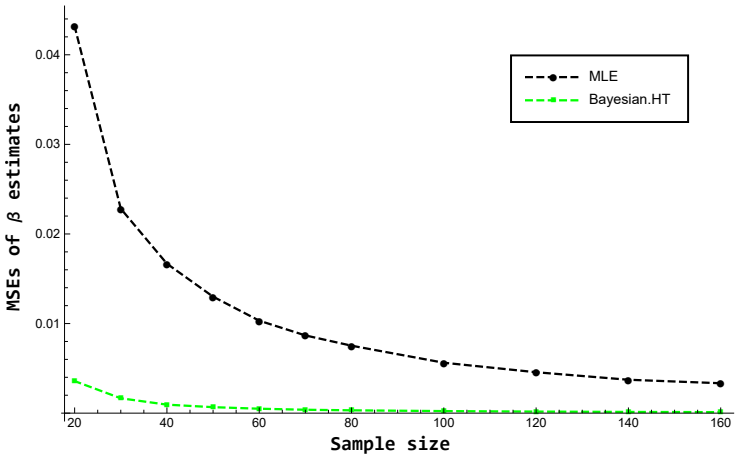


Figure 2.3.: β estimates versus sample size.

Regardless of sample size, the MSE of the Bayesian estimate of the key parameter β is significantly smaller than the MSE of the MLE of β (Figure 2.3).

Since the Bayesian estimate under the H-T loss function for β is better than its MLE, Molinares and Tsokos proposed to adjust the MLE of the parameter θ using (3.2.1.6) with a Bayesian estimate of β instead of its MLE, both of which are needed to estimate the $V(t;\beta,\theta)$, as given below:

$$\hat{\theta}_{B.HT} = \frac{t_n}{n^{1/\hat{\beta}_{B.HT}}}. \tag{2.3.1.1}$$

For various sample sizes and the same β ($\beta = 0.7054$), the Bayesian MLE and MLE of the

parameter θ and their corresponding MSEs were computed, averaging over the $10,000$ repetitions, using the MLE of θ ($\theta = 1.7441$) of the Crow data.

Table 2.4: MLE and Bayesian estimates under the H-T loss function for the parameter θ= 1.7441 averaged over 10,000 repetitions.

n	θ	$\hat{\theta}_{MLE}$	$\hat{\theta}_{B.HT}$
20	1.7441	3.17139	1.36422
30	1.7441	2.908	1.5097
40	1.7441	2.73107	1.58115
50	1.7441	2.59245	1.61985
60	1.7441	2.48865	1.64406
70	1.7441	2.41782	1.66084
80	1.7441	2.36522	1.67294
100	1.7441	2.26774	1.68902
120	1.7441	2.20117	1.69923
140	1.7441	2.15539	1.70659
160	1.7441	2.11872	1.71193

Table 2.4 shows the inferior performance for the MLE of θ and the slow convergence of its average values to $\theta = 1.7441$, whereas the adjusted estimate of θ ($\hat{\theta}_{B.HT}$) using the Bayesian estimate of β under the H-T loss function performed better in estimating the true value of θ. The MLE of and the Bayesian MLE estimate of θ had a tendency to overestimate and underestimate the parameter θ, respectively.

As expected, based on the Bayesian influence on β, $\hat{\theta}_{B.HT}$ is a better estimate than the MLE of θ ($\hat{\theta}$). This can be seen in Figure 2.4 where the MSEs of θ estimates were ploted against various sample sizes, which demonstrates the excellent performance of $\hat{\theta}_{B.HT}$.

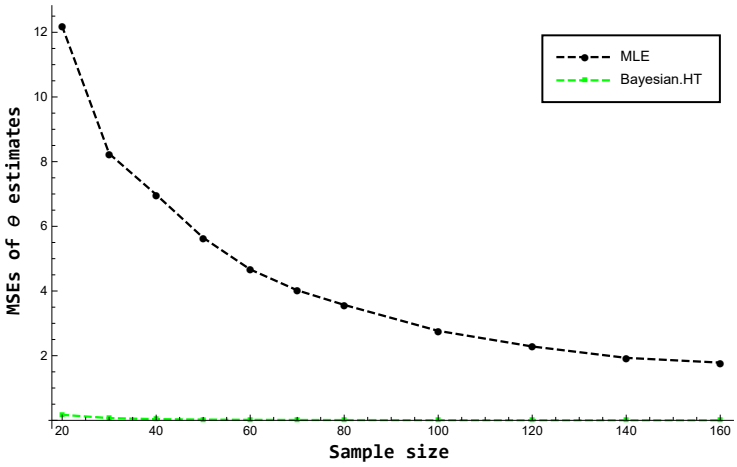


Figure 2.4.: MSE of θ estimates versus sample size.

We also computed the proposed estimate for the parameter θ ($\hat{\theta}_{B.HT}$) and its MSE over $100,000$ repetitions for different values of θ (0.5, 1.7441, 4) and sample size $n = 40$. The results are given by Table 2.5. The θ values (including 1.7441) were selected for this simulation are smaller and larger than the MLE of θ of the Crow data.

Table 2.5: MSE of θ estimates using Bayesian of β under the H-T loss function.

θ	$\hat{\theta}_{B.HT}$	MSE of $\hat{\theta}_{B.HT}$
0.5	0.503314	**0.00691164**
1.7441	1.7509	**0.0827802**
4	4.01025	**0.439035**

Table 2.5 shows that the $\hat{\theta}_{B.HT}$ performed well for the selected θ values. This is particularly true for the small and medium value of θ values.

For a fixed value of $\theta = 1.7441$ and a sample size similar to the size of the collected data, $n = 40$, the estimates of the intensity function $\hat{V}_{MLE}(t)$ and $\hat{V}_{B.HT}(t)$ were obtained using $\hat{\beta}$ and $\hat{\beta}_{B.HT}$, respectively, in equation (3.1.0.2). That is,

$$\hat{V'}_{MLE}(t) = \frac{\hat{\beta}}{\theta}\left(\frac{t}{\theta}\right)^{\hat{\beta}-1},\ \theta > 0, t > 0, \tag{2.3.1.2}$$

and

$$\hat{V'}_{B.HT}(t) = \frac{\hat{\beta}_{B.HT}}{\theta}\left(\frac{t}{\theta}\right)^{\hat{\beta}_{B.HT}-1},\ \theta > 0, t > 0. \tag{2.3.1.3}$$

Their graphs (Figure 2.5) reveal the superior performance of $\hat{V'}_{B.HT}(t)$.

In order to obtain Bayesian estimate of the intensity function, $\hat{V^*}_{B.HT}$, we substituted the Bayesian estimates of β and its corresponding θ MLE in equation (2.1.2). That is,

$$\hat{V^*}_{B.HT}(t) = \frac{\hat{\beta}_{B.HT}}{\hat{\theta}}\left(\frac{t}{\hat{\theta}}\right)^{\hat{\beta}_{B.HT}-1},\ t > 0. \tag{2.3.1.4}$$

The MLE of the intensity function, $\hat{V}_{MLE}$, is obtained using the MLEs of β and θ. That is,

$$\hat{V}_{MLE}(t) = \frac{\hat{\beta}}{\hat{\theta}}\left(\frac{t}{\hat{\theta}}\right)^{\hat{\beta}-1},\ t > 0. \tag{2.3.1.5}$$

The Bayesian MLE of the intensity function under the influence of the Bayesian estimate of β, denoted by $\hat{V}_{B.HT}$, is obtained by substituting $\hat{\beta}_{B.HT}$ and $\hat{\theta}_{B.HT}$ in equation (3.1.0.2):

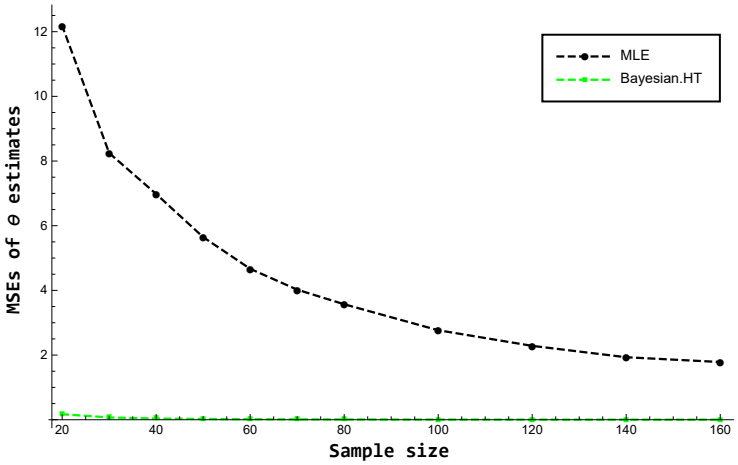


Figure 2.5.: Graph for $\theta = 1.7441$ and the corresponding β Bayesian estimate and MLE's used in $\hat{V}'_{MLE}$ and $\hat{V}'_{B.HT}$, estimates of $V(t;\beta,\theta)$ with n = 40.

$$\hat{V}_{B.HT}(t) = \frac{\hat{\beta}_{B.HT}}{\hat{\theta}_{B.HT}} \left(\frac{t}{\hat{\theta}_{B.HT}} \right)^{\hat{\beta}_{B.HT}-1}, \; t > 0. \tag{2.3.1.6}$$

To measure the robustness of $\hat{V}_{B.HT}$ with respect to $\hat{V}_{MLE}$, we calculated the relative efficiency (RE) of the estimate $\hat{V}_{B.HT}$ compared to the estimate $\hat{V}_{MLE}$, which is defined as

$$RE(\hat{V}_{B.HT}, \hat{V}_{MLE}) = \frac{IMSE(\hat{V}_{B.HT})}{IMSE(\hat{V}_{MLE})} = \frac{\int_{-\infty}^{\infty} [\hat{V}_{B.HT}(t) - V(t)]^2 dt}{\int_{-\infty}^{\infty} [\hat{V}_{MLE}(t) - V(t)]^2 dt}. \tag{2.3.1.7}$$

If $RE = 1$, $\hat{V}_{B.HT}$ and $\hat{V}_{MLE}$ will be interpreted as equally effective. If $RE < 1$, $\hat{V}_{B.HT}$ is more efficient than $\hat{V}_{MLE}$, contrary to $RE > 1$, in which case $\hat{V}_{B.HT}$ is less efficient than $\hat{V}_{MLE}$.

Bayesian estimates and MLEs for the parameters β= 0.7054 and θ=1.7441 (Table 2.6), averaged over 10,000 repetitions, were used, for $n = 40$, to compare $\hat{V}_{B.TH}$ and $\hat{V}_{MLE}$ using equation (3.3.1.9).

Table 2.6: Averages of the Bayesian (under the H-T loss function) and MLE estimates of β and θ

β	$\hat{\beta}$	$\hat{\beta}_{B.HT}$	θ	$\hat{\theta}$	$\hat{\theta}_{B.HT}$
0.7054	0.743982	0.696467	1.7441	2.73107	1.58115

Table 2.6 above reveals that the averages of the Bayesian and Bayesian MLE estimates of the parameters β and θ under the H-T loss function, respectively, are closer to the true values than their corresponding MLE estimates. The results of the comparison among the $\hat{V}_{B.TH}$ and $\hat{V}_{MLE}$ using equation (3.3.1.9) are given in Tables 3.8 and 2.8.

Table 2.7: Intensity functions with Bayesian and MLE estimates for β and θ

$V(t)$	$\hat{V}_{MLE}$	$\hat{V^*}_{B.HT}$	$\hat{V}_{B.HT}$
$0.476465 \cdot t^{-0.2946}$	$0.352321 \cdot t^{-0.256018}$	$0.345946 \cdot t^{-0.303533}$	$0.5062 \cdot t^{-0.303533}$

The analytical forms of the $V(t)$, $\hat{V}_{MLE}$, $\hat{V^*}_{B.TH}$, and $\hat{V}_{B.TH}$ (Table 2.7) were derived by substituting the initialized values, MLE estimates of both parameters, Bayesian estimate β and MLE of θ, and Bayesian estimates, respectively.

Table 2.8: Relative efficiency of $\hat{V}_{B.HT}$ compared to $\hat{V}_{MLE}$.

$RE(\hat{V}_{B.HT}, \hat{V}_{MLE})$	$RE(\hat{V}_{B.HT}, \hat{V^*}_{B.HT})$
0.0761919	0.00550275

Table 2.8 shows the comparison result of $\hat{V}_{B.HT}$ and $\hat{V}_{MLE}$, where the $RE(\hat{V}_{B.HT}, \hat{V}_{MLE})$ is less than 1, which implies that the intensity function using $\hat{\beta}_{B.HT}$ is more efficient than the intensity function under $\hat{\beta}_{MLE}$, establishing the superior relative efficiency of Bayesian estimates under the H-T loss function over MLE estimates. The corresponding graph for the intensity functions is given by Figure 2.6. In addition, $\hat{V^*}_{B.HT}$ computed using a Bayesian estimate for β and MLE estimate for θ, is less efficient compared to $\hat{V}_{MLE}$, and $\hat{V}_{B.HT}$.

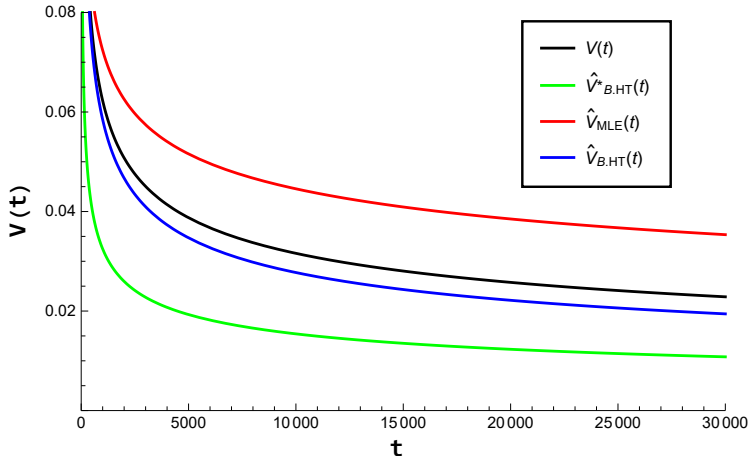


Figure 2.6.: Estimates of the intensity function using values in Table 3.7, n = 40.

The $\hat{V}_{B.HT}$ is a better estimate of $V(t)$, compared to the $\hat{V}_{MSE}$ and $\hat{V^*}_{B.HT}$. Based on the results of this section, the Bayesian estimates under the H-T loss function will be used to analyze real data in the following section.

2.3.2 Using Real Data

Using the software reliability growth data from Table 3.1, we computed $\hat{\beta}_{B.HT}$ and the adjusted estimate of θ ($\hat{\theta}_{B.HT}$) in order to obtain a Bayesian estimate of the intensity function under the H-T loss function. We followed the algorithm given by Figure 2.7 to obtain the Bayesian intensity function for the given real data.

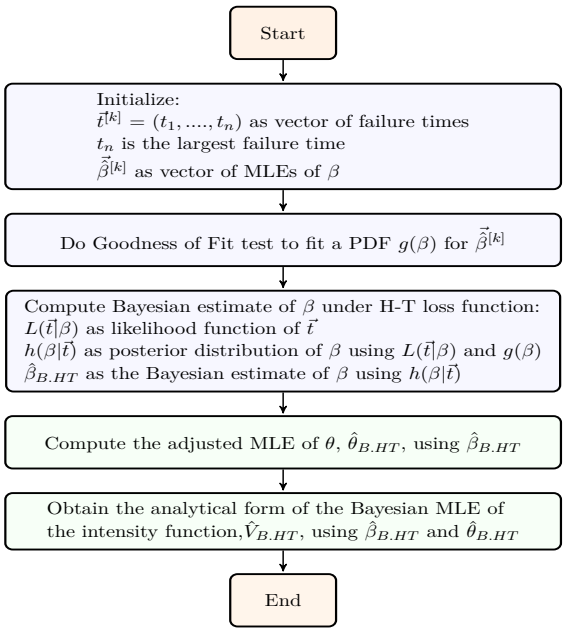


Figure 2.7.: Estimate of the intensity function using Crow data in Table 3.1

.

For the failure data of Crow, provided in Table 3.1, $\hat{\beta}_{B.HT}$ is 0.501199 and $\hat{\theta}_{B.HT}$ is 2.07144. Therefore, with the use of $\hat{\theta}_{B.HT}$, the Bayesian MLE of the intensity function for the data is given by:

$$\hat{V}_{B.HT}(t) = 0.347933 \cdot t^{-0.498801},\ t > 0. \tag{2.3.2.1}$$

A graphical display of $\hat{V}_{B.HT}(t)$ is given by Figure 2.8.

Figure 2.8 shows the Bayesian MLE estimate of the intensity function ($V(t;\beta,\theta)$) under the H-T loss function ($\hat{V^*}_{B.HT}$), which indicates the improvement of the software reliability over time.

To obtain a Bayesian MLE for the reliability function under the H-T loss function, we use this Bayesian estimate for the intensity function. The analytical form for the corresponding Bayesian

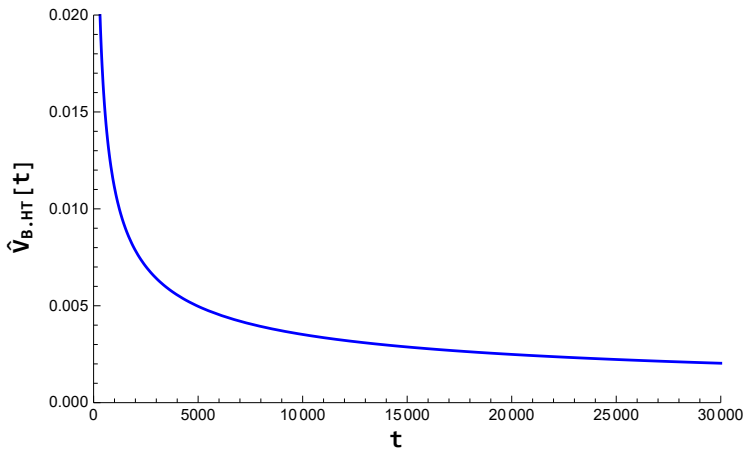


Figure 2.8.: Estimate of the intensity function for the real data in Table 3.1, using $\hat{\beta}_{B.HT}$ and $\hat{\theta}_{B.HT}$.

reliability estimate, based on the real data, is given by:

$$\hat{R}_{B.HT}(t_i|t_1,..,t_{i-1}) = \exp\left\{-0.347933\int_{t_{i-1}}^{t_i} x^{-0.498801}dx\right\},\ t_i > t_{i-1} > 0. \tag{2.3.2.2}$$

Thus far, we demonstrated not only the applicability of the Bayesian analysis to the PLP, but also, using real data, the superiority of its performance and influence compared to the MLE of the parameters β and θ, respectively, assuming the Burr PDF is the prior knowledge of the key parameter β. Next section, we study the sensivity of the prior selections, in which an engineer might lack a prior knowledge of parameter β.

2.3.3 Sensitivity of Prior Selection

In the implementation of the simulation procedure we followed Algorithm 1. Random failure times (time to failures) distributed according to the PLP are simulated for a realization of the stochastic scale parameter β, which follows a Burr type XII probability distribution. Informative parametric priors were considered, such as inverted gamma and Burr probability distributions, whereas the Jefferys prior was chosen as a non-informative prior. In addition, non-parametric priors like kernel density were applied during the sensitivity analysis study. Kernel density estimation depends on several variables, including sample size, bandwidth, and kernel function. In this study, the optimal bandwidth (h^*) and kernel function were chosen such that the asymptotic mean integrated squared error (AMISE) is minimized. The simplified analytical form of AMISE, equation (2.3.3.1), is given

by:

$$AMISE\left(\hat{f}(\beta)\right) = \frac{C(K)}{n \cdot h} + (\frac{1}{4} \cdot h^4 \cdot k_2^2 \cdot R\left(f^{(2)}(\beta)\right)) \tag{2.3.3.1}$$

Where:

- C(K)= $\int (K(u))^2 du$,
- n: sample size,
- h: bandwidth,
- $k_2 = \int_{-\infty}^{+\infty} u^2 \cdot K(u) du$,
- $f^{(2)}(\beta)$ is the second derivative of Burr PDF,
- $R(f^{(2)}(\beta))= \int (f^{(2)}(\beta))^2 d\beta$,
- $h^* = \left[\frac{C(K)}{k_2^2 \cdot R\left(f^{(2)}(\beta)\right)}\right]^{1/5} \cdot n^{-1/5}$, (Silverman [90]).

Table 2.9 shows the calculations of AMISE using the optimal bandwidth (h^*), with respect to different samples sizes for each kernel function considered in this study, namely Epanechnikov, cosine, biweight, triweight, Gaussian, triangle, uniform, tricube, and logistic kernel functions.

Table 2.9: Calculations of the AMISE with respect to different sample size, optimal bandwidth, and kernel function

Kernel function	Sample size				
	50	100	150	200	500
Epanechnikov	**0.362827**	**0.208389**	**0.150662**	**0.119688**	**0.0575042**
Cosine	0.362986	0.208481	0.150728	0.119741	0.0575295
Biweight	0.364607	0.209412	0.151401	0.120275	0.0577863
Triweight	0.36674	0.210637	0.152286	0.120979	0.0581243
Gaussian	**0.377644**	**0.2169**	**0.156814**	**0.124576**	**0.0598525**
Triangle	0.366972	0.21077	0.152383	0.121056	0.0581612
Uniform	0.384675	0.220938	0.159734	0.126895	0.0609669
Tricube	0.363433	0.208737	0.150913	0.119888	0.0576002
Logistic	0.399132	0.229241	0.165737	0.131665	0.0632582

The minimum AMISE corresponds to the Epanechnikov kernel function ($K(u)=\frac{3}{4}(1-u^2)I_{|u|\leq 1}$). In addition to the Epanechnikov kernel function, the Gaussian kernel function ($K(u)=\frac{1}{\sqrt{2\pi}}\exp\left(\frac{-u^2}{2}\right)I_{\mathbb{R}}$) was also used in the calculation since it is commonly used for its analytical tractability.

Numerical integration techniques were used to compute the Bayesian estimates of β under the H-T loss function, according to the equations in Section **2.2.3**, for each of the five densities and three distinct values of θ. Samples of sizes of 20, 30, 40, 50, 60, 70, 80, 100, 120, 140, and 160 were generated, where the parameter θ was assumed to be the MLE of θ (1.7441) using Crow's data. The results, for $10,000$ repetitions, are presented by Table 2.10 and Figure 2.9.

Table 2.10: Averages of the Bayesian estimates (using the subject prior PDFs), under H-T loss functions, and MLEs of the parameter β over 10,000 repetitions

n	β	$\hat{\beta}_{MLE}$	$\hat{\beta}_{B.HT}$	$\hat{\beta}_{IG.HT}$	$\hat{\beta}_{J.HT}$	$\hat{\beta}_{KG.HT}$	$\hat{\beta}_{KE.HT}$
20	0.7054	0.781303	0.674095	0.799535	0.710047	0.675698	0.675693
30	0.7054	0.753465	0.688951	0.762536	0.707341	0.693849	0.693881
40	0.7054	0.740665	0.695087	0.745074	0.70651	0.698597	0.698631
50	0.7054	0.732738	0.697886	0.734888	0.705885	0.699751	0.69976
60	0.7054	0.728669	0.699823	0.728731	0.705847	0.700727	0.700699
70	0.7054	0.725499	0.70101	0.724471	0.705776	0.70148	0.701405
80	0.7054	0.723539	0.701979	0.721552	0.705882	0.702323	0.7022
100	0.7054	0.719621	0.70285	0.717324	0.705664	0.70338	0.703159
120	0.7054	0.717465	0.703468	0.71478	0.705632	0.704258	0.703959
140	0.7054	0.716224	0.703954	0.713141	0.705692	0.704928	0.70457
160	0.7054	0.714739	0.704188	0.711865	0.705629	0.7053	0.704893

It can be observed that the Bayesian estimate of β for the Jeffreys prior PDF under the H-T loss function converges to the true value faster than other prior PDFs; followed very closely by the Bayesian estimates using the kernel prior PDFs, which tend to converge faster than the inverted gamma prior PDF and are superior to the maximum likelihood approach.

Figure 2.9 presents the graphical behavior of the MLE and Bayesian estimates represented by their average values, where the average values of the Jeffery Bayesian estimates were the closest to the true value of β for all sample sizes. Followed by the average values of the kernel Bayesian and the Burr Bayesian estimates, which they tend to have similar behavior in estimating for sample sizes of 50 and larger. Bayesian estimates seems to have insignificant difference for sample sizes of 100 and larger except the inverted gamma Bayesian estimate which performed as poor as the MLE in estimating the key parameter β.

Table 2.11 shows the MSE of the Bayesian estimates of β under the H-T loss function and the subject prior PDFs with respect to various sample sizes.

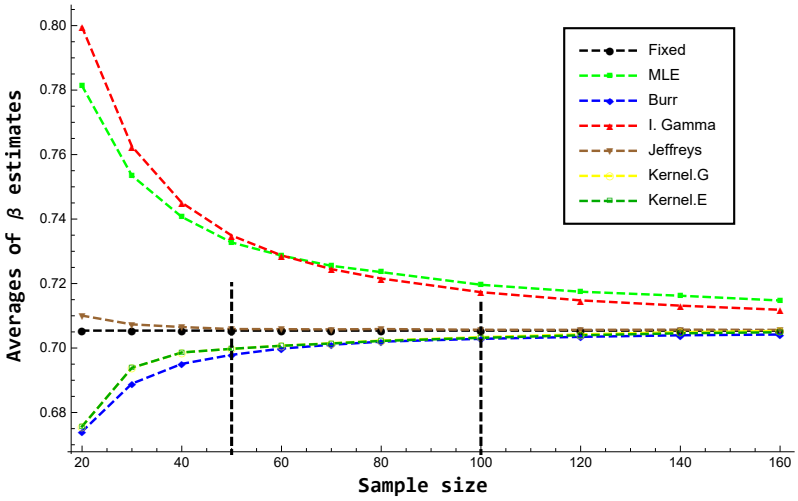


Figure 2.9.: Averages of β estimates for different sample sizes.

Table 2.11: MSE of the Bayesian estimates, under the H-T loss functions, and MLEs of the parameter β averaged over 1000 repetitions for different priors.

n	MSE of $\hat{\beta}_{MLE}$	MSE of $\hat{\beta}_{B.HT}$	MSE of $\hat{\beta}_{IG.HT}$	MSE of $\hat{\beta}_{J.HT}$	MSE of $\hat{\beta}_{KG.HT}$	MSE of $\hat{\beta}_{KE.HT}$
20	0.0417095	0.00362378	0.0115968	**0.00285493**	0.0037072	0.00371019
30	0.0238951	0.00167959	0.00460464	0.00139859	0.00166486	0.00166402
40	0.016305	0.00100242	0.00242657	0.000885863	0.000976117	0.000974269
50	0.0123248	0.00067513	0.00146266	0.000612733	0.000653313	0.000653343
60	0.00997557	0.000502602	0.000998874	0.000467943	0.000476869	0.000478856
70	0.00831476	0.000394076	0.000726943	0.000372566	**0.00036205**	0.000365206
80	0.00701888	0.000320076	0.000561229	0.000307048	0.000284152	0.000288374
100	0.00546558	0.000227799	0.000358957	0.000220557	0.000194561	0.000199899
120	0.00452469	0.000177734	0.000259043	0.000173553	0.000149927	0.000156087
140	0.00386396	0.000145074	0.00020085	0.000142729	0.000121572	0.000128407
160	0.00329065	0.000120261	0.00015916	0.000118617	**0.00010159**	0.000108463

For a sample size of 20, the Jeffery Bayesian estimate of β is the best estimate based on the MSE, followed by the Burr and kernel Bayesian estimates (Table 2.11). The MSE of the Gaussian Kernel Bayesian esimate was the lowest for moderate to large sample sizes ($n = 70, .., 160$). The inverted gamma Bayesian estimate had the lowest performance compared to other Bayesian estimates. The MSE of the MLE of the key parameter β indicated its poor performance compared to the Bayesian estimates.

Figure 2.10 shows the MLE and Bayesian estimates of β. It can be observed the Bayesian estimates under the H-T loss function are superior to the MLE of the key parameter β for all sample sizes considered in this study. Moreover, the Bayesian estimates are presented without MLE in Figure 2.11 to look closely at the performance of Bayesian estimates, under the H-T loss

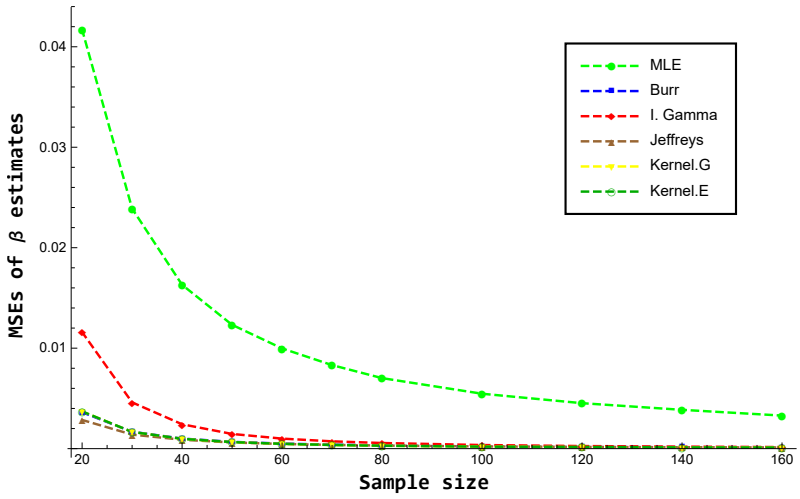


Figure 2.10.: MSEs of β estimates for different sample sizes.

function and the subject prior PDFs, based on their MSEs.

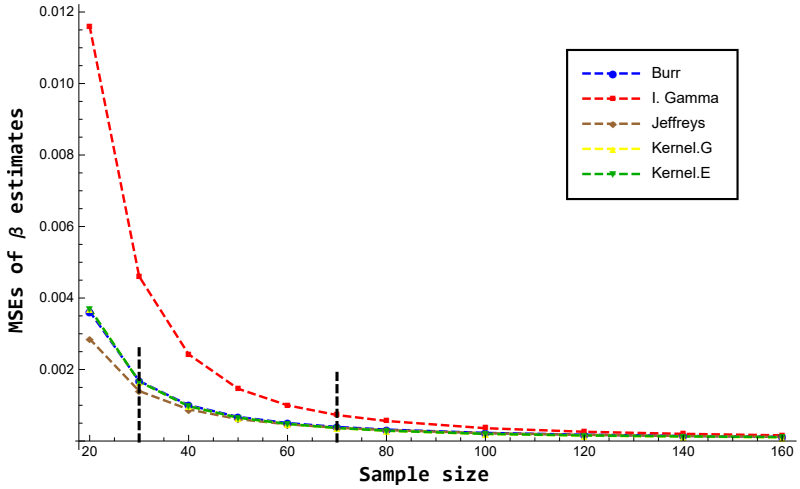


Figure 2.11.: MSEs of β Bayesian estimates for different sample sizes.

The MSEs of the Jefferys prior, for various sample sizes, were the smallest, followed by MSEs of the kernels and Burr prior PDFs. Bayesian estimates using Gaussian and Epanechnikov kernel probability densities as priors performed similarly, whereas the Bayesian estimate using the inverted gamma PDF had the lowest performance compared to other Bayesian estimates. It also shows that for a sample size of $n = 30$ and larger, the Bayesian estimates using Burr, Jeffery, and kernel PDFs are similar in their convergence to the true value. For a sample size of $n = 70$ and larger, all Bayesian estimates tend to converge to the true value of the key parameter β. The Bayesian

estimates of β under Jefferys and inverted gamma prior PDFs tend to overestimate β, whereas Burr and kernel prior PDFs tend to underestimate β.

For each sample of size 40 based on Monte Carlo simulation, the Bayesian estimates and MLEs of β were calculated when $\theta \in \{0.5, 1.7441, 4\}$. The comparison is based on the MSE averaged over the $2,000$ simulated samples. The results are given by Table 2.12.

Table 2.12: MSE of β Bayesian estimates with Burr, Jeffreys, inverted gamma, and kernel PDFs as priors under the H-T loss function. MSE of MLE estimate of the parameter β in an NHPP with samples of n = 40 and different values of the parameter θ.

θ	MSE of $\hat{\beta}_{MLE}$	MSE of $\hat{\beta}_{B.HT}$	MSE of $\hat{\beta}_{IG.HT}$	MSE of $\hat{\beta}_{J.HT}$	MSE of $\hat{\beta}_{KG.HT}$	MSE of $\hat{\beta}_{KE.HT}$
0.5	0.0106298	**0.000520244**	0.00229073	0.000577579	0.000535304	0.000534979
1.7441	0.00996687	0.000524716	0.00225675	0.000585273	**0.000516497**	0.000516599
4	0.0112937	0.000558022	0.00246984	0.000632173	**0.000556695**	0.000556804

It can be observed that all Bayesian estimates are superior to MLE ($\hat{\beta}$) in estimating β, with sample size n = 40, while maintaining a consistent behavior for the different values of θ (Table 2.12). For small value of θ, Jeffrey Bayesian estimate had the lowest MSE value, whereas the MSE of the Gaussian kernel Bayesian estimate was the lowest for moderate and large values of θ.

For the case in which we misleadingly assumed the true probability distribution of the key parameter β, we found that the Jeffreys and kernel Bayesian estimates of β had the best performance compared to the inverted gamma Bayesian estimate of β, indicating that the Bayesian estimate of β is sensitive to the choice of its prior PDF.

As expected, the adjusted MLE of θ produced a better estimate under the mentioned Bayesian influence with respect to the H-T loss function. The average values of the MLE and Bayesian estimates of θ using the MLE and Bayesian estimates of β for various sample sizes, respectively, are presented by Figure 2.12. where the average values of the Jeffery Bayesian estimates were the closest to the true value of θ for all sample sizes. Followed by the average values of the kernel Bayesian and the Burr Bayesian estimates, which tend to have similar behavior in estimating for sample sizes of 40 and larger. When considering sample sizes of 100 and larger, the Bayesian estimates seems to have insignificant difference except for the inverted gamma Bayesian estimate which which still performs better than the MLE in estimating the parameter θ.

The MSE of θ estimates using MLE and Bayesian estimates of β, namely $\hat{\theta}_{MLE}$, $\hat{\theta}_{B.HT}$, $\hat{\theta}_{IG.HT}$, $\hat{\theta}_{J.HT}$, $\hat{\theta}_{KG.HT}$, and $\hat{\theta}_{KE.HT}$, are shown in Table 2.13 with various sample sizes. For a small sample, moderate, and large sample sizes, $n = 20, 70, 160$ respectively, the Jeffrey Bayesian estimate of θ ($\hat{\beta}_{J.HT}$) performed better than the other estimates, followed by the kernel Bayesian and Burr

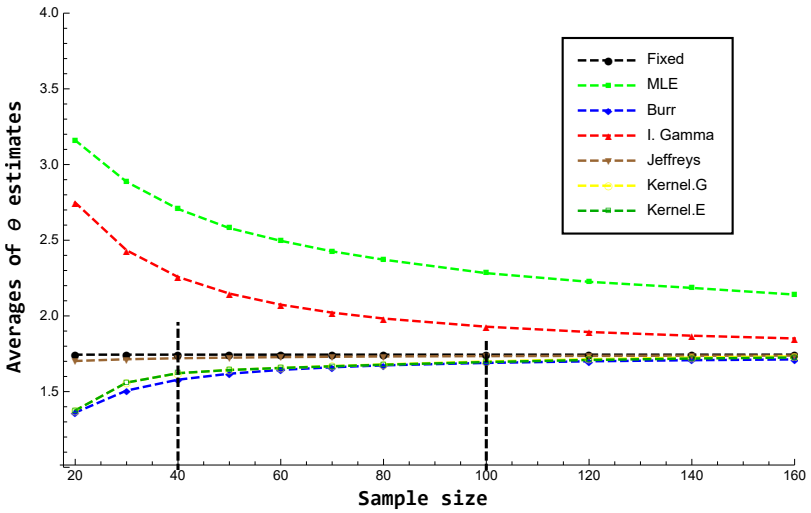


Figure 2.12.: Averages of MLEs of θ using the Bayesian estimates of the parameter β with respect to different priors.

Bayesian estimates.

Table 2.13: MSE of the Bayesian estimates, under the H-T loss functions, and MLEs of the parameter θ averaged over 1000 repetitions for different priors.

n	MSE of $\hat{\theta}_{MLE}$	MSE of $\hat{\theta}_{B.HT}$	MSE of $\hat{\theta}_{IG.HT}$	MSE of $\hat{\theta}_{J.HT}$	MSE of $\hat{\theta}_{KG.HT}$	MSE of $\hat{\theta}_{KE.HT}$
20	11.5735	0.175991	1.0789	**0.0310651**	0.174974	0.175081
30	8.6721	0.0726953	0.50118	0.0166961	0.0538952	0.0537907
40	6.67926	0.0378416	0.27733	0.0107884	0.0264298	0.026306
50	5.45282	0.0233418	0.171955	0.00777231	0.0182648	0.018213
60	4.55577	0.015811	0.115456	0.00593216	0.0139622	0.0139218
70	3.92543	0.0114843	0.0820851	**0.0047144**	0.0110305	0.0109659
80	3.38362	0.00867134	0.0610179	0.00382098	0.00880575	0.00869148
100	2.73005	0.00566115	0.0369537	0.0027899	0.00591856	0.00572456
120	2.33339	0.00402454	0.0245551	0.00215943	0.004114	0.00387668
140	2.03741	0.00302931	0.0174259	0.00174116	0.003041	0.00277349
160	1.75235	0.00238185	0.0129141	**0.0014346**	0.0023649	0.00206666

While the inverted gamma Bayesian estimate of θ outperformed that of the MLE, it did not perform as well as the other Bayesian estimates based on their MSE values across all sample size. This indicates the MLE of θ had the poorest performance compared to the Bayesian estimates.

The MSEs of θ estimates using the MLE and Bayesian estimates of the parameter β with respect to different priors is displayed in Figure 2.13, from which it can be seen that the MLE of θ was extrmely weak estimator since it has the largest MSEs across the sample sizes. The adjusted θ estimates were were displayed without the MLE estimates by Figure 2.14.

It can be noted that the θ estimate using the inverted gamma Bayesian estimate of β had the lowest performance compared to other estimates that used Bayesian estimates of β. In addition, above the sample size $n = 40$, the estimate of θ using Burr kernels Bayesian estimates of β performed

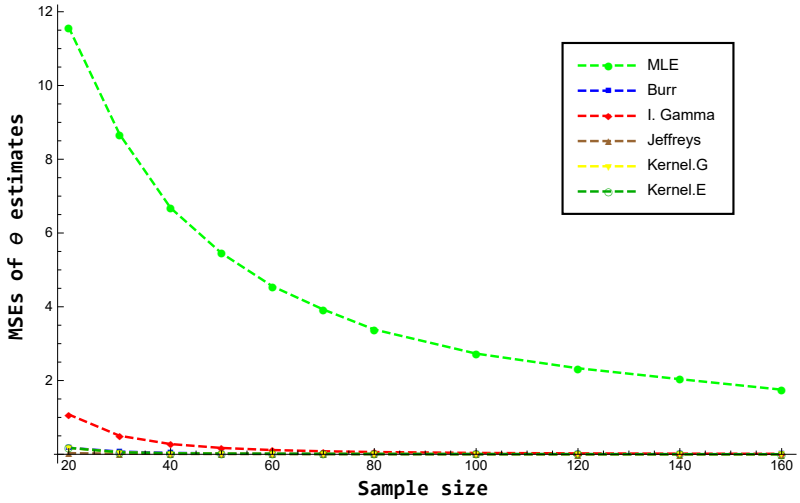


Figure 2.13.: MSE of the MLEs of θ using MLE and Bayesian estimates of β with respect to different prior β

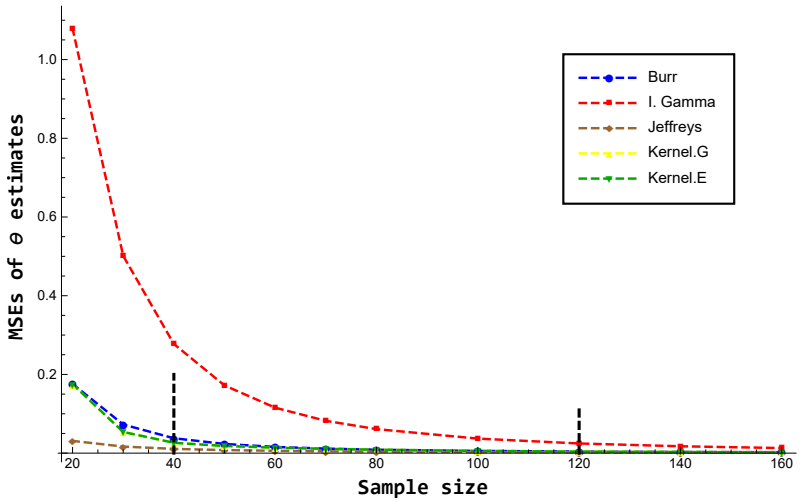


Figure 2.14.: MSE of the MLEs of θ using Bayesian estimates of β with respect to different prior β.

similarly in estimating the true value of θ. All θ estimates using Bayesian estimates of β tend to converge to the true value in similar trajectories, whereas the θ estimate using the Jeffrey Bayesian estimate of β converges slightly faster.

Therefore, the Bayesian analysis to the PLP under the H-T loss function is sensitive to the prior selection. Based on this finding, an engineer, for example, is recommended to use the Jeffreys or kernel PDFs if he/she lacks any prior knowledge of β.

2.4 Contributions

We developed the analytical Bayesian form of the key parameter β, under the H-T loss function, in the intensity function, where the underlying failure distribution is the PLP that is used for software reliability assessment.

The behavior of β is characterized by the Burr type XII probability distribution. Real data and numerical simulation were used to illustrate the efficiency improvement in the estimation of the intensity function of PLP under the H-T loss function ($\hat{V}_{B.HT}(t)$). Based on $100,000$ samples of software failure times, using Monte Carlo simulations and sample size of 40, we found that the Bayesian estimate of β under the H-T loss function performed better than the MLE of β with respect to three different values of θ. Even for different sample sizes similar results were achieved using β=0.7054, θ=1.7441, and averaged over $10,000$ samples of software failure times.

Because the MLE of the second parameter (θ) in the intensity function depends on the estimate of β, the adjusted estimate of θ $\hat{\beta}_{B.HT}$ performed better than MLE of θ. Moreover, by comparing the relative efficiency metric, using Bayesian estimates of β under the H-T loss function, and Bayesian MLE of θ ($\hat{V}_{B.HT}(t)$), we found that $\hat{V}_{B.HT}(t)$ is more efficient in estimating the intensity function $V(t;\beta,\theta)$.

We answered the second research question: Is the Bayesian estimate of the key parameter, β, using the H-T loss function in the PLP, sensitive to the selection of the prior, whether parametric or non-parametric? Using the proposed algorithm, the Bayesian estimate of β under the H-T loss function and Burr PDF as a prior performed better than the other Bayesian estimates for a small value of θ. The Bayesian estimate of β under H-T loss function and Gaussian kernel PDF as a prior performed better compared to Bayesian estimates for moderate and large values of θ. MLE of β had poor performance compared to Bayesian estimates using the subject prior PDFs. The Bayesian estimate of β under H-T loss function and inverted gamma PDF as a prior performed poorly compared to other Bayesian estimates using Burr, Jeffrey, Gaussian, and Epanechnikov PDFs as priors. Even for different sample sizes, the MLE of β has the poorest performance compared to Bayesian estimates using different prior PDFs. For small to moderate sample sizes, the Bayesian estimate of β under H-T loss function and Jeffrey PDF as a prior has the best performance based on the MSE. For moderate to large sample sizes, the Bayesian estimate of β under H-T loss function and Gaussian kernel PDF as a prior has the best performance based on the MSE.

The Bayesian estimates under the H-T loss function and the parametric and non-parametric

priors were used to compute the adjusted estimate of θ. The adjusted θ estimate using the Jeffery Bayesian estimate of β performed best compared to the MLE and other Bayesian estimates for various sample sizes.

The Bayesian analysis approach under Higgins-Tsokos loss function is superior to the maximum likelihood approach in estimating the reliability function of the Power Law Process. The results of this study have the potential to contribute not only to the reliability analysis field but also to other fields that employ the Power Law Process.

Chapter 3

The Effectiveness of The Squared Error And Higgins-Tsokos Loss Functions On The Bayesian Reliability Analysis Of Software Failure Times Under The Power Law Process

In this chapter, we investigate the effectiveness, in Bayesian Analysis, of using the commonly used squared-error (S-E) loss function versus the Higgins-Tsokos (H-T) loss function that puts the loss at the end of the process, for modeling software failure times. To accomplish this, we used the underline failure distribution to be the Power Law Process subject to using Burr PDF as a prior of the key parameter β. In addition, we utilize both loss functions to perform sensitive analysis of the prior selections. We perform parametric and non-parametric priors, namely Burr, Inverted Gamma, Jeffery, and two Kernel PDFs. One article is published [5] based on findings of this Chapter.

The chapter is organized as follows, Section 2 describes the theory and development of the Bayesian reliability model. Section 3 presents the results and discussion. Section 4 are the conclusions.

3.1 Introduction

Reliability analysis of a software under development is a key to assess whether a desired level of a quality product is achieved. Specially, when a software package is considered, and is tested after each failure detection, and then corrected until a new failure is observed. Over the past few decades, the reliability analysis of a software package has been studied, where graphical and numerical metrics have been introduced. One of the earliest, Duane (1964) [29], who introduced a graph to assess the reliability of a software over time using its failure times. It has the cumulative failure rate and the time on the y-axis and x-axis, respectively. In this graph, one can conclude a software reliability improvement if a negative curve is observed whereas a positive curve means the software reliability is deteriorating. On other hand, a horizontal line indicates that the software reliability is stable. The failure numbers $N(t)$ in time interval $(0, t]$ is considered a Poisson counting process after satisfying the following conditions:

1. $N(t=0)=0$.

2. Independent increment (counts of disjoint time intervals are independent).

3. It has an intensity function

$$V(t)=\lim_{\Delta t\to 0}\frac{P(N(t,t+\Delta t)=1)}{\Delta t}.$$

4. Simultaneous failures do not exist

$$(\lim_{\Delta t\to 0}\frac{P(N(t,t+\Delta t)=2)}{\Delta t}=0).$$

The probability of random value $N(t)=n$ is given by:

$$P(N(t)=n)=\frac{\exp\left\{-\int_0^t V(t)dt\right\}\left\{\int_0^t V(t)dt\right\}^n}{n!},\ t>0. \tag{3.1.0.1}$$

Crow (1974) proposed a Non-Homogeneous Poisson Process (NHPP) , which is a Poisson Process with a time varying intensity function, given by:

$$V(t)=V(t;\beta,\theta)=\frac{\beta}{\theta}\left(\frac{t}{\theta}\right)^{\beta-1},\ t>0,\ \beta>0,\ \theta>0, \tag{3.1.0.2}$$

with β and θ are the shape and scale parameters, respectively. This Non-Homogeneous Poisson Process is also known as the Power Law Process (PLP).

The joint probability density function (PDF) of the ordered failure times T_1, T_2, ..., T_n from a NHPP with intensity function $V(t;\beta,\theta)$ is given by:

$$f\left(t_1,..,t_n\right)=\prod_{i=1}^n V\left(t_i;\beta,\theta\right)exp\left\{-\int_0^w V\left(t;\beta,\theta\right)dt\right\}, \tag{3.1.0.3}$$

where w is the so-called stopping time; w $=t_n$ for the failure truncated case. Considering the failure truncation case, the conditional reliability function of the failure time T_n given $T_1=t_1$,

$T_2 = t_2$, $T_3 = t_3$,..., $T_{n-2} = t_{n-2}$, $T_{n-1} = t_{n-1}$ is a function of $V(t; \beta, \theta)$.

As a numerical assessment, the estimate of the key parameter β in the $V(t; \beta, \theta)$ has an important role in evaluating the reliability of a software package. When the estimates of β are less and larger than 1, they indicate that the software reliability is improving and decreasing, respectively. The PLP is reduced to a homogeneous Poisson process when the estimate of β equals to 1.
The NHPP has been used for analyzing software's failure times, and prediction of the next failure time. The subject model has been shown to be effective and useful not only in software reliability assessment [26, 100, 39, 21, 9, 40, 97, 82, 83, 84], but also in cybersecurity; the attack detection in cloud systems [61, 67], breast and skin cancer treatments' effectiveness, [98, 93, 45], respectively, finance; modeling of financial markets at the ultra-high frequency level [60], trnasportation; modeling passengers' arrivals [56, 86, 77, 63, 102], and in the formulation of a software cost model [58].

Since the conditional reliability function of the PLP is a function of the $V(t; \beta, \theta)$, which includes the key parameter β. That being said updating the estimation methods for the key parameter will affect positively the $V(t; \beta, \theta)$ and the software reliability estimation, and therefore help the structuring of maintenance strategies. The authors [65] and [4] obtained the Bayesian estimates of the parameter β under the the squared-error and Higgins-Tsokos loss functions, respectively, and compared them to their approximate maximum likelihood estimate (MLE). They also showed the superiority of the Bayesian estimates to the MLE of the key parameter β, and the improvement in the reliability assessment under the PLP.

To perform Bayesian analysis on a real world problem, one needs to justify the applicability of such analysis. Then, the analysis process starts by identifying the probability distribution of the failure times of a software under development, the prior PDF of the key parameter β, and a loss function. The analytical tractability have made the squared-error loss function commonly used, where it places more weight on the estimates that are far from the true value than the estimates close to true value. Higgins and Tsokos [47] proposed a new loss function that maintains the analytical tractability feature and places exponentially more weight on extreme estimates of the true value.

In the present study, we investigate the effectiveness, in Bayesian Analysis, of using the commonly used squared-error (S-E) loss function versus the Higgins-Tsokos (H-T) loss function that puts the loss at the end of the process, for modeling software failure times. To accomplish this, we used the underline failure distribution to be the Power Law Process subject to using Burr PDF as a prior

of the key parameter β. In addition, we utilize both loss functions to perform sensitive analysis of the prior selections. We perform parametric and non-parametric priors, namely Burr, Inverted Gamma, Jeffery, and two Kernel PDFs. Therefore, the primary objective of the study is to answer the following questions within a Bayesian framework:

1) How robust is the assumption of the squared-error loss function being challenged by the Higgins-Tsokos loss function in estimating the key parameter β of PLP for modeling software failure times?

2) Is the Bayesian estimate of the intensity function, $V(t;\beta,\theta)$, of the PLP sensitive to the selections of the prior (parametric and non-parametric) and loss function (Higgins-Tsokos and S-E loss functions)?

3.2 Theory and Bayesian Estimates

3.2.1 Review of the Analytical Power Law Process

The probability of achieving n failures of a given system in the time interval $(0,t]$ can be written as

$$P(x=n;t)=\frac{\exp\left\{-\int_0^t V(x)dx\right\}\left\{\int_0^t V(x)dx\right\}^n}{n!},\ t>0, \tag{3.2.1.1}$$

where $V(t)$ is the intensity function given by (3.1.0.2). The reduced expression is given by:

$$P(x=n;t)=\frac{1}{n!}\exp\left\{-\frac{t}{\theta}^{\beta}\right\}\frac{t}{\theta}^{n\beta}, \tag{3.2.1.2}$$

is the PLP that is commonly known as Weibull or Non- Homogeneous Poisson Process.

When the PLP is the underlying failure model of the failure times t_1, t_2, t_3,... , t_{n-1}, and t_n, the conditional reliability function of t_n given t_1, t_2, t_3,... , t_{n-1} can be written mathematically as a function of the intensity function, given by:

$$R(t_n|t_1,t_2,...,t_{n-1})=\exp\left\{\int_{t_{n-1}}^{t_n}-V(t;\beta,\theta)dt\right\},\ tn>t_{n-1}>0, \tag{3.2.1.3}$$

since it is independent of t_1, t_2, t_3, ... , t_{n-2}.

Note that the improvement in estimating the key parameter β in the $R(t_n|t_1,t_2,...,t_{n-1})$ of the

PLP, equation (3.2.1.3), will improve the reliability estimation.

The maximum likelihood estimation (MLE) of β is a function of the largest failure time and the MLE of θ is also a function of the MLE of β. Let T_1, T_2, ..., T_n denote the first n failure times of the PLP, where $T_l < T_2 < ... < T_n$ are measured in global time; that is, the times are recorded since the initial startup of the system. Thus, the truncated conditional probability distribution function, $f_i(t|t_1, ..., t_{i-1})$, in the Weibull process is given by

$$f_i(t|t_1, ..., t_{i-1}) = \frac{\beta}{\theta}\left(\frac{t}{\theta}\right)^{\beta-1} \exp\left\{-\frac{t}{\theta}^{\beta} + \frac{t_{i-1}}{\theta}^{\beta}\right\},\ t_{i-1} < t. \tag{3.2.1.4}$$

With $t = (t_1, t_2, ..., t_n)$, the Likelihood function for the first n failure times of the PLP $T_1 = t_1, T_2 = t_2, ..., T_n = t_n$ can be written as

$$L(t, \beta) = \exp\left(-\left(\frac{t_n}{\theta}\right)^{\beta}\right)\left(\frac{\beta}{\theta}\right)^{n}\prod_{i=1}^{n}\left(\frac{t_i}{\theta}\right)^{\beta-1}. \tag{3.2.1.5}$$

The MLE for the shape parameter is given by

$$\hat{\beta}_n = \frac{n}{\sum_{i=1}^{n}\log\left(\frac{t_n}{t_i}\right)}, \tag{3.2.1.6}$$

and for the scale parameter is

$$\hat{\theta}_n = \frac{t_n}{n^{1/\hat{\beta}_n}}. \tag{3.2.1.7}$$

Note that the MLE of θ depends on the MLE of β.

3.2.2 Development of the Bayesian Estimates

The authors [65] and [4] justified the applicability of Bayesian analysis to the PLP based on the Crow, [26, 25], failure data from a system undergoing developmental testing (Table 3.1), by showing that the MLE of the key parameter β varies depending on the last failure time (largest time).

Moreover, the authors used the Crow data (40 successive failure times) to compute the MLE of β ($\hat{\beta}_{40} = 0.49$), then computed the estimate considering the $t_{39} = 3181$ is the largest failure time ($\hat{\beta}_{39} = 0.48$) and so on. After computing all MLEs of the key parameter β, they found that the MLEs of β follows a four-parameter Burr probability distribution, $g(\beta; \alpha, \gamma, \delta, \kappa)$, known as the four-parameter Burr type XII probability distribution, with a PDF given by:

Table 3.1: Crow's failure times of a system under development

Failure times						
0.7	3.7	13.2	17.6	54.5	99.2	112.2
120.9	151	163	174.5	191.6	282.8	355.2
486.3	490.5	513.3	558.4	678.1	688	785.9
887	1010.7	1029.1	1034.4	1136.1	1178.9	1259.7
1297.9	1419.7	1571.7	1629.8	1702.4	1928.9	2072.3
2525.2	2928.5	3016.4	3181	3256.3	–	–

$$g_B(\beta) = g(\beta;\alpha,\gamma,\delta,\kappa) = \begin{cases} \frac{\alpha\kappa\left(\frac{\beta-\gamma}{\delta}\right)^{\alpha-1}}{\delta\left(1+\left(\frac{\beta-\gamma}{\delta}\right)^{\alpha}\right)^{\kappa+1}} & \gamma \le \beta < \infty \\ 0 & otherwise \end{cases}, \tag{3.2.2.1}$$

where the hyperparameters α, γ, δ and κ are being estimated using MLEs in the Goodness of Fit (GOF) test applied to the β estimates. The MLE of the key parameter β is always affected by the largest failure, and therefore it is recommended not to consider it unknown constant. This recommendation provides the opportunity to study Bayesian analysis in the PLP with respect to various selections of loss functions and priors.

The Bayesian estimates of β will be derived using the squared-error and Higgins-Tsokos loss functions.

3.2.2.1 Bayesian Estimates Using Squared Error (S-E) Loss Function

The S-E loss function is given by:

$$L(\hat{\xi},\xi) = (\hat{\xi}-\xi)^2. \tag{3.2.2.1}$$

The risk using the S-E loss function, where $\xi = \beta$ represents the estimate of $\hat{\xi} = \hat{\beta}$, is given by:

$$E[L(\hat{\beta},\beta)] = \int_{-\infty}^{\infty} [(\hat{\beta}-\beta)^2] h(\beta|t) d\beta. \tag{3.2.2.2}$$

By differentiating $E[L(\hat{\beta},\beta)]$ with respect to β and setting it equal to zero we solve for $\hat{\beta}$, the Bayesian estimate of β with respect to the S-E loss function and Burr probability distribution, Eq. (3.2.2.1), given by:

$$\hat{\beta}_{B.SE} = \int_{-\infty}^{\infty} \beta \cdot h(\beta|t) d\beta, \tag{3.2.2.3}$$

where the posterior PDF of β given data (t), $h(\beta|t)$, using the Bayes' theorem, is given by:

$$h(\beta|t) = \frac{L(t|\beta) g_B(\beta)}{\int_{-\infty}^{\infty} L(t|\beta) g_B(\beta) d\beta}. \tag{3.2.2.4}$$

Then, the Bayesian estimate of β, under the squared-error loss, is given by

$$\hat{\beta}_{B.SE} = \frac{\int_{\gamma}^{\infty} \frac{\beta^{n+1}}{\theta^n} \exp\left\{-\left(\frac{t_n}{\theta}\right)^{\beta}\right\} \prod_{i=1}^{n} \left(\frac{t_i}{\theta}\right)^{\beta-1} \frac{(\frac{\beta-\gamma}{\delta})^{\alpha-1}}{(1+(\frac{\beta-\gamma}{\delta})^{\alpha})^{\kappa+1}} d\beta}{\int_{\gamma}^{\infty} \frac{\beta^{n}}{\theta^n} \exp\left\{-\left(\frac{t_n}{\theta}\right)^{\beta}\right\} \prod_{i=1}^{n} \left(\frac{t_i}{\theta}\right)^{\beta-1} \frac{(\frac{\beta-\gamma}{\delta})^{\alpha-1}}{(1+(\frac{\beta-\gamma}{\delta})^{\alpha})^{\kappa+1}} d\beta}. \tag{3.2.2.5}$$

3.2.2.2 Bayesian Estimates Using the Higgins-Tsokos Loss Function

The H-T loss function (1976) is given by

$$L(\hat{\xi}, \xi) = \frac{f_1 \exp\left\{f_2(\hat{\xi} - \xi)\right\} + f_2 \exp\left\{-f_1(\hat{\xi} - \xi)\right\}}{f_1 + f_2} - 1, \; f_1, f_2 > 0. \tag{3.2.2.1}$$

Higgins and Tsokos [47] showed that it places more weight on the extreme underestimation and overestimation when $f_1 > f_2$ and $f_1 < f_2$, respectively. The risk using the H-T loss function, where $\xi = \beta$ represents the estimate of $\hat{\xi} = \hat{\beta}$, is given by:

$$E[L(\hat{\beta}, \beta)] = \int_{-\infty}^{\infty} [\frac{f_1 \exp\left\{f_2(\hat{\beta} - \beta)\right\} + f_2 \exp\left\{-f_1(\hat{\beta} - \beta)\right\}}{f_1 + f_2} - 1] h(\beta|t) d\beta. \tag{3.2.2.2}$$

By differentiating $E[L(\hat{\beta}, \beta)]$ with respect to β and setting it equal to zero we solve for $\hat{\beta}$, the Bayesian estimate of β with respect to the H-T loss function, given by:

$$\hat{\beta}_{B.TH} = \frac{1}{f_1 + f_2} \ln[\frac{\int_{-\infty}^{\infty} \exp\{f_1\beta\} h(\beta|t) d\beta}{\int_{-\infty}^{\infty} \exp\{-f_2\beta\} h(\beta|t) d\beta}]. \tag{3.2.2.3}$$

The Bayesian estimate of β with respect to the Higgins-Tsokos loss function and Burr probability distribution, as the prior, has $h(\beta|t)$ given by

$$h(\beta|t) = \frac{(\frac{\beta}{\theta})^n \exp\left\{-(\frac{t_n}{\theta})^\beta\right\} \prod_{i=1}^n \left(\frac{t_i}{\theta}\right)^{\beta-1} \frac{(\frac{\beta-\gamma}{\delta})^{\alpha-1}}{(1+(\frac{\beta-\gamma}{\delta})^\alpha)^{\kappa+1}} d\beta}{\int_\gamma^\infty (\frac{\beta}{\theta})^n \exp\left\{-(\frac{t_n}{\theta})^\beta\right\} \prod_{i=1}^n \left(\frac{t_i}{\theta}\right)^{\beta-1} \frac{(\frac{\beta-\gamma}{\delta})^{\alpha-1}}{(1+(\frac{\beta-\gamma}{\delta})^\alpha)^{\kappa+1}} d\beta}. \tag{3.2.2.4}$$

With the use of Eq. (3.2.1.3), the conditional reliability of t_i, the analytical structure of the conditional Bayesian reliability estimate for the PLP that is subject to the above information is given by:

$$\hat{R}_B(t_i|t_1, t_2, ..., t_{i-1}) = \exp\left\{-\int_{t_{i-1}}^{t_i} \hat{V}'_B(t;\beta,\theta)dt\right\},\ t_i > t_{i-1} > 0, \tag{3.2.2.5}$$

where

$$\hat{V}'_B(t;\hat{\beta}_{B^*},\theta) = \frac{\hat{\beta}_{B^*}}{\theta}\left(\frac{t}{\theta}\right)^{\hat{\beta}_{B^*}-1},\ \theta > 0, t > 0, \tag{3.2.2.6}$$

where $\hat{\beta}_{B^*}$ is the Bayesian estimate using $\hat{\beta}_{B.SE}$ or $\hat{\beta}_{B.TH}$ for the squared error or Higgins-Tsokos loss functions, respectively. We are also interested in comparing the Bayesian estimates, using Higgins-Tsokos loss function, of the subject parameter for different parametric and non-parametric priors, and with respect to its MLE given by Eq. (3.2.1.6), assuming β has a random behavior and θ as known; as well as, comparing Eq. (3.2.1.7) with an adjusted MLE considered as a function of β.

3.2.3 Sensitivity Analysis: Prior and loss function

In this section, we seek the answer to the following question: Is the Bayesian MLE estimate of the intensity function, $V(t;\beta,\theta)$, of the PLP sensitive to the selections of the prior(parametric and non-parametric) and loss function (Higgins-Tsokos and S-E loss functions)? Assuming β is a random variable, using simulated data, sensitive analysis was done for the following parametric and non-parametric priors ([4]):

(i) Jeffreys' prior ([51]): Jeffreys' prior is proportional to the square root of the determinant of the Fisher information matrix ($I(\beta)$). It is a non-informative prior, where the Jeffreys' prior for the key parameter of the PLP $I(\beta)$ is scalar in this case, is given by:

$$g^J(\beta) \propto \sqrt{I(\beta)} = \sqrt{-E(\frac{\partial^2 LogL(t;\beta)}{\partial\beta^2})} \propto \frac{1}{\beta},\ \beta > 0. \tag{3.2.3.1}$$

(ii) The inverted gamma: The PLP and inverted gamma probability distributions belong to the exponential family of probability distributions, which makes the latter a logical choice for an informative parametric prior for β. The inverted gamma probability distribution is given by:

$$g^{IG}(\beta) \propto \left(\frac{\mu}{\beta}\right)^{v+1} \frac{1}{\mu\Gamma(v)} \exp\left\{\frac{-\mu}{\beta}\right\},\ \beta > 0, \mu > 0, v > 0, \tag{3.2.3.2}$$

where v and μ are the shape and scale parameters.

(iii) Kernel' prior:

The kernel probability density estimation is a non-parametric method to approximately estimate the PDF of β using a finite data set. It is given by:

$$g^{K}(\beta) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{\beta - \beta_i}{h}\right), \tag{3.2.3.3}$$

where K is the kernel function and h is a positive number called the bandwidth.

3.2.3.1 The Jeffreys' Prior

Assuming Jeffreys' PDF (3.2.3.1) as the prior of β and using the likelihood (3.2.1.5) and (3.2.2.4), the posterior density of β is:

$$h_J(\bar{t}|\beta) = \frac{\exp\left\{\left(\frac{t_n}{\theta}\right)^{\beta}\right\} \frac{\beta^{n-1}}{\theta^{n\beta}} \prod_{i=1}^{n} (t_i)^{\beta-1}}{\int_0^{\infty} \exp\left\{\left(\frac{t_n}{\theta}\right)^{\beta}\right\} \frac{\beta^{n-1}}{\theta^{n\beta}} \prod_{i=1}^{n} (t_i)^{\beta-1} d\beta}. \tag{3.2.3.1}$$

Thus, the Jeffreys' Bayesian estimate of the key parameter β under the S-E and H-T loss functions, using (3.2.2.3) and (3.2.2.3), are given by:

$$\hat{\beta}^{J}_{B.SE} = \int_0^{\infty} \beta \cdot h_J(\bar{t}|\beta) d\beta, \tag{3.2.3.2}$$

and

$$\hat{\beta}^{J}_{B.HT} = \frac{1}{f_1 + f_2} \ln[\frac{\int_0^{\infty} \exp\{f_1\beta\}\, h_J(\bar{t}|\beta) d\beta}{\int_0^{\infty} \exp\{-f_2\beta\}\, h_J(\bar{t}|\beta) d\beta}]. \tag{3.2.3.3}$$

We must rely on a numerical estimation because we cannot obtain close solutions for both $\hat{\beta}^{J}_{B.SE}$ and $\hat{\beta}^{J}_{B.HT}$. Also note that it depends on knowing or being able to estimate the scale parameter θ.

3.2.3.2 The inverted gamma prior

The following is an examination of the problem when the prior density of β is given by the inverted gamma (3.2.3.2). Using the likelihood (3.2.1.5), the posterior density of β is given by:

$$h_{IG}(t|\beta) = \frac{\frac{\beta^{n-v-1}}{\theta^{n\beta}} \exp\left\{-\left(\frac{t_n}{\theta}\right)^{\beta} - \frac{\mu}{\beta}\right\} \prod_{i=1}^{n}(t_i)^{\beta-1}}{\int_0^{\infty} \frac{\beta^{n-v-1}}{\theta^{n\beta}} \exp\left\{-\left(\frac{t_n}{\theta}\right)^{\beta} - \frac{\mu}{\beta}\right\} \prod_{i=1}^{n}(t_i)^{\beta-1} d\beta}. \tag{3.2.3.1}$$

Thus, the Bayesian estimates of β under the inverted gamma with respect to the S-E and H-T loss functions, using (3.2.2.3) and (3.2.2.3), are given by:

$$\hat{\beta}_{B.SE}^{IG} = \int_0^{\infty} \beta \cdot h_{IG}(\bar{t}|\beta) d\beta, \tag{3.2.3.2}$$

and

$$\hat{\beta}_{B.HT}^{IG} = \frac{1}{f_1 + f_2} \ln[\frac{\int_0^{\infty} \exp\{f_1\beta\} h_{IG}(t|\beta) d\beta}{\int_0^{\infty} \exp\{-f_2\beta\} h_{IG}(t|\beta) d\beta}]. \tag{3.2.3.3}$$

Here as well, we must rely on a numerical estimation because we cannot obtain close solutions for $\hat{\beta}_{B.SE}^{IG}$ and $\hat{\beta}_{B.HT}^{IG}$. Also note that it depends on knowing or being able to estimate the scale parameter θ.

3.2.3.3 The kernel prior

Assuming Kernel density (3.2.3.3) as the prior of β and using the likelihood (3.2.1.5), the posterior density of β is:

$$h_k(\bar{t}|\beta) = \frac{\exp\left\{\left(\frac{t_n}{\theta}\right)^{\beta}\right\} \frac{\beta^n}{\theta^{n\beta}} \prod_{i=1}^{n}(t_i)^{\beta-1} \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{\beta-\beta_i}{h}\right)}{\int_0^{\infty} \exp\left\{\left(\frac{t_n}{\theta}\right)^{\beta}\right\} \frac{\beta^n}{\theta^{n\beta}} \prod_{i=1}^{n}(t_i)^{\beta-1} \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{\beta-\beta_i}{h}\right) d\beta}. \tag{3.2.3.1}$$

Thus, the kernel' Bayesian estimates of the key parameter β under the S-E and H-T loss functions, (3.2.2.3) and (3.2.2.3), are given by:

$$\hat{\beta}_{B.SE}^{K} = \int_0^{\infty} \beta \cdot h_K(\bar{t}|\beta) d\beta, \tag{3.2.3.2}$$

and

$$\hat{\beta}_{B.HT}^{K} = \frac{1}{f_1 + f_2} \ln[\frac{\int_{\gamma}^{\infty} \exp\{f_1\beta\} h_k(\bar{t}|\beta) d\beta}{\int_{\gamma}^{\infty} \exp\{-f_2\beta\} h_k(\bar{t}|\beta) d\beta}]. \tag{3.2.3.3}$$

We must rely on a numerical estimation because we cannot obtain close solutions for $\hat{\beta}^{K}_{B.SE}$ and $\hat{\beta}^{K}_{B.HT}$. Also note that it depends on knowing or being able to estimate the scale parameter θ. In addition, the kernel function, $K(u)$, and bandwidth, h, will be chosen to minimize the asymptotic mean integrated squared error (AMISE) given by:

$$AMISE\left(\hat{f}(\beta)\right) = \int E\left[\left(\hat{f}(\beta) - f(\beta)\right)^2\right] d\beta, \tag{3.2.3.4}$$

where $\hat{f}(\beta)$ and $f(\beta)$ are the estimated probability density of β and the true probability density of β respectively.

Table 3.2 shows the acronyms and notations used in this study.

Table 3.2: Acronyms and notations used in this study

Acronyms	
HPP	Homogeneous Poisson Process
NHPP	Non-Homogeneous Poisson Process
PLP	Power Law Process
MLE	Maximum likelihood estimate
PDF	Probability density function
CDF	Cumulative density function
Notations	
$\hat{\beta}$ and $\hat{\theta}$	MLEs of the PLP shape and scale parameters
T_1, T_2, .., T_n	First n ordered failure times of the PLP
$V(t;\beta,\theta)$	Intensity function of the PLP
RE	Relative efficiency
$AMISE$	Asymptotic mean integrated squared error
H-T	Higgin-Tsokos
S-E	Squared-Error
$\hat{\beta}_{HT}$	Bayesian estimate of β under H-T loss function
$\hat{\beta}_{SE}$	Bayesian estimate of β under S-E loss function
$\hat{V}_{HT}$	Bayesian MLE estimate of $V(t;\beta,\theta)$ under H-T loss function
$\hat{V}_{SE}$	Bayesian MLE estimate of $V(t;\beta,\theta)$ under S-E loss function
$B.HT$	Bayesian estimate under Burr PDF and H-T loss function
$B.SE$	Bayesian estimate under Burr PDF and S-E loss function

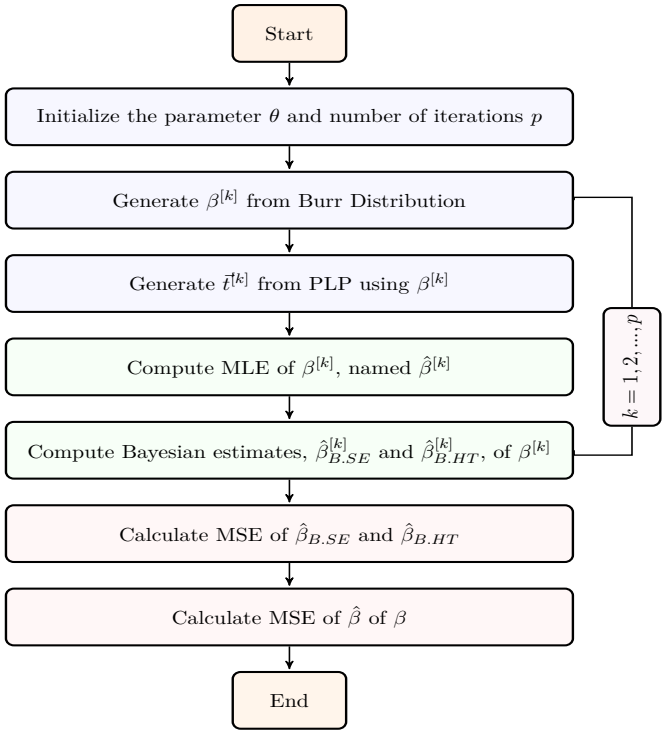
3.3 Results and Discussion

3.3.1 Numerical simulation

A Monte Carlo simulation was used to compare the Bayesian, under the S-E and H-T loss functions, and the MLE approaches. The parameter β of the intensity function for the PLP was calculated

using numerical integration techniques in conjunction with a Monte Carlo simulation to obtain its Bayesian estimates. Substituting these estimates in the intensity function we obtained the Bayesian intensity function estimates, from which the reliability function can be estimated.

For a given value of the parameter θ, a stochastic value for the parameter β was generated from a prior probability density. For a pair of values of θ and β, 400 samples of 40 failure times that follow a PLP were generated. This procedure was repeated 250 times and for three distinct values of θ. The procedure is based on the schematic diagram given by the algorithm that is given by Figure 3.1.



Simulation to analyze Bayesian estimates of β for a given θ

For each sample of size 40, the Bayesian estimates and MLEs of the parameter were calculated when $\theta \in \{0.5, 1.7441, 4\}$. The comparison is based on the mean squared error (MSE) averaged over the $100,000$ repetitions. The results are given in Table 3.3.

It is observed that $\hat{\beta}_{B.SE}$ and $\hat{\beta}_{B.HT}$ maintain similar accuracy, where both are superior to $\hat{\beta}$ in estimating β.

For different sample sizes, the Bayesian estimates under S-E and H-T loss functions and the MLEs of the parameter β were calculated and averaged over $10,000$ repetitions. Table 3.4 displays the simulated result of comparing a true value of β with respect to its MLE and Bayesian estimates

Table 3.3: MSE for Bayesian estimates, under squared error and Higgin-Tsokos loss functions, and MLEs of β

θ	MSE of $\hat{\beta}$	MSE of $\hat{\beta}_{B.SE}$	MSE of $\hat{\beta}_{B.HT}$
0.5	0.0112436	0.000507761	0.000507356
1.7441	0.0110573	0.000516356	0.000516057
4	0.010961	0.000519055	0.000518632

for $n = 20, 30, \ldots, 160$.

Table 3.4: Bayesian estimates, under squared error and Higgin-Tsokos loss functions, and MLEs for the parameter β= 0.7054 averaged over 10000 repetitions

n	β_{Fixed}	$\hat{\beta}$	$\hat{\beta}_{B.SE}$	$\hat{\beta}_{B.HT}$
20	0.7054	0.784026	0.673706	0.675263
30	0.7054	0.756617	0.689413	0.690189
40	0.7054	0.743982	0.695989	0.696467
50	0.7054	0.73531	0.698826	0.699158
60	0.7054	0.729563	0.700393	0.700642
70	0.7054	0.725977	0.701493	0.70169
80	0.7054	0.723338	0.70222	0.702382
100	0.7054	0.719117	0.703049	0.703165
120	0.7054	0.716315	0.703496	0.703585
140	0.7054	0.714821	0.703909	0.70398
160	0.7054	0.713641	0.704185	0.704244

It can be observed that the Bayesian estimates of β are closer to the true value than the MLE of β, where the Bayesian estimate under the H-T loss function is slightly performing better even for a very small sample size of $n = 20$. A graphical comparison of the true estimate of β along with the Bayesian estimates (under both S-E and H-T loss functions) and MLE as a function of sample size is given in Figure 3.2.

Figure 3.2 shows the the excellent performance of he Bayesian estimates compared to the MLE of the key parameter β. The Bayesian estimates tend to underestimate while while the MLE estimate tends to overestimate the true value, especially for small sample sizes. The MSEs of the MLE and Bayesian estimates of β for each sample size are given below by Figure 3.3.

For the considered sample sizes, the MSEs of the Bayesian estimates of β are sufficiently smaller than the MSEs for the MLE of β. The Bayesian estimate under the H-T loss function performed slightly better than the Bayesian estimate under the S-E loss function.

Since the Bayesian estimates under both loss functions for β are superior to its MLE, Molinares and Tsokos [65] showed the improvement in the scale paramter (θ) when its estimate (3.2.1.7) is adjusted by using the Bayesian estimate of β instead of the corresponding MLE. Therefore, we

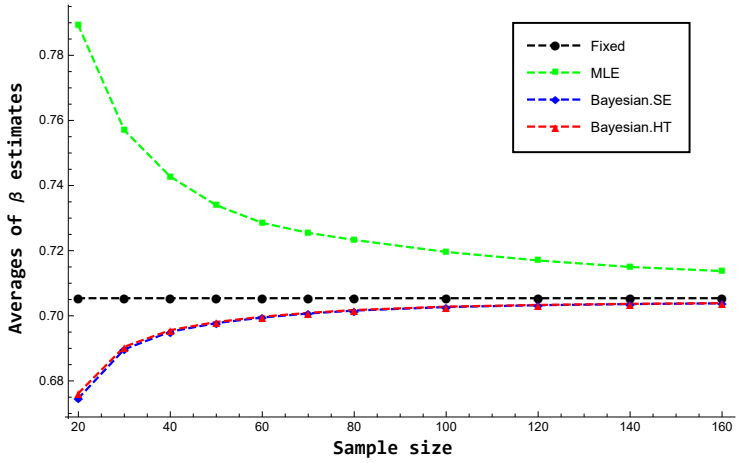


Figure 3.2.: β estimates versus sample size.

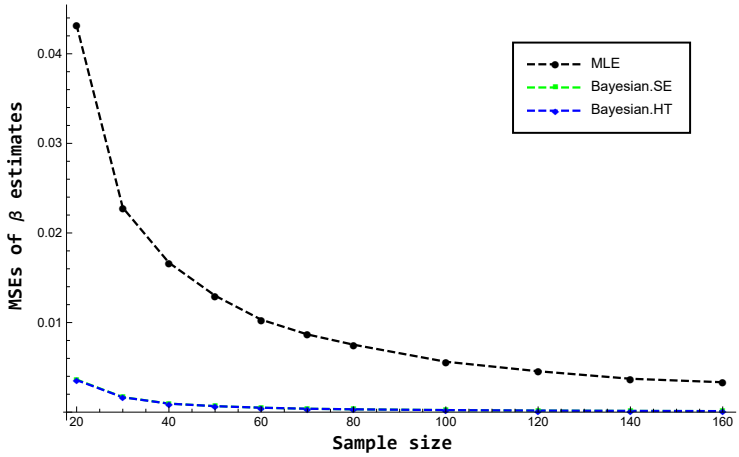


Figure 3.3.: MSE of β Bayesian estimates versus sample size.

calculated the adjusted estimate of θ using MLE and Bayesian estimates under S-E and H-T loss functions of β, shown in Table 3.5.

This proposed adjusted estimates, $\hat{\theta}_{B.SE}$ and $\hat{\theta}_{B.HT}$, were averaged over the $10,000$ repetitions. It can be appreciated that, based on the Bayesian influence on β, $\hat{\theta}_{B.SE}$ and $\hat{\theta}_{B.HT}$, as expected, are better estimates than the MLE of θ $(\hat{\theta})$. This also can be seen in Figure 3.4, which visualize the performance of $\hat{\theta}_{B.SE}$ and $\hat{\theta}_{B.HT}$ compared to the corresponding MLE.

Figure 3.4 shows the excellent performance of the adjusted estimates of θ, where the adjusted estimate under the H-Twas slightly closer to the true value. The MSEs of these estimates of θ are displayed in Figure 3.5 given below.

The MSEs of the adjusted estimates of the shape parameter (θ) are significantly smaller than

Table 3.5: MLE Bayesian estimates, under squared error and Higgin-Tsokos loss functions, and and MLEs for the parameter θ= 1.7441 averaged over 10,000 repetitions

n	θ	$\hat{\theta}_{MLE}$	$\hat{\theta}_{B.SE}$	$\hat{\theta}_{B.HT}$
20	1.7441	3.17139	1.3507	1.36422
30	1.7441	2.908	1.50142	1.5097
40	1.7441	2.73107	1.57545	1.58115
50	1.7441	2.59245	1.61556	1.61985
60	1.7441	2.48865	1.64065	1.64406
70	1.7441	2.41782	1.65803	1.66084
80	1.7441	2.36522	1.67055	1.67294
100	1.7441	2.26774	1.68719	1.68902
120	1.7441	2.20117	1.69776	1.69923
140	1.7441	2.15539	1.70537	1.70659
160	1.7441	2.11872	1.71089	1.71193

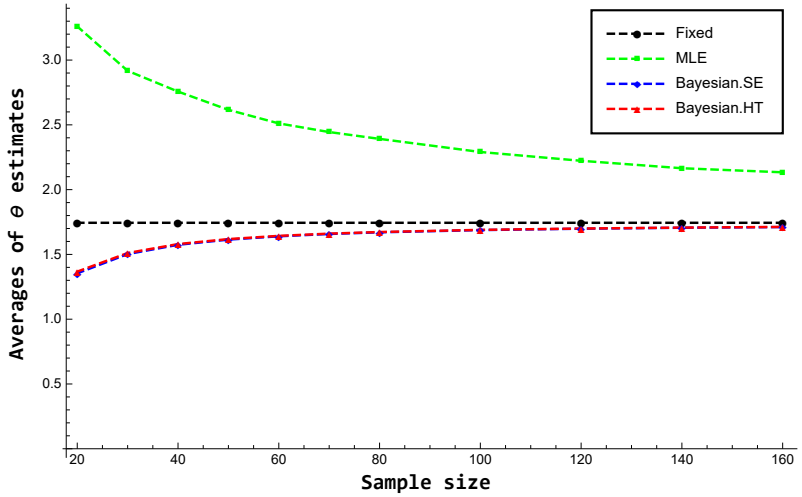


Figure 3.4.: θ estimates versus sample size.

the MSEs of the MLE estimate. The MSEs of the adjusted estimates are then displayed alone in Figure 3.6 to look closer at their performance.

It can be noticed that the adjusted estimate of θ under the influence of the Bayesian estimate with the H-T loss function, is slightly better, particularly when considering small sample sizes.

We computed the adjusted estimate for the parameter θ and its MSE over $10,000$ repetitions for different values of θ and sample size $n = 40$. The results are given in Table 3.6.

The adjusted estimate of θ are more accurate when considering small true values of θ than for larger values.

The slight improvements in the estimation of the shape and scale parameters of the PLP is expected to jointly improve the estimate of the intensity function and therefore the reliability

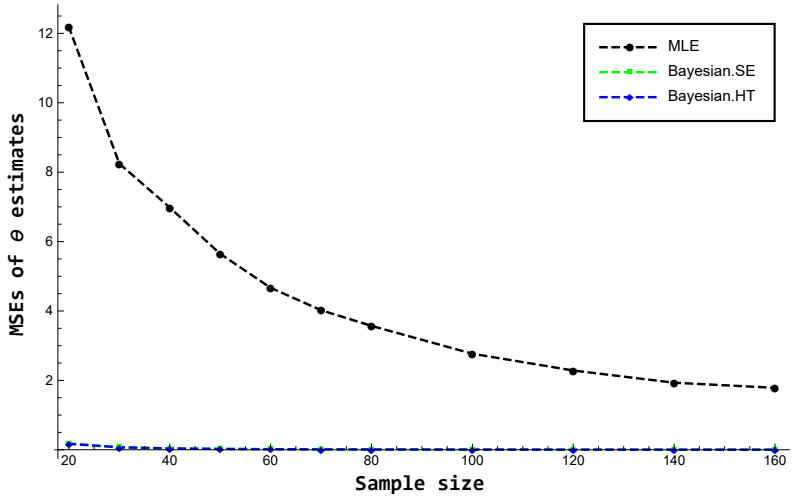


Figure 3.5.: MSE of θ Bayesian and MLE estimates versus sample size.

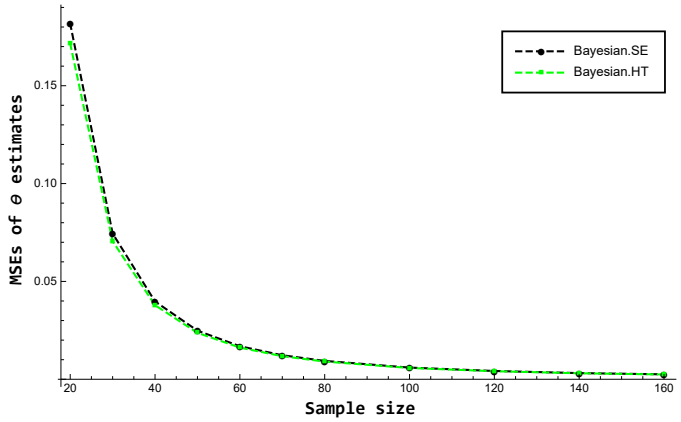


Figure 3.6.: MSE of θ Bayesian estimates versus sample size.

Table 3.6: MSE of θ estimates using Bayesian estimates, under squared error and Higgin-Tsokos loss functions, and MLE of β

θ	$\hat{\theta}_{B.SE}$	$\hat{\theta}_{B.HT}$	MSE of $\hat{\theta}_{B.SE}$	MSE of $\hat{\theta}_{B.HT}$
0.5	0.502022	0.503314	0.00692919	0.00691164
1.7441	1.74639	1.7509	0.0830342	0.0827802
4	3.99992	4.01025	0.440474	0.439035

estimation of a software. For a fixed value of $\theta = 1.7441$ and a sample size similar to the size of the collected data, $n = 40$, the estimates of the intensity function $\hat{V}_{MLE}(t)$, $\hat{V}_{B.SE}(t)$, and $\hat{V}_{B.HT}(t)$ were obtained when we use $\hat{\beta}$, $\hat{\beta}_{B.SE}$, and $\hat{\beta}_{B.HT}$, respectively, in (3.1.0.2). That is,

$$\hat{V}'_{MLE}(t) = \frac{\hat{\beta}}{\theta}\left(\frac{t}{\theta}\right)^{\hat{\beta}-1}, \ \theta > 0, t > 0, \tag{3.3.1.1}$$

$$\hat{V}'_{B.SE}(t) = \frac{\hat{\beta}_{B.SE}}{\theta}\left(\frac{t}{\theta}\right)^{\hat{\beta}_{B.SE}-1},\ \theta > 0, t > 0, \tag{3.3.1.2}$$

and

$$\hat{V}'_{B.HT}(t) = \frac{\hat{\beta}_{B.HT}}{\theta}\left(\frac{t}{\theta}\right)^{\hat{\beta}_{B.HT}-1},\ \theta > 0, t > 0. \tag{3.3.1.3}$$

Their graphs (Figure 3.7) reveal, as expected, the superior performance of $\hat{V}'_{B.SE}(t)$ and $\hat{V}'_{B.HT}(t)$.

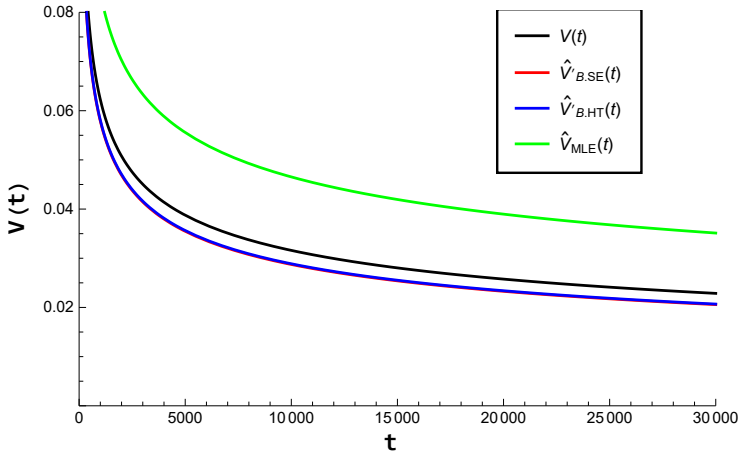


Figure 3.7.: Graph for $\theta = 1.7441$ and the corresponding β Bayesian estimates and MLE's used in $\hat{V}'_{MLE}$, $\hat{V}'_{B.SE}$, and $\hat{V}'_{B.HT}$ (of time t) , n = 40.

In order to obtain Bayesian estimates of the intensity function, $\hat{V}^*_{B.SE}$ and $\hat{V}^*_{B.HT}$, we substituted the Bayesian estimates of β and its corresponding θ MLE in (3.1.0.2):

$$\hat{V}^*_{B.SE}(t) = \frac{\hat{\beta}_{B.SE}}{\hat{\theta}}\left(\frac{t}{\hat{\theta}}\right)^{\hat{\beta}_{B.SE}-1},\ t > 0, \tag{3.3.1.4}$$

and

$$\hat{V}^*_{B.HT}(t) = \frac{\hat{\beta}_{B.HT}}{\hat{\theta}}\left(\frac{t}{\hat{\theta}}\right)^{\hat{\beta}_{B.HT}-1},\ t > 0. \tag{3.3.1.5}$$

The MLE of the intensity function, $\hat{V}_{MLE}$, is obtained using the MLEs of β and θ. That is,

$$\hat{V}_{MLE}(t) = \frac{\hat{\beta}}{\hat{\theta}}\left(\frac{t}{\hat{\theta}}\right)^{\hat{\beta}-1},\ t > 0. \tag{3.3.1.6}$$

The Bayesian MLE of the intensity function under the influence of the Bayesian estimates of

β, denoted by $\hat{V}_{B.SE}$ and $\hat{V}_{B.HT}$, are obtained by substituting $\hat{\beta}_{B.HT}$ and $\hat{\beta}_{B.SE}$ with $\hat{\theta}_{B.HT}$ and $\hat{\theta}_{B.SE}$, respectively, in (3.1.0.2):

$$\hat{V}_{B.SE}(t) = \frac{\hat{\beta}_{B.SE}}{\hat{\theta}_{B.SE}} \left(\frac{t}{\hat{\theta}_{B.SE}} \right)^{\hat{\beta}_{B.SE}-1}, \; t > 0, \tag{3.3.1.7}$$

and

$$\hat{V}_{B.HT}(t) = \frac{\hat{\beta}_{B.HT}}{\hat{\theta}_{B.HT}} \left(\frac{t}{\hat{\theta}_{B.HT}} \right)^{\hat{\beta}_{B.HT}-1}, \; t > 0. \tag{3.3.1.8}$$

To measure the robustness of $\hat{V}_{B.HT}$ with respect to $\hat{V}_{B.SE}$ and $\hat{V}_{MLE}$, we calculated the relative efficiency (RE) of the estimate $\hat{V}_{B.HT}$ compared to the estimate $\hat{V}_{B.SE}$ defined by:

$$RE(\hat{V}_{B.HT}, \hat{V}_{B.SE}) = \frac{\int_{-\infty}^{\infty} [\hat{V}_{B.HT}(t) - V(t)]^2 dt}{\int_{-\infty}^{\infty} [\hat{V}_{B.SE}(t) - V(t)]^2 dt}. \tag{3.3.1.9}$$

If $RE = 1$, $\hat{V}_{B.HT}$ and $\hat{V}_{B.SE}$ will be interpreted as equally efficient. If $RE < 1$, $\hat{V}_{B.HT}$ is more efficient than $\hat{V}_{B.SE}$. To the contrary, if $RE > 1$, $\hat{V}_{B.HT}$ is less efficient than $\hat{V}_{B.SE}$. Similarly, we compared $\hat{V}_{B.HT}$ and $\hat{V}_{MLE}$. Bayesian estimates and MLEs for the parameter $\beta = 0.7054$ and $\theta = 1.7441$ (Table 3.7), averaged over $10,000$ repetitions, are used, for $n = 40$, to compare $\hat{V}_{B.TH}$, $\hat{V}_{B.SE}$ and $\hat{V}_{MLE}$ using (3.3.1.9). The results are given in Tables 3.8 and 3.9.

Table 3.7: Averages of the Bayesian (under the under squared error and Higgin-Tsokos loss functions) and MLE estimates of β and θ

β	$\hat{\beta}$	$\hat{\beta}_{B.SE}$	$\hat{\beta}_{B.HT}$	θ	$\hat{\theta}$	$\hat{\theta}_{B.SE}$	$\hat{\theta}_{B.HT}$
0.7054	0.743982	0.695989	0.696467	1.7441	2.73107	1.57545	1.58115

Table 3.8: Intensity functions with Bayesian and MLE estimates for β and θ

$V(t)$	$\hat{V}_{MLE}$	$\hat{V}_{B.SE}$	$\hat{V}_{B.HT}$
$0.476465 \cdot t^{-0.2946}$	$0.352321 \cdot t^{-0.256018}$	$0.507238 \cdot t^{-0.304011}$	$0.5062 \cdot t^{-0.303533}$

Table 3.9: Relative efficiency of $\hat{V}_{B.HT}$ to $\hat{V}_{MLE}$ and $\hat{V}_{B.BS}$

$RE(\hat{V}_{B.SE}, \hat{V}_{MLE})$	$RE(\hat{V}_{B.HT}, \hat{V}_{MLE})$	$RE(\hat{V}_{B.HT}, \hat{V}_{B.SE})$
0.087746	0.0761919	0.868324

For the comparison of $\hat{V}_{B.HT}$ and $\hat{V}_{B.SE}$, the $RE(\hat{V}_{B.HT}, \hat{V}_{B.SE})$ is less than 1, which implies that the intensity function using $\hat{\beta}_{B.HT}$ and $\hat{\theta}_{B.HT}$ is more efficient than the intensity function under $\hat{\beta}_{B.SE}$ and $\hat{\theta}_{B.SE}$. Comparing $\hat{V}_{B.HT}$ and $\hat{V}_{B.SE}$ to $\hat{V}_{MLE}$, we obtained a similar result,

establishing, as expected, the superior relative efficiency of Bayesian estimates over MLE estimates. The corresponding graphs for the intensity functions are given in Figure 3.8.

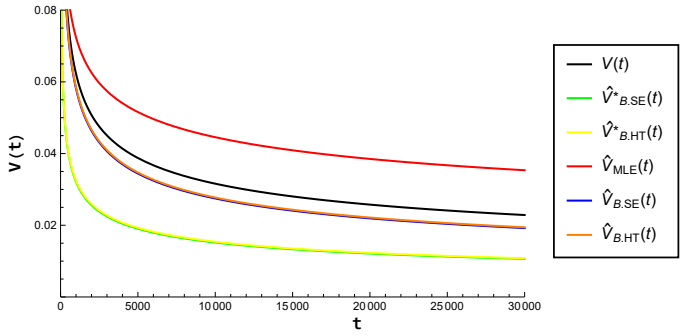


Figure 3.8.: Estimates of the intensity function (of time t) using values in Table 3.7, n = 40

In addition, $\hat{V}^*_{B.HT}$ and $\hat{V}^*_{B.SE}$ are computed using Bayesian estimates for β and MLE estimates θ, which were less efficient compare to $\hat{V}_{MLE}$, $\hat{V}_{B.SE}$, and $\hat{V}_{B.HT}$. Based on the results, the Bayesian estimates under the H-T loss function will be used to analyze the real data.

3.3.2 Using real data

Using the reliability growth data from Table 3.1, we computed $\hat{\beta}_{B.HT}$ and the adjusted estimate $\hat{\theta}_{B.HT}$ in order to obtain a Bayesian intensity function under H-T loss function. We followed the algorithm given by Figure 3.9 to obtain the Bayesian intensity function for the given real data.

For the failure data of Crow, provided in Table 3.1, $\hat{\beta}_{B.HT}$ is approximately 0.501199 and $\hat{\theta}_{B.HT}$ is approximately 2.07144. Therefore, with the use of $\hat{\theta}_{B.HT}$, the Bayesian MLE of the intensity function ($\hat{V}_{B.HT}(t)$) for the real data is given by:

$$\hat{V}_{B.HT}(t) = 0.347933 \cdot t^{-0.498801},\ t > 0. \tag{3.3.2.1}$$

To obtain a Bayesian MLE for the reliability function under H-T loss function, we use this Bayesian estimate for the intensity function. The analytical form for the corresponding Bayesian reliability estimate, based on the data, is given by:

$$\hat{R}_{B.HT}(t_i|t_1,..,t_{i-1}) = \exp\left\{-0.347933 \int_{t_{i-1}}^{t_i} x^{-0.498801} dx\right\},\ t_i > t_{i-1} > 0. \tag{3.3.2.2}$$

Thus, the conditional reliability of the software given that the last two failure times were $t_{39} = 3181$ and $t_{40} = 3256.3$ is approximately 63%.

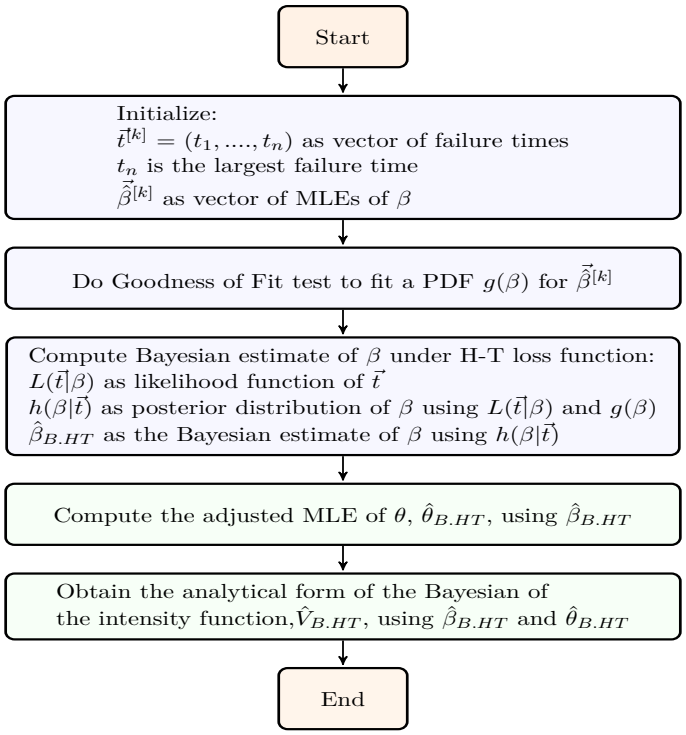


Figure 3.9.: Estimate of the intensity function using Crow real data in Table 3.1

3.3.3 Sensitivity Analysis: Prior and Loss Function

To answer the second research question, "Is the Bayesian estimate of the intensity function, $V(t;\beta,\theta)$, of the PLP sensitive to the selections of the prior (both parametric and non-parametric priors) and loss function?", we developed a simulation procedure, and is given by Figure 3.10.

The algorithm compares the Bayesian and MLE estimates of the intensity function, $V(t;\beta,\theta)$, under different prior PDFs, for various sample sizes, with the H-T and S-E loss functions. The relative efficiency is used to compare these estimates of the $V(t;\beta,\theta)$. The relative efficiency with a value less than 1, larger than 1, and approximately equal to 1 indicate that the Bayesian estimates under the H-T loss function are more, less, equally efficient to the Bayesian estimate under the S-E loss function and the same analysis is applied when we compared to the MLE of $V(t;\beta,\theta)$, respectively. The algorithm starts by initializing the shape and scale parameters of the PLP, β and θ, respectively, and the number of iterations p.

For various sample sizes ($n = 20, 40, 80, 140$), random failure times (time to failures) distributed according to the PLP are simulated using the initialized values of the PLP parameters. Then, the Bayesian and MLE estimates of the key parameter β are computed and used to compute the Bayesian estimates of θ, respectively. After a predetermined number of iterations, the average

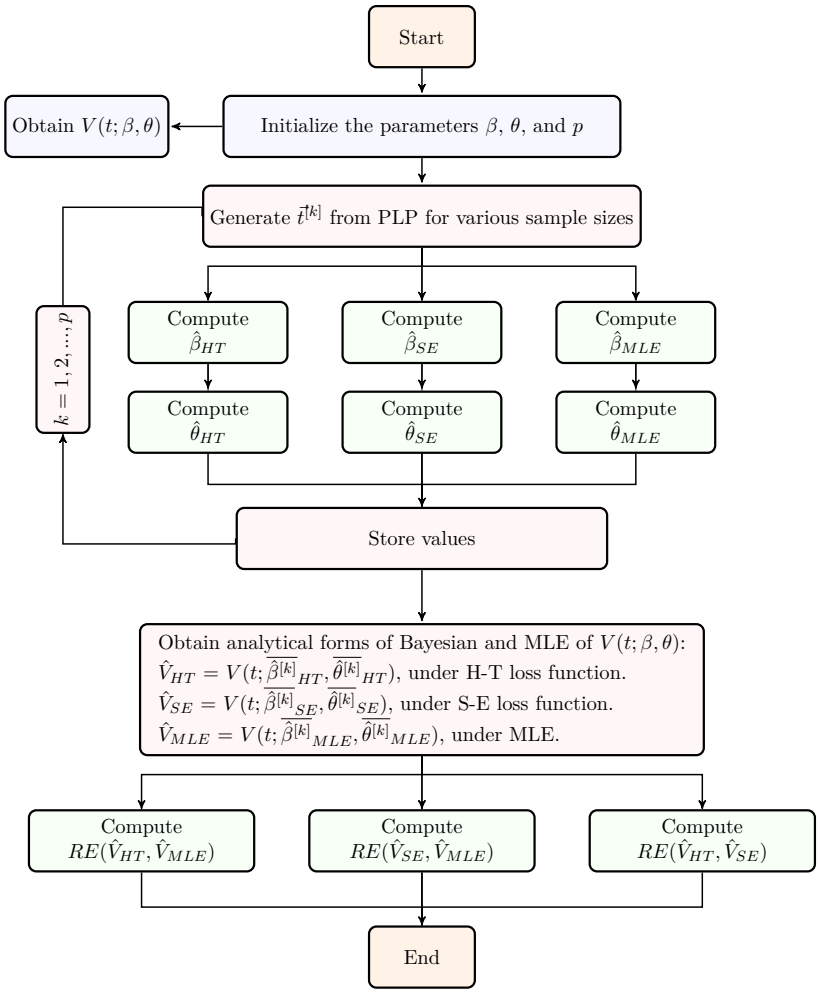


Figure 3.10.: Simulation to compare Bayesian and MLE estimates of the intensity function. Notations found in Table 3.2

values of the Bayesian and MLE estimates of β and θ were used to obtain the analytical forms of the $V(t;\beta,\theta)$ under Bayesian, for both H-T and S-E loss functions and MLE, namely $\hat{V}_{HT}, \hat{V}_{SE}$, and $\hat{V}_{MLE}$, respectively. Informative parametric priors were considered such as the inverted gamma and the Burr PDFs, whereas the Jeffery prior was chosen as non-informative prior. In addition, probability kernel density function is selected as a non-parametric prior PDF. Probability kernel density estimation depends on the sample size, bandwidth, and the choice of the kernel function ($K(u)$). In this study, the optimal bandwidth (h^*) and kernel function were chosen to minimize the asymptotic mean integrated squared error (AMISE). The simplified form of the AMISE is reduced to:

$$AMISE\left(\hat{f}(\beta)\right) = \frac{C(K)}{n \cdot h} + (\frac{1}{4} \cdot h^4 \cdot k_2^2 \cdot R\left(f^{(2)}(\beta)\right)) \tag{3.3.3.1}$$

where:

- C(K)= $\int (K(u))^2 du$,
- n: sample size,
- h: bandwidth,
- $k_2 = \int_{-\infty}^{+\infty} u^2 \cdot K(u) du$,
- $f^{(2)}(\beta)$ is the second derivative of Burr PDF,
- $R(f^{(2)}(\beta))= \int (f^{(2)}(\beta))^2 d\beta$,
- $h^* = \left[\frac{C(K)}{k_2^2 \cdot R\left(f^{(2)}(\beta)\right)}\right]^{1/5} \cdot n^{-1/5}$, (Silverman [90]).

AMISE was numerically calculated using the optimal bandwidth (h^*), with respect to different samples sizes for each kernel function considered in this study, namely Epanechnikov, Cosine, Biweight, Triweight, Gaussian, Triangle, Uniform, Tricube, and Logistic kernel functions. The results are given in Table 3.10.

Table 3.10: Calculations of the AMISE with respect to different sample size, optimal bandwidth, and kernel function

Kernel function	Sample size				
	50	100	150	200	500
Epanechnikov	**0.362827**	**0.208389**	**0.150662**	**0.119688**	**0.0575042**
Cosine	0.362986	0.208481	0.150728	0.119741	0.0575295
Biweight	0.364607	0.209412	0.151401	0.120275	0.0577863
Triweight	0.36674	0.210637	0.152286	0.120979	0.0581243
Gaussian	**0.377644**	**0.2169**	**0.156814**	**0.124576**	**0.0598525**
Triangle	0.366972	0.21077	0.152383	0.121056	0.0581612
Uniform	0.384675	0.220938	0.159734	0.126895	0.0609669
Tricube	0.363433	0.208737	0.150913	0.119888	0.0576002
Logistic	0.399132	0.229241	0.165737	0.131665	0.0632582

The minimum AMISE corresponds to the Epanechnikov kernel function ($K(u)=\frac{3}{4}(1-u^2)I_{|u|\leq 1}$). In addition to the Epanechnikov kernel function, the Gaussian kernel function ($K(u)=\frac{1}{\sqrt{2\pi}} \exp\left(\frac{-u^2}{2}\right) I_{\mathbb{R}}$) was also used in the calculation since it is commonly used for its analytical tractability.

Numerical integration techniques were used to compute the Bayesian estimates of the intensity function, $V(t;\beta,\theta)$, parameters under both H-T and S-E loss functions according to the equations

defined in Section **3.2.3**, for each of the parametric and non-parametric prior PDFs. Samples of size 20, 40, 80, and 140 were generated where the parameters β and θ were initialized to be 0.7054 and 1.7441, respectively. In the analytical form (3.2.2.1), f_1 and f_2 are conditioned to be positive numbers and play a big role in assigning the weight of loss depending on the estimator's behavior, whether underestimating or overestimating. Therefore, the simulation procedure was repeated three times according to the following cases:

1) $f_1 > f_2$

2) $f_1 < f_2$

3) $f_1 = f_2$

The results for $1,000$ repetitions, $f_1 > f_2$, and $n = 20, 40, 80, 140$, are shown in Table 3.11.

Table 3.11: The relative efficiency (RE) of the Bayesian estimate under H-T loss function, $\hat{V}_{HT}$ when $f_1 > f_2$, compared to the Bayesian estimate under S-E loss function, $\hat{V}_{SE}$, and the MLE, $\hat{V}_{MLE}$, of $V(t;\beta,\theta)$

Prior PDF		$RE(\hat{V}_{HT},\hat{V}_{MLE})$	$RE(\hat{V}_{SE},\hat{V}_{MLE})$	$RE(\hat{V}_{HT},\hat{V}_{SE})$
Burr	$n=20$	0.1356	0.1519	0.8923
Inverted gamma		4.2461	4.1632	1.0199
Jeffrey		0.0365	0.0289	1.2616
Gaussian kernel		0.1187	0.1346	0.8818
Epanechnikov kernel		0.1187	0.1346	0.8818
Burr	$n=40$	0.3047	0.3345	0.9107
Inverted gamma		6.3934	6.2832	1.0175
Jeffrey		0.0166	0.0119	1.3947
Gaussian kernel		0.1234	0.1424	0.8663
Epanechnikov kernel		0.1221	0.1411	0.8659
Burr	$n=80$	0.0136	0.0151	0.9007
Inverted gamma		0.8058	0.7934	1.0156
Jeffrey		0.0159	0.0144	1.1065
Gaussian kernel		0.0105	0.0117	0.8988
Epanechnikov kernel		0.0114	0.0127	0.8999
Burr	$n=140$	0.0035	0.0037	0.9367
Inverted gamma		0.1421	0.1399	1.0155
Jeffrey		0.004	0.0037	1.068
Gaussian kernel		0.0019	0.0018	1.0119
Epanechnikov kernel		0.0021	0.0022	0.9670

It can be observed that the Bayesian estimate of the $V(t;\beta,\theta)$ under the H-T loss function ($\hat{V}_{HT}$) and S-E loss function ($\hat{V}_{SE}$), as expected, had an outstanding efficiency compared to the MLE of the $V(t;\beta,\theta)$ ($\hat{V}_{MLE}$) for all sample sizes and prior PDFs, with the exception of the sample sizes

20 and 40 when inverted gamma PDF was the selected prior. The $\hat{V}_{HT}$ was more efficient (6-11% estimation improvement) compared to the $\hat{V}_{SE}$ when Burr PDF is selected to be the prior. The $\hat{V}_{HT}$ had similar efficiency compared to the $\hat{V}_{SE}$ when Jeffrey prior is selected and for large sample sizes, whereas unsurprisingly $\hat{V}_{SE}$ was more efficient for small sample sizes since Jeffrey Bayesian estimate of the key parameter β tends to overestimate and for the H-T loss function gives more exponential weight on the extreme overestimate loss than the extreme underestimate loss when $f_1 > f_2$. For Bayesian Gaussian and Epanechnikov kernel estimates, the $\hat{V}_{HT}$ was more efficient compared to the $\hat{V}_{SE}$ for sample sizes $n = 20, 40$, and 80 with 11-13 % of estimation improvement even though they tend to underestimate and the H-T loss function puts more exponential weight on the extreme underestimation, but tend to have similar efficiency for sample size $n = 140$.

The results for $1,000$ repetitions, $f_1 > f_2$, and $n = 20, 40, 80, 140$, are shown in Table 3.12.

Table 3.12: The relative efficiency (RE) of the Bayesian estimate under H-T loss function, $\hat{V}_{HT}$ when $f_1 < f_2$, compared to the Bayesian estimate under S-E loss function, $\hat{V}_{SE}$, and the MLE, $\hat{V}_{MLE}$, of $V(t;\beta,\theta)$

Prior PDF		$RE(\hat{V}_{HT},\hat{V}_{MLE})$	$RE(\hat{V}_{SE},\hat{V}_{MLE})$	$RE(\hat{V}_{HT},\hat{V}_{SE})$
Burr	$n=20$	0.2068	0.186	1.1116
Inverted gamma		4.7351	4.8309	0.9802
Jeffrey		0.0232	0.0306	0.7589
Gaussian kernel		0.1948	0.1735	1.1226
Epanechnikov kernel		0.1949	0.1736	1.1227
Burr	$n=40$	0.15	0.1327	1.1305
Inverted gamma		5.9173	6.0152	0.9837
Jeffrey		0.0673	0.0785	0.8581
Gaussian kernel		0.0516	0.0431	1.198
Epanechnikov kernel		0.051	0.0425	1.1985
Burr	$n=80$	0.0126	0.0121	1.0406
Inverted gamma		0.8155	0.8274	0.9856
Jeffrey		0.0326	0.0349	0.9365
Gaussian kernel		0.0111	0.0108	1.0307
Epanechnikov kernel		0.0116	0.0112	1.0356
Burr	$n=140$	0.018	0.0183	0.9814
Inverted gamma		0.2545	0.2576	0.988
Jeffrey		0.0329	0.0338	0.9733
Gaussian kernel		0.0222	0.0227	0.9762
Epanechnikov kernel		0.0204	0.0209	0.9772

Again, the Bayesian MLE estimate of the $V(t;\beta,\theta)$ under the H-T loss function ($\hat{V}_{HT}$) and S-E loss function ($\hat{V}_{SE}$) had an outstanding efficiency compared to the MLE of the $V(t;\beta,\theta)$ ($\hat{V}_{MLE}$) for all sample sizes and prior PDFs. When the inverted gamma was selected as prior, the $\hat{V}_{HT}$ was more efficient compared to the $\hat{V}_{SE}$ for all sample sizes with an approximately 2% of estimation

improvement. As expected, the $\hat{V}_{HT}$ was less efficient compared to the $\hat{V}_{SE}$ when Burr PDF, and Gaussian and Epanechnikov kernel densities are selected as priors for sample sizes 20 and 40, since they tend to underestimate the $V(t;\beta,\theta)$ parameters, and the H-T loss function tends to put more weight on the extreme overestimation than on the extreme underestimation when $f_1 > f_2$. But the $\hat{V}_{HT}$ and $\hat{V}_{SE}$ had approximately similar efficiency for sample size $n = 80$, and the $\hat{V}_{HT}$ tends to be slightly more efficient for large sample size ($n = 140$). The $\hat{V}_{HT}$ was more efficient (4-24% estimation improvement) compared to the $\hat{V}_{SE}$ when Burr Jeffrey is chosen to be the prior PDF. The $\hat{V}_{HT}$ had similar efficiency compared to the $\hat{V}_{SE}$ for large sample sizes and when Jeffrey prior is selected, whereas unsurprisingly $\hat{V}_{SE}$ was more efficient for small sample sizes since Jeffrey Bayesian estimate of the key parameter β tends to overestimate and for the H-T loss function gives more exponential weight on the extreme overestimate loss than the extreme underestimate loss when $f_1 > f_2$. For Bayesian Gaussian and Epanechnikov kernel estimates, the $\hat{V}_{HT}$ was more efficient compared to the $\hat{V}_{SE}$ for sample sizes $n = 20, 40$, and 80 with 11-13 % of estimation improvement even though they tend to underestimate and the H-T loss function puts more exponential weight on the extreme underestimation, but tend to have similar efficiency for sample size $n = 140$.

The results for $1,000$ repetitions, $f_1 > f_2$, and $n = 20, 40, 80, 140$, are shown in Table 3.13.

Again, the Bayesian MLE estimate of the $V(t;\beta,\theta)$ under the H-T loss function ($\hat{V}_{HT}$) and S-E loss function ($\hat{V}_{SE}$) had an outstanding efficiency compared to the MLE of the $V(t;\beta,\theta)$ ($\hat{V}_{MLE}$) for all sample sizes and prior PDFs, with the exception of the sample sizes 20 and 40 when inverted gamma PDF was the selected prior. It is observed that both $\hat{V}_{HT}$ and $\hat{V}_{SE}$ had similar efficiency in estimation of the $V(t;\beta,\theta)$ for all sample sizes and priors considered in this study.

The sensitivity analysis shows that the Bayesian estimates of the intensity function of the PLP is sensitive to the prior and loss function selections. Tables 3.11, 3.12, and 3.13 indicate the efficiency of the Bayesian estimates under the H-T loss function when compared to the Bayesian estimate under S-E loss function and to the MLE, given that the engineer should choose the values of f_1 and f_2 based on his/her estimator's behaviour (underestimating and over estimating). Moreover, $f_1 > f_2$ is the recommended choice when the engineer selects Burr or kernel PDFs as their prior knowledge of the behavior of the key parameter β. On the other hand, if the engineer does not have a prior knowledge of the key parameter β, it is still recommended to use H-T loss function in the Bayesian calculations with $f_1 < f_2$.

Table 3.13: The relative efficiency (RE) of the Bayesian estimate under H-T loss function, $\hat{V}_{HT}$ when $f_1 = f_2$, compared to the Bayesian estimate under S-E loss function, $\hat{V}_{SE}$, and the MLE, $\hat{V}_{MLE}$, of $V(t;\beta,\theta)$

Prior PDF		$RE(\hat{V}_{HT},\hat{V}_{MLE})$	$RE(\hat{V}_{SE},\hat{V}_{MLE})$	$RE(\hat{V}_{HT},\hat{V}_{SE})$
Burr	$n=20$	0.0703	0.0702	1.0011
Inverted gamma		3.7132	3.7135	0.9999
Jeffrey		0.0612	0.0613	0.9981
Gaussian kernel		0.0585	0.0583	1.0037
Epanechnikov kernel		0.0585	0.0583	1.0037
Burr	$n=40$	0.1195	0.1194	1.0008
Inverted gamma		7.3018	7.3022	0.9999
Jeffrey		0.1351	0.1352	0.9993
Gaussian kernel		0.0384	0.0384	1.0008
Epanechnikov kernel		0.0381	0.0381	1.0008
Burr	$n=80$	0.0144	0.0144	1.0002
Inverted gamma		0.8626	0.8734	0.9876
Jeffrey		0.025	0.025	0.9998
Gaussian kernel		0.0122	0.0122	1.0003
Epanechnikov kernel		0.0131	0.0131	1.0002
Burr	$n=140$	0.0065	0.0065	1
Inverted gamma		0.1863	0.1863	1
Jeffrey		0.0117	0.0117	0.9999
Gaussian kernel		0.007	0.007	0.9999
Epanechnikov kernel		0.0064	0.0064	0.9999

3.3.4 Interactive User Interface Application

The performed extensive analysis requires efficiency in utilizing the existing programming languages, which therefore requires some programming experience, we developed an interactive user interface application using Wolfram language to compute and visualize the Bayesian and maximum likelihood estimates of the intensity and reliability functions of the Power Law Process for a given data. Figure 3.11 below shows the interface of the application.

The application is very simple to use by any user with a minimal programming experience. First, the use import a given data of failure times by choosing the following option: **Import & Generate Data**, which it will allow him to select the saved file of the given data in his computer. Then, the user can use the Bayesian estimates of the intensity and reliability functions of the Power Law Process under the squared error or the Higgins-Tsokos loss functions by choosing the following options: **Squared error** and **Higgins-Tsokos**, respectively. The approximate maximum likelihood estimates of the intensity and reliability functions of the Power Law Process will directly be reported, along with graph of the intensity function as a function of time. In addition, the

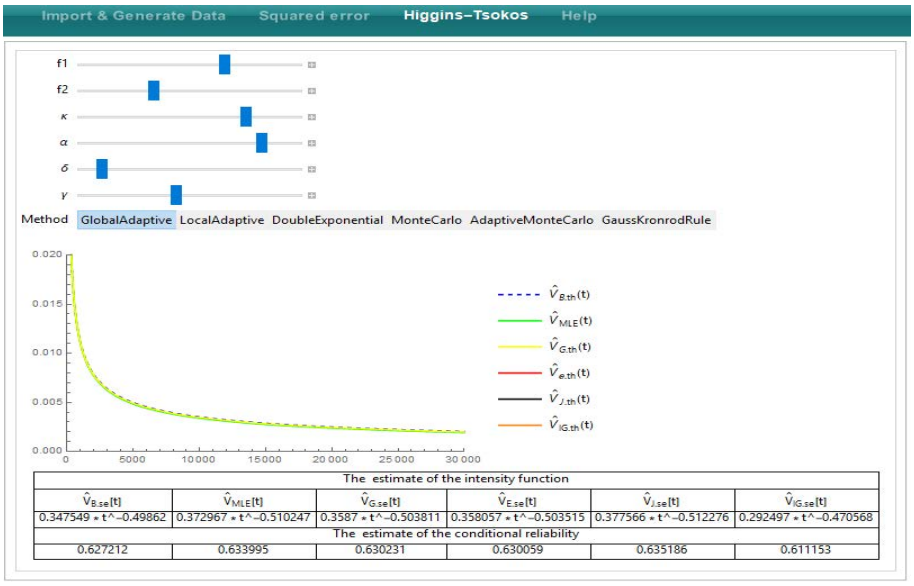


Figure 3.11.: Interactive User Interface Application to Estimate the Intensity and Reliability Functions of the Power Law Process for a Given Data.

user has several numerical integration techniques to choose from. Finally, the user can be further assisted, if needed, by choosing the following option: **Help**.

3.4 Contributions

We developed the analytical Bayesian estimates of the key parameter β, under Higgin-Tsokos and squared-error loss functions, in the intensity function where the underlying failure distribution is the Power Law Process, that is used for software reliability assessment, among others. The reliability function of the subject model is written analytically as a function of the intensity function.
The behavior of the key parameter β is characterized by the Burr type XII probability distribution. Real data and numerical simulation were used to illustrate not only the robustness of the squared-error loss function being challenged by the assumption of the Higgins-Tsokos loss function, but also the efficiency improvement in the estimation of the intensity function of PLP under Higgins-Tsokos loss function.

For $100,000$ samples of software failure times, based on Monte Carlo simulations and sample size of 40, we found the Bayesian estimate of β under Higgins-Tsokos loss function performed better than the Bayesian estimate of β under squared-error loss function with respect to three different values of θ. Even for different sample sizes similar results were achieved using β=0.7054, θ=1.7441, and averaged over $10,000$ samples of software failure times.

As the MLE of the scale parameter (θ) in the intensity function depends on the estimate of β, the adjusted estimate of θ using the Bayesian estimate of β under Higgins-Tsokos loss function provided better performance compared to the adjusted estimate of θ using the Bayesian estimate of β under squared-error loss function. Moreover, the Relative Efficiency was used to compare the intensity function estimations, mainly using MLEs for both β and θ, using Bayesian estimate of β under the squared-error loss function and the adjusted Bayesian of θ, and using Bayesian estimate of β under the Higgins-Tsokos loss function and the adjusted Bayesian of θ, showing that the Bayesian of the intensity function under Higgins-Tsokos loss function is more efficient in estimating the intensity function $V(t)$ with about 12% estimation improvement.

With respect to the question: Is the Bayesian estimate of the intensity function of the PLP sensitive to the selections of the prior, both parametric and non-parametric priors, and loss function? Using the developed algorithm, the Bayesian estimate of the intensity under Higgins-Tsokos loss function performed better than Bayesian estimate of the intensity under squared-error loss function. Based on the findings, it is recommended to choose a value of f_1 larger than a value of f_2 when the engineer thinks the prior knowledge of β is best characterized by Burr or Kernel based probability distributions with a proper justification, whereas a choice of f_1 less than f_2 with Jeffery's prior is suggested when the engineer does not have a prior knowledge of β.

Thus, based on this aspect of our analysis, we can conclude that the Bayesian analysis approach under Higgins-Tsokos loss function not only as robust as the Bayesian analysis approach under squared error loss function but also performed better, where both are superior to the maximum likelihood approach in estimating the reliability function of the Power Law Process. The interactive user interface application can be used without any prior coding knowledge to compute and visualize the Bayesian and maximum likelihood estimates of the intensity and reliability functions of the Power Law Process for a given data.

Chapter 4

Copula-Based Bayesian Reliability Analysis To The Power Law Process

In this chapter, we utilize Copula theory to develop a bivariate probability distribution for the parameters of the intensity function of the Power Law Process (PLP). In addition, we investigate the effectiveness of Copula-based Bayesian analysis using the squared-error loss function for modeling software failure times. We use the PLP as the underlying failure distribution subject to using the developed bivariate probability density function (PDF) as a bivariate prior of β and θ.

The chapter is organized as follows: Section 2 describes the theory and development of the Copula-based Bayesian reliability estimate; Section 3 presents the results and discussion; The contributions are given in Section 4.

4.1 Introduction

The Power Law Process (PLP) has been widely used since it was proposed by Crow [25] in the early 1970s. The PLP was shown to be useful in the assessment of software reliability, as well as in the fields of cybersecurity, breast and skin cancer treatments' effectiveness, finance, and transportation. It is also known as the Non-Homogeneous Process (NHPP) with an intensity function given by $V(t;\beta,\theta) = \frac{\beta}{\theta}\left(\frac{t}{\theta}\right)^{\beta-1}$. Some authors refer to it as Weibull process; we follow Ascher and Bar-Lev, [8, 10], using the term Power Law Process.

In the assessment of software reliability the estimate of the key parameter β in the $V(t;\beta,\theta)$ has an important role in evaluating the reliability of a software package. When the estimate of β is larger than 1, it indicates deterioration in software reliability. When the estimate of β is less than 1, it indicates improvement in software reliability. The homogeneous Poisson process is obtained when the estimate of β is equal to 1. In addition, the intensity function of the PLP, being a time-dependent function, provides relevant information about the rate of change of software reliability. Therefore, the PLP parameters, β and θ, have an out-sized role in the estimation process. As such, several researchers have proposed approaches to improve the estimates of β and θ.

Authors such as Calabria, Tsokos, and Rao, [19, 97, 96], have considered a frequency viewpoint when improving the estimate of the intensity function, $V(t;\beta,\theta)$. Alternately, Goel, Okumoto, Guida, Calabria, Tsokos, Molinares, and Alenezi, [40, 42, 20, 65, 5], have considered a Bayesian approach. Most previous efforts using the Bayesian perspective were focused on assuming independent priors for the parameters β and θ. Having said that, with ordered times $t = (t_1, t_2, ..., t_n)$, the Likelihood function for the first n failure times of the PLP $T_1 = t_1, T_2 = t_2, ..., T_n = t_n$ is written as:

$$L(t,\beta,\theta) = \exp\left(-\left(\frac{t_n}{\theta}\right)^{\beta}\right)\left(\frac{\beta}{\theta}\right)^{n}\prod_{i=1}^{n}\left(\frac{t_i}{\theta}\right)^{\beta-1}. \tag{4.1.1}$$

The approximate maximum likelihood estimate (MLE) for the shape parameter is given by:

$$\hat{\beta}_n = \frac{n}{\sum_{i=1}^{n}\log\left(\frac{t_n}{t_i}\right)}, \tag{4.1.2}$$

and for the scale parameter is:

$$\hat{\theta}_n = \frac{t_n}{n^{1/\hat{\beta}_n}}. \tag{4.1.3}$$

The MLE of β (4.1.2) is a function of the largest failure time and the MLE of θ depends on the MLE of β. This justifies applying Bayesian analysis to the PLP; as the parameters estimates vary depending on the last failure time, [65, 5]. The dependency structure of the parameters estimates, as seen in (4.1.3) have not been considered when developing Bayesian estimates.

In the present study, we utilize Copula theory to develop a bivariate probability distribution for the PLP parameters β and θ. We investigate the effectiveness of the Copula-based Bayesian analysis under the squared-error loss function for modeling software failure times. To accomplish this, we use the PLP as the underlying failure distribution, using the developed bivariate PDF as a bivariate prior of β and θ. Therefore, the primary objective of the study in this chapter is to answer the following questions within the framework of Copula theory and Bayesian analysis:

1. Is the Copula theory applicable to the development of a bivariate probability distribution for the PLP intensity function parameters, β and θ?

2. What are the performances of the Copula-based Bayesian estimates of β and θ under the squared-error loss function compared to their MLEs when modeling software failure times using

the PLP?

3. What are the performances of the Copula-based Bayesian estimates of β and θ under the squared-error loss function compared to the Bayesian estimate of β under the Higgins-Tsokos loss function and Jeffery prior for a fixed θ when modeling software failure times using the PLP?

4.2 Theory and Bayesian Estimates

4.2.1 Review of the Analytical PLP

The probability of achieving n failures of a given system in the time interval $(0, t]$ can be written as [5, 65]:

$$P(x = n; t) = \frac{\exp\left\{-\int_0^t V(x;\beta,\theta)dx\right\}\left\{\int_0^t V(x;\beta,\theta)dx\right\}^n}{n!},\ t > 0. \tag{4.2.1}$$

The PLP is obtained by reducing the previous expression into

$$P(x = n; t) = \frac{1}{n!}\exp\left\{-\frac{t}{\theta}^{\beta}\right\}\frac{t}{\theta}^{n\beta}. \tag{4.2.2}$$

When the PLP is the underlying failure model of the ordered failure times t_1, t_2, t_3,... , t_{n-1}, and t_n, the conditional reliability function of t_n given t_1, t_2, t_3,... , t_{n-1} can be written mathematically as a function of the intensity function, given by:

$$R(t_n|t_1, t_2, ..., t_{n-1}) = \exp\left\{\int_{t_{n-1}}^{t_n} -V(t;\beta,\theta)dt\right\},\ tn > t_{n-1} > 0, \tag{4.2.3}$$

since it is independent of the ordered t_1, t_2, t_3, ... , t_{n-2}.

4.2.2 Review of Copula Theory

Linguistically, *Copula*, C , means join together, which is from the Latin word *Copulare*. Classically, it is a function that joins several probability distribution functions to their individual marginal distribution functions [69]. It is used to capture the dependencies among multiple variables.

$$\text{Joint Distribution} = \text{Copula} + \text{Marginal Distributions}\,. \tag{4.2.2.1}$$

Consider a univariate case; for an arbitrary random variable (continuous) X with its cumulative

distribution and probability density functions F_X and f_X, respectively. They are defined in such a way that for any set of values $\mathbf{x}$ of the random variable, the following hold:

$$\mathbb{P}\left(X \in \mathbf{x}\right) \equiv \int_{\mathbf{x}} f_X(x)dx, \tag{4.2.2.2}$$

and

$$F_X(x) \equiv \int_{-\infty}^{x} f_X(z)dz\,. \tag{4.2.2.3}$$

If we substitute x by X (using all values) in (4.2.2.3) and equalize it to U ($F_X(X) = U$), we obtain what is called a grade of X. Theoretically, U follows a Standard Uniform probability distribution ($U \sim U_{[0,1]}$), since $F_U(u) \equiv \mathbb{P}\left(U \leq u\right) = \mathbb{P}\left(F_X(X) \leq u\right) = \mathbb{P}\left(X \leq F_X^{-1}(u)\right) = F_X(F_X^{-1}(u)) = u$. Conversely, if we feed U (any uniform random variable) to the inverse cumulative distribution function F_X^{-1}, we then obtain a random variable X with a probability distribution f_X, which is the Inverse Transform method. It is also useful in generating random variables from a desired probability distribution using uniform random variables (i.e. as in the Monte Carlo Simulation).
Now, consider a 2-dimensional vector of random variables $\mathbf{X} \equiv (X_1, X_2)^{'}$ with bivariate cumulative distribution and probability density functions $F_{\mathbf{X}}$ and $f_{\mathbf{X}}$, respectively. They are defined for any set of joint values $\mathbf{x_1}$ and $\mathbf{x_2}$ in $\mathbb{R}^2$ of the random variables $\mathbf{X}$ as follows:

$$\mathbb{P}\left((X_1, X_2) \in (\mathbf{x_1}, \mathbf{x_2})\right) \equiv \int_{(\mathbf{x_1}, \mathbf{x_2})} f_{\mathbf{X}}(\mathbf{x_1}, \mathbf{x_2})dx_1dx_2 \tag{4.2.2.4}$$

The marginal probability density function of X_1 is defined by:

$$f_{\mathbf{X_1}}(\mathbf{x_1}) \equiv \int_{\mathbb{R}^1} f_{\mathbf{X}}(\mathbf{x_1}, \mathbf{x_2})dx_2\,. \tag{4.2.2.5}$$

Then, marginal cumulative distribution function can be defined as in (4.2.2.5), and similarly the following hold:

$$U_n \equiv F_{X_n}(X_n) \sim U_{[0,1]}, \quad \text{where n } = 1, 2. \tag{4.2.2.6}$$

Note that the values of $\mathbf{U} \equiv (U_1, U_2)^{'}$ are dependent and then the $f_{\mathbf{U}}$ (with a domain $[0,1] \times [0,1]$) is not uniform in its domain. Therefore, for an arbitrary probability distribution $f_{\mathbf{X}}$, the Copula is the joint probability distribution $f_{\mathbf{U}}$ of its marginal grades. That is,

$$\begin{pmatrix} U_1 = F_{X_1}(X_1) \\ U_2 = F_{X_2}(X_2) \end{pmatrix} \sim f_{\mathbf{U}}\,. \tag{4.2.2.7}$$

This is the joint information among the random variables, since feeding the random variable X_n into its own cumulative distribution function, sweeping away the information in each marginal probability distribution. That is,

$$\begin{aligned} F_{\mathbf{U}}(\mathbf{u}) &\equiv \mathbb{P}\left(U_1 \leq u_1, U_2 \leq u_2\right) \\ &\equiv \mathbb{P}\left(F_{X_1(X_1)} \leq u_1, F_{X_2(X_2)} \leq u_2\right) \\ &\equiv \mathbb{P}\left(X_1 \leq F_{X_1}^{-1}(u_1), X_2 \leq F_{X_2}^{-1}(u_2)\right) \\ &\equiv F_X\left(F_{X_1}^{-1}(u_1), F_{X_2}^{-1}(u_2)\right)\,. \end{aligned} \tag{4.2.2.8}$$

Definition: (**Copula**) A d-dimensional copula is a function C: $[0,1]^d \longrightarrow [0,1]$ with the following properties [69]:

I) For every $\mathbf{U} \equiv (U_1, U_2, ..., U_d)' \in [0,1]^d$, $C(\mathbf{U})= 0$ if at least one cordinate of $\mathbf{U}$ is 0. And if all cordinates of $\mathbf{U}$ are 1 except $\mathbf{U_k}$, $k \in \{1, ..., d\}$, then $C(\mathbf{U})= C(\mathbf{U_k})$.

II) C is d-increasing.

Theorem: (**Sklar's Theorem 1959**)

Let F be a d-dimensional distribution function with margins $F_1, F_2, ..., F_d$. Then there exists a d-dimensional Copula, C, such that for all $\mathbf{X} \equiv (X_1, X_2, ..., X_d)' \in \mathbb{R} \cup (-\infty, \infty)^d$:

$F(X_1, X_2, ..., X_d) = \mathrm{C}(F_1(X_1), F_2(X_2), ..., F_d(X_d))$, [69].

If $F_1, F_2, ..., F_d$ are all continuous, then the copula C is unique.

Sklars theorem relates the joint distribution of $\mathbf{X}$, the Copula ($f_{\mathbf{U}}$), and the marginals ($F_{X_d}(\mathbf{x}_d)$). That is,

$$\begin{aligned} &\underbrace{f_{\mathbf{X}}(X_1, ..., X_d)} = \underbrace{f_{\mathbf{U}}(U_1, ..., U_d)} \times (\underbrace{f_{X_1} \times ..... \times f_{X_d}}), \\ &\text{Joint Distribution} = \text{Copula Density} \times Marginal \end{aligned} \tag{4.2.2.9}$$

and

$$c(F(X_1) \times ..... \times F(X_d)) = f_{\mathbf{U}}(U_1, ..., U_d) = \frac{\partial^d C(U_1, ..., U_d))}{\partial U_1 \times ..... \times \partial U_d}, \tag{4.2.2.10}$$

where $c(F(X_1) \times ..... \times F(X_d))$ is the Copula density. Thus, for a bivariate distribution development, the joint and conditional densities can be derived as follows:

$$
\begin{aligned}
&f(X_1, X_2) = f(X_1).f(X_2), c(F(X_1), F(X_2)) \\
&f(X_1|X_2) = f(X_1), \frac{c(F(X_1),F(X_2))}{c(F(X_2))}, \\
&and \\
&f(X_2|X_1) = f(X_2).\frac{c(F(X_1),F(X_2))}{c(F(X_1)}.
\end{aligned}
\tag{4.2.2.11}
$$

Thus far, we illustrated the Copula theory. Next, we show the development process for a bivariate probability distribution of the PLP parameters.

4.2.3 Copula-Based Bayesian Reliability Estimate

4.2.3.1 Development of Bivariate Probability Distribution Of the PLP Parameters

Table 4.1, below, contains the MLEs of β and θ. We collected the first two failure times, then the first three failure times, and so on. It can be noted that the variation of the MLE for β exists since it depends on the largest time to failure, which creates the motive to apply Bayesian analysis in the PLP.

Figure 4.1, below, represents the scatter plot of MLEs of β and θ without taking the first failure time into account.

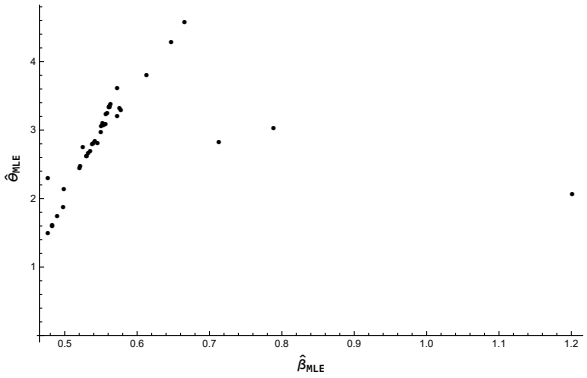


Figure 4.1.: Scatter plot of β and θ estimates with first failure time excluded

It can be seen that there are at least three possible outliers. Since the MLEs of β do not follow a Normal distribution, Z-scores (Z_s) might not be appropriate to check for outliers. Iglewicz and Hoaglin (1993) suggested the modified Z-scores ($Z_{m.s}$) criteria for checking whether outliers are present in the data. They suggested that an absolute value of larger than 3.5 should be labled as an outlier. The $Z_{m.s}$ is given by:

Table 4.1: Crow failure data and parameters' approximate maximum likelihood estimates of the Power Law Process

Time	Failure Times	β estimates	θ estimates
1	0.7	Inf	0.7
2	3.7	1.2	2.08
3	13.2	0.71	2.81
4	17.6	0.79	3.04
5	54.5	0.52	2.47
6	99.2	0.48	2.37
7	112.2	0.53	2.85
8	120.9	0.58	3.35
9	151	0.58	3.42
10	163	0.61	3.74
11	174.5	0.65	4.36
12	191.6	0.67	4.7
13	282.8	0.57	3.14
14	355.2	0.55	2.93
15	486.3	0.5	2.16
16	490.5	0.53	2.62
17	513.3	0.55	2.97
18	558.4	0.56	3.2
19	678.1	0.53	2.62
20	688	0.55	2.97
21	785.9	0.54	2.8
22	887	0.53	2.6
23	1010.7	0.52	2.43
24	1029.1	0.54	2.86
25	1034.4	0.56	3.3
26	1136.1	0.55	3.04
27	1178.9	0.56	3.28
28	1259.7	0.56	3.28
29	1297.9	0.57	3.53
30	1419.7	0.56	3.27
31	1571.7	0.55	3.05
32	1629.8	0.56	3.34
33	1702.4	0.56	3.31
34	1928.9	0.54	2.81
35	2072.3	0.54	2.86
36	2525.2	0.5	1.95
37	2928.5	0.48	1.58
38	3016.4	0.48	1.54
39	3181	0.48	1.54
40	3256.3	0.49	1.75

$$Z_{m.s} = \frac{0.6745(x_i - \tilde{x})}{MAD}, \tag{4.2.3.1.1}$$

where x_i, $\tilde{x}$, and MAD are the data, median of the data, and median absolute deviation (median of $|x_i - \tilde{x}|$), respectively. Table 4.2 shows the Z_s and $Z_{m.s}$ values for the MLEs of β and θ. As expected, the Z_s was not able to identify all possible outliers, whereas $Z_{m.s}$ identified four potential outliers.

In order to better estimate the $V(t;\beta,\theta)$, we investigated the relationship between $\hat{\beta}_{MLE}$ and $\hat{\theta}_{MLE}$. Pearson correlation was computed, and the correlation estimate was insignificant (estimate of the correlation coefficient was 0.16) at the 0.05 level of significance. But, after excluding the potential outliers, the Pearson correlation was significant (P value approximately zero) and the estimate of the correlation coefficient was 0.96. Kolmogorov-Smirnov goodness of fit test was applied to find the approximate distributions that best describe the behaviors of the MLEs of β

Table 4.2: Z-scores (Z_s) and Z-modified scores ($Z_{m.s}$) for MLEs of β and θ using Crow dataset

$Z_{s.\widehat{\beta}_{MLE}}$	$Z_{m.s.\widehat{\beta}_{MLE}}$	$Z_{s.\widehat{\theta}_{MLE}}$	$Z_{m.s.\widehat{\theta}_{MLE}}$
5.29	**21.92**	1.16	1.64
1.16	**5.4**	0.09	0.23
1.83	**8.09**	0.25	0.21
0.44	1.01	0.59	0.89
0.78	2.36	0.74	1.08
0.36	0.67	0.03	0.15
0.06	1.01	0.71	0.81
0.06	1.01	0.81	0.94
0.32	2.02	1.28	1.56
0.65	3.37	2.2	2.76
0.82	**4.05**	2.7	**3.41**
0.02	0.67	0.4	0.4
0.19	0	0.09	0
0.61	1.69	1.05	1.48
0.36	0.67	0.37	0.6
0.19	0	0.15	0.08
0.11	0.34	0.49	0.52
0.36	0.67	0.37	0.6
0.19	0	0.15	0.08
0.27	0.34	0.1	0.25
0.36	0.67	0.4	0.64
0.44	1.01	0.65	0.96
0.27	0.34	0.01	0.13
0.11	0.34	0.63	0.71
0.19	0	0.25	0.21
0.11	0.34	0.61	0.67
0.11	0.34	0.61	0.67
0.02	0.67	0.97	1.16
0.11	0.34	0.59	0.66
0.19	0	0.27	0.23
0.11	0.34	0.69	0.79
0.11	0.34	0.65	0.73
0.27	0.34	0.09	0.23
0.27	0.34	0.01	0.13
0.61	1.69	1.36	1.89
0.78	2.36	1.9	2.6
0.78	2.36	1.96	2.68
0.78	2.36	1.96	2.68
0.7	2.02	1.65	2.27

and θ. The best fit of the data was Dagum probability distribution for both MLEs of β and θ, with P-values of 0.698 and 0.94, respectively. The PDF and CDF of Dagum probability distribution are given by:

$$\begin{aligned} &f(x) = \tfrac{apx^{ap-1}}{[b^{ap}[1+(\frac{x}{b})^a]^{p+1}]}\,, \\ &and \\ &F(x) = [1+(\tfrac{x}{b})^{-a}]^{-p}\,, \end{aligned} \tag{4.2.3.1.2}$$

where b, a, and p are the scale, shape-1, and shape-2 parameters of the Dagum probability distribution, respectively. After proving the significant association among the MLEs of β and θ, and the approximate probability distributions that best fit them, we carry on the analysis and apply Copula theory to develop the bivariate distribution that best describes the behavior of the MLEs of β and θ, using their univariate probability distributions and their dependency structure.

To select the best Copula, we developed the procedure displayed in Figure 4.2 below.

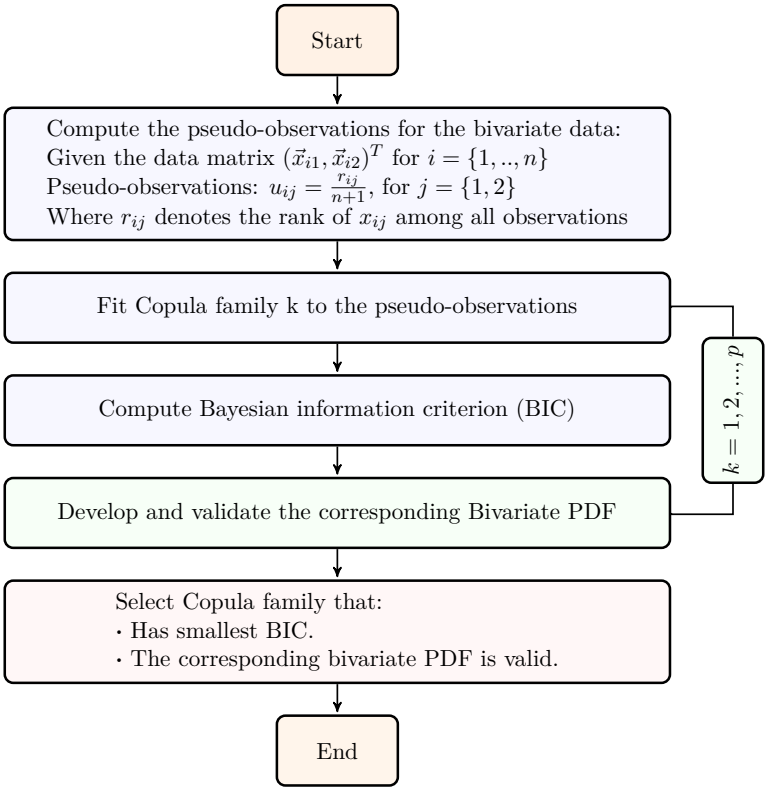


Figure 4.2.: Selection procedure of a Copula that fits the MLE of β and θ

Using the bivariate data (MLEs of β and θ), we fit all existing Copulas and then chose the Frank Copula function, [35], based on its smallest Bayesian information criterion (BIC) while the developed bivariate probability distribution maintains the following conditions:

1. $f(\beta, \theta) \geq 0$, for $0 < \beta < \infty$ and $0 < \theta < \infty$,

and

2. $\int_0^\infty \int_0^\infty f(\beta, \theta)\ d\beta d\theta = 1$.

Table 4.3, shows the analytical forms of Frank Copulas probability distribution and density functions.

Using (4.2.2.11) and Table 4.3, we developed the approximate bivariate probability density function that best characterizes the bivariate behavior of β and θ (Biv-f(β, θ)), the parameters of the PLP by substituting u and v with the CDFs of MLEs of β and θ, respectively. It is given by:

$$\text{Biv.f}(\beta, \theta) = C_d(\beta, \theta) \cdot f(\beta) \cdot f(\theta), \tag{4.2.3.1.3}$$

Table 4.3: Analytical forms of Frank Copula distribution and density functions.

Frank Copula	**Analytical form**
Distribution Copula (C):	$=-\frac{\log\left(\frac{\left(e^{\alpha(-u)}-1\right)\left(e^{\alpha(-v)}-1\right)}{e^{-\alpha}-1}+1\right)}{\alpha}$
Density Copula (C_d):	$=\frac{(e^{\alpha}-1)\alpha e^{\alpha(u+v+1)}}{\left(e^{\alpha}-e^{\alpha(u+1)}+e^{\alpha(u+v)}-e^{\alpha(v+1)}\right)^2}$

where:

$$C_d = \frac{\left(1-e^{-\alpha}\right)\alpha\exp\left(-\alpha\left(\left(\left(\frac{\beta}{b_1}\right)^{-a_1}+1\right)^{-p_1}+\left(\left(\frac{\theta}{b_2}\right)^{-a_2}+1\right)^{-p_2}\right)\right)}{\left(-\left(1-e^{-\alpha\left(\left(\frac{\beta}{b_1}\right)^{-a_1}+1\right)^{-p_1}}\right)\left(1-e^{-\alpha\left(\left(\frac{\theta}{b_2}\right)^{-a_2}+1\right)^{-p_2}}\right)-e^{-\alpha}+1\right)^2},$$

$$f(\beta) = a_1 p_1 b_1^{-a_1 p_1}\beta^{a_1 p_1-1}\left(\left(\frac{\beta}{b_1}\right)^{a_1}+1\right)^{-p_1-1},$$

and

$$f(\theta) = a_2 p_2 b_2^{-a_2 p_2}\theta^{a_2 p_2-1}\left(\left(\frac{\theta}{b_2}\right)^{a_2}+1\right)^{-p_2-1}.$$

The MLEs of α for Frank Copula and a_1, p_1, b_1, a_2, p_2, and b_2 for Dagum PDFs (of β and θ), respectively, are used in the next sections.

4.2.3.2 Bayesian MLE Reliability Estimate of The PLP

The Bayesian estimate of β with respect to the Higgins-Tsokos loss function (defined in equation 3.2.2.1) and Jeffreys' PDF, as the prior, was presented in previous chapters, and is given by:

$$\hat{\beta}^{J}_{B.HT} = \frac{1}{f_1+f_2}\ln[\frac{\int_0^{\infty}\exp\{f_1\beta\}\,h_J(\bar{t}|\beta)d\beta}{\int_0^{\infty}\exp\{-f_2\beta\}\,h_J(\bar{t}|\beta)d\beta}], \tag{4.2.3.2.1}$$

where the posterior PDF of β given data (t), $h(\beta|t)$, using the Bayes' theorem, is given by:

$$h_J(\bar{t}|\beta) = \frac{\exp\left\{\left(\frac{t_n}{\theta}\right)^{\beta}\right\}\frac{\beta^{n-1}}{\theta^{n\beta}}\prod_{i=1}^{n}(t_i)^{\beta-1}}{\int_0^{\infty}\exp\left\{\left(\frac{t_n}{\theta}\right)^{\beta}\right\}\frac{\beta^{n-1}}{\theta^{n\beta}}\prod_{i=1}^{n}(t_i)^{\beta-1}d\beta}. \tag{4.2.3.2.2}$$

The Bayesian MLE estimate of the scale parameter of the PLP intensity function is given by:

$$\hat{\theta}^{J}_{B.HT} = \frac{t_n}{n^{1/\hat{\beta}^{J}_{B.HT}}}. \tag{4.2.3.2.3}$$

The Bayesian MLE estimate of the conditional reliability function of the PLP is obtained using

$\hat{\beta}^{J}_{B.HT}$ and $\hat{\theta}^{J}_{B.HT}$ of the PLP intensity function parameters. That is,

$$\hat{R}^{J}_{B.HT}(t_n|t_1,t_2,...,t_{n-1}) = \exp\left\{\int_{t_{n-1}}^{t_n} -V(t;\hat{\beta}^{J}_{B.HT},\hat{\theta}^{J}_{B.HT})dt\right\},\ tn > t_{n-1} > 0, \qquad (4.2.3.2.4)$$

since it is independent of the ordered t_1, t_2, t_3, ... , t_{n-2}.

We must rely on a numerical estimation because we cannot obtain a close solution for $\hat{\beta}^{J}_{B.HT}$. Also note that it depends on knowing or being able to estimate the scale parameter θ.

4.2.3.3 Copula-based Bayesian Reliability Estimate of the PLP

The Bayesian estimates of the parameters β and θ of the the PLP depend on a prior probability distribution and a loss function, along with the likelihood function of the PLP. The Bayesian estimates of the β ($\hat{\beta}_{B.BSE}$) and θ ($\hat{\theta}_{B.BSE}$), under the squared-error loss function and the developed bivariate probability distribution as a prior (4.2.3.1.3) are given by:

$$\hat{\beta}_{B.BSE} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \beta \cdot h(\beta,\theta|t)d\beta d\theta, \qquad (4.2.3.3.1)$$

and

$$\hat{\theta}_{B.BSE} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \theta \cdot h(\beta,\theta|t)d\beta d\theta, \qquad (4.2.3.3.2)$$

where the posterior PDF of β and θ given data (t), $h(\beta,\theta|t)$, using the Bayes' theorem, is given by:

$$h(\beta,\theta|t) = \frac{L(t|\beta,\theta) \times \text{Biv.f}(\beta,\theta)}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} L(t|\beta,\theta) \times \text{Biv.f}(\beta,\theta)d\beta d\theta}. \qquad (4.2.3.3.3)$$

When the PLP is the underlying failure model of the ordered failure times t_1, t_2, t_3,... , t_{n-1}, and t_n, the conditional reliability function of t_n given t_1, t_2, t_3,... , t_{n-1} can be written mathematically as a function of the intensity function, given by: The Bayesian estimate of the conditional reliability function of the PLP is obtained using the Copula-based estimates of the PLP intensity function parameters. That is,

$$\hat{R}_{B.BSE}(t_n|t_1,t_2,...,t_{n-1}) = \exp\left\{\int_{t_{n-1}}^{t_n} -V(t;\hat{\beta}_{B.BSE},\hat{\theta}_{B.BSE})dt\right\},\ tn > t_{n-1} > 0, \qquad (4.2.3.3.4)$$

since it is independent of the ordered t_1, t_2, t_3, ... , t_{n-2}.

We cannot obtain closed analytical forms of the Bayesian estimates, $\hat{\beta}_{B.BSE}$ and $\hat{\theta}_{B.BSE}$, thus we must utilize numerical methods to obtain the subject estimates.

Table 4.4 shows the acronyms and notations used in this study.

Table 4.4: Acronyms and notations used in this study

Acronyms	
PLP	Power Law Process.
MLE	Maximum likelihood estimate..
PDF	Probability density function.
CDF	Cumulative density function.
C	Distribution Copula.
C_d	Density Copula.
Notations	
β and θ	Shape and Scale parameters of PLP.
T_1, T_2, .., T_n	First n successive failure times of the PLP.
$V(t;\beta,\theta)$	Intensity function of the PLP.
RE	Relative efficiency.
$AMISE$	Asymptotic mean integrated squared error.
H-T	Higgin-Tsokos.
S-E	Squared-Error.
$\hat{\beta}^J_{HT}$	Bayesian estimate of β under H-T loss function and Jeffreys prior.
$\hat{\theta}^J_{HT}$	Bayesian MLE estimate of θ using MLE and $\hat{\beta}^J_{HT}$.
$\hat{\beta}_{B.BSE}$	Copula-based Bayesian estimate of β under S-E loss function.
$\hat{\theta}^J_{B.BSE}$	Copula-based Bayesian estimate of θ under S-E loss function.
$\hat{V}^J_{HT}$	Bayesian MLE estimate of $V(t;\beta,\theta)$ under H-T loss function.
$\hat{V}_{B.BSE}$	Copula-based Bayesian estimate of $V(t;\beta,\theta)$ under S-E loss function.
Bayesian C.	Copula-based Bayesian estimate.
Bayesian J.	Bayesian estimate under Jeffreys prior.

4.3 Results and Discussion

4.3.1 Numerical simulation

A Monte Carlo simulation was used to compare the Copula-based Bayesian (Bayesian C.) under the squared-error loss function, Bayesian with Jeffreys prior (Bayesian J.) under Higgins-Tsokos loss function, and the MLE approaches of the PLP intensity function parameters. The Bayesian estimate of the intensity function was also computed by substituting these estimates, from which the reliability function can be estimated. The Kronrod extension of a Gaussian quadrature rule

[74, 49] was used to numerically compute the Bayesian estimates of the PLP parameters, namely, β and θ. The numerical procedure we followed is shown in the following schematic diagram, Figure 4.3.

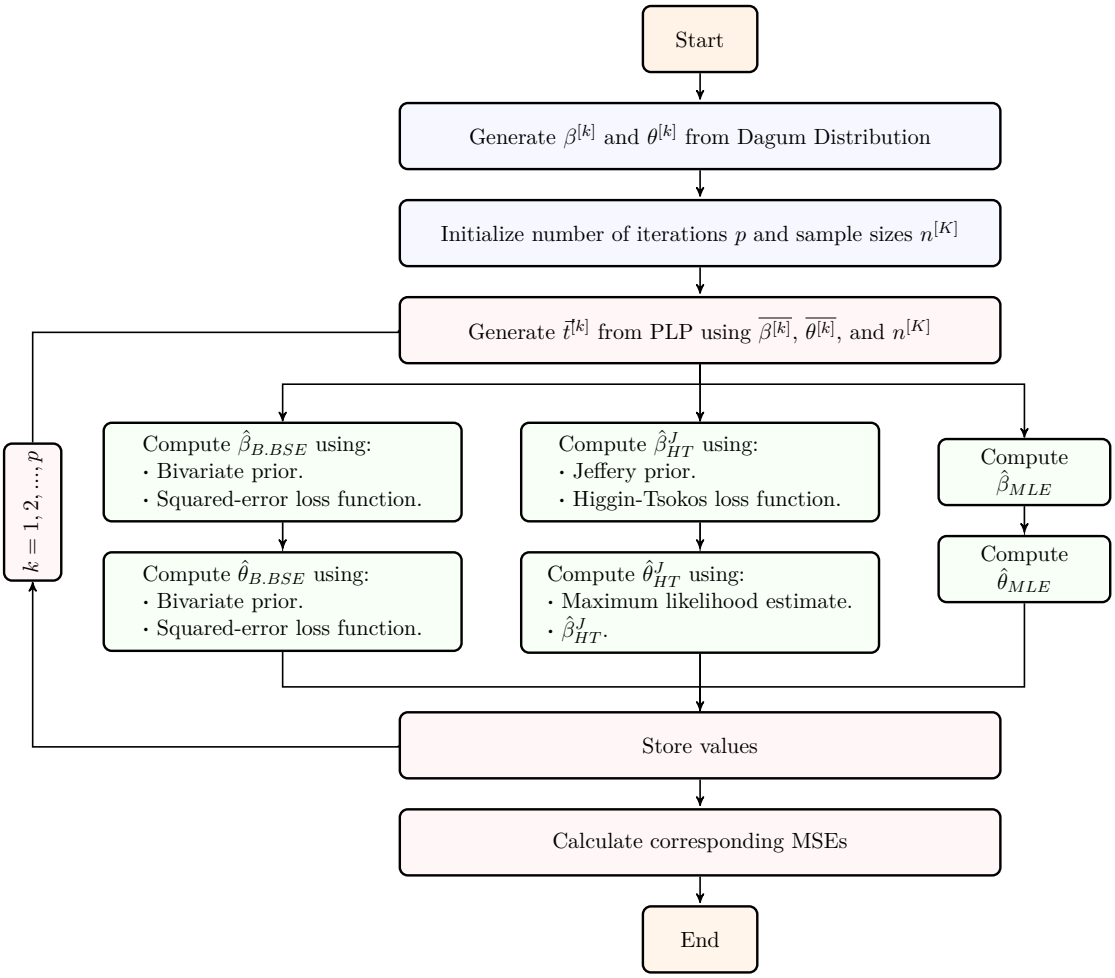


Figure 4.3.: Simulation to compare Bayesian and MLE estimates of the PLP intensity function parameters β and θ.

The simulation starts by generating random samples of the parameters β and θ from their corresponding Dagum PDFs. The number of iterations was set to $20,000$ for each sample size; the sample sizes selected were 10, 20, 30, 40, 60, 80, and 100. Failure times were then generated using the average values of the β and θ samples, repeatedly, for each sample size. Then, the Bayesian C., and Bayesian J. with a fixed θ=1.7441, and MLE estimates of the PLP intensity function parameters were computed in parallel. After storing all estimates, the mean square errors (MSEs) were computed as an evaluation metric to compare the results. The MLEs of α for Frank Copula and a_1, p_1, b_1, a_2, p_2, and b_2 for Dagum PDFs (of β and θ), respectively, were computed using Crow

failure times data (Table 4.1).

The results for the key parameter β are presented in Table 4.5.

Table 4.5: Averages of the Copula-based Bayesian, Jeffreys Bayesian, and maximum likelihood estimates of β

n	β	$\hat{\beta}$	$\hat{\theta}^{J}_{HT}$	$\hat{\beta}_{B.BSE}$
10	0.5439	0.873628	0.502337	0.543544
20	0.5439	0.704088	0.506783	0.545261
30	0.5439	0.652561	0.509773	0.545991
40	0.5439	0.627306	0.511788	0.546369
60	0.5439	0.601183	0.514179	0.546521
80	0.5439	0.588122	0.515737	0.546515
100	0.5439	0.579951	0.51684	0.546447

It can be seen that the Copula-based Bayesian estimate (Bayesian C.) of the parameter β under the squared-error loss function and the bivariate probability distribution function of β and θ as a prior converges to the true value faster than the Bayesian estimate under Higgins-Tsokos loss function and Jeffreys as a prior of the parameter β (Bayesian J.) and the MLE of the parameter β. This is true for small and large sample sizes. The Figure 4.4, visualizes the values in Table 4.5.

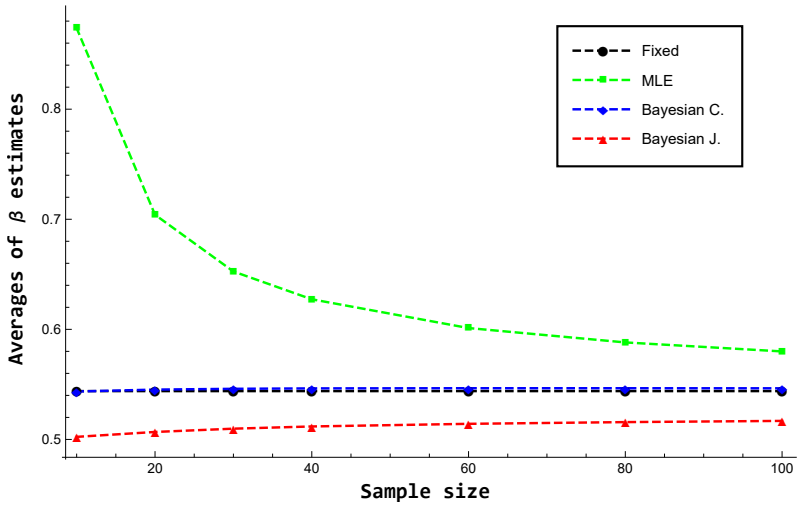


Figure 4.4.: Averages of β Bayesian and MLEs estimates versus sample size.

Although the Copula-based Bayesian estimate (Bayesian C.) of the parameter β tend to slightly underestimate the true value considering small sizes, it still performed better than the Bayesian estimate under Jeffreys prior and the MLE of β. Figure 4.5 shows the Bayesian estimates of the parameter β only. Clearly, we see that the average values of the Copula-based Bayesian estimates

under the bivariate prior of both parameters β and θ were closer to the true value compared to the Bayesian estimate of β under the Jeffrey prior where the latter considered the β parameter only to behave as a random quantity.

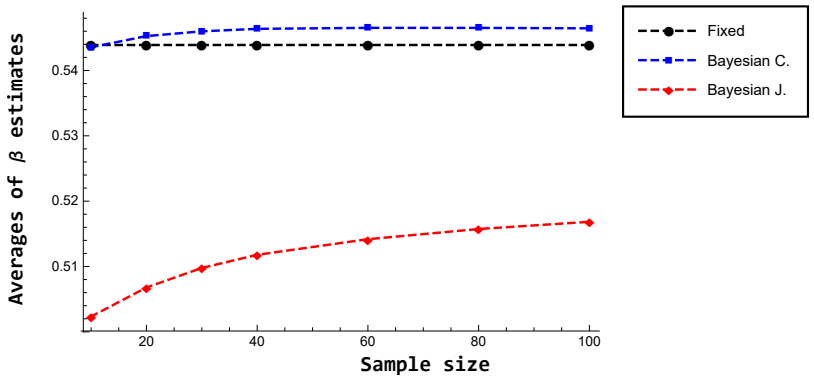


Figure 4.5.: Averages of β Bayesian estimates versus sample size.

The MSE values of both estimates are shown in Table 4.6.

Table 4.6: MSEs of the Copula-based Bayesian, Jeffreys Bayesian, and maximum likelihood estimates of β

n	MSE of $\hat{\beta}$	MSE of $\hat{\beta}_{HT}$	MSE of $\hat{\beta}_{B.BSE}$
10	0.219481	0.00544172	0.000140865
20	0.0555504	0.00249975	0.000186411
30	0.0278312	0.00177337	0.000185548
40	0.0177889	0.00142745	0.000167423
60	0.00964394	0.00110436	0.000133234
80	0.00653891	0.000943688	0.000110736
100	0.00479485	0.000843372	0.0000928318

Based on the MSE values, the Copula-based Bayesian estimate (Bayesian C.) of the parameter β under the squared-error loss function and the bivariate probability distribution function of β and θ as a prior outperformed the Bayesian estimate under Higgins-Tsokos loss function and Jeffreys of the parameter β as a prior (Bayesian J.) and the MLE of the parameter β across the sample sizes. Figure 4.6 presents the MSE values of the Bayesian estimates of β considering different sample sizes.

Figure 4.6 shows that the Copula-based Bayesian estimates of the parameter β has lower MSE values than the Jeffreys Bayesian estimates. This is true for different sample sizes.

The results for the parameter θ are presented in Table 4.7.

Across sample sizes, the Copula-based Bayesian estimates of the scale parameter θ, under the

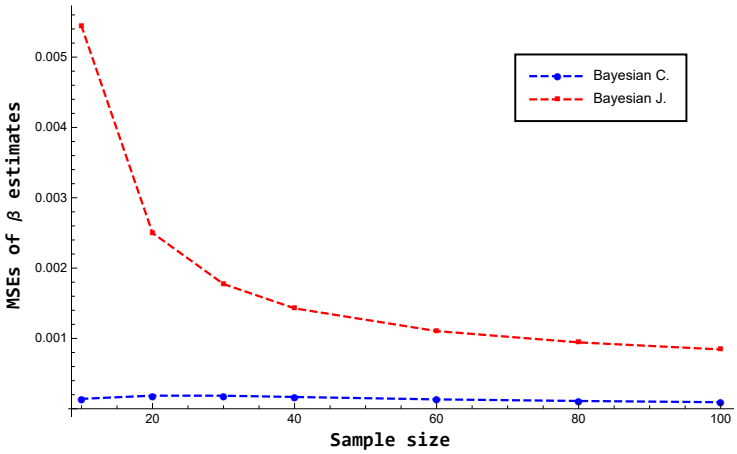


Figure 4.6.: MSE of β Bayesian estimates versus sample size.

Table 4.7: Averages of the Copula-based Bayesian, Jeffreys Bayesian, and maximum likelihood estimates of θ.

n	θ	$\hat{\theta}$	$\hat{\theta}_{HT}$	$\hat{\theta}_{B.BSE}$
10	2.7834	19.0286	2.05062	2.9012
20	2.7834	13.9351	1.90993	2.91079
30	2.7834	11.2111	1.86419	2.91956
40	2.7834	9.61614	1.84117	2.92588
60	2.7834	7.73523	1.81698	2.9339
80	2.7834	6.74773	1.80447	2.9379
100	2.7834	6.07977	1.79666	2.94014

squarred-error loss function and the bivariate probability distribution of β and θ as a prior had an excellent performance compared to the Bayesian MLE and the MLE. The MLE of θ tends to overestimate the true value, which was also shown in previous studies [65, 5]. Figure 4.7 visualizes the values in Table 4.7.

The averages of the θ's Copula-based Bayesian estimates were much more steady around the true value (θ=2.7834) than the averages of the θ's MLEs and Bayesian MLEs. Figure 4.8 shows only the Bayesian estimates of the scale parameter θ.

Again, the Copula-based Bayesian estimate of θ, where both β and θ parameters are considered unknown random quantities, performed better than the Bayesian estimate of β assuming θ as a fixed known value and then using it to obtain the Bayesian MLE of θ (equation 4.2.3.2.3). Clearly, the Bayesian MLE estimate of θ tends to converge to the assumed fixed value, which shows the excellent performance of the θ's Copula-based Bayesian estimate.

The MSE values of all estimates are shown in Table 4.8.

The MSEs of the Bayesian estimates of the scale parameter θ were far less than the MSEs of the

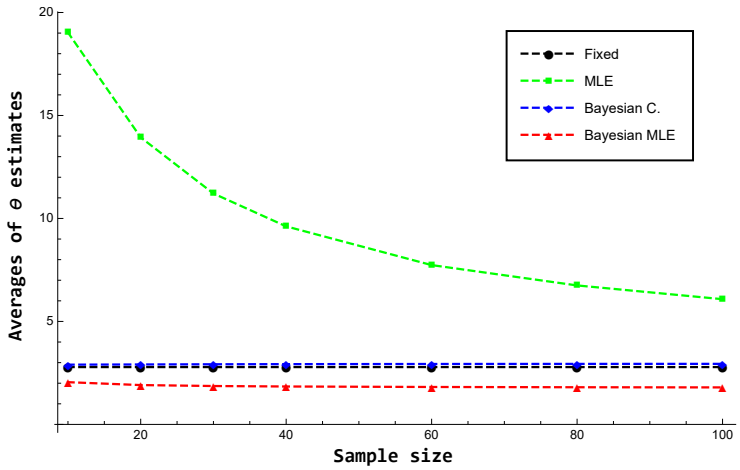


Figure 4.7.: Averages of θ Bayesian, Bayesian MLEs, and MLEs estimates versus sample size.

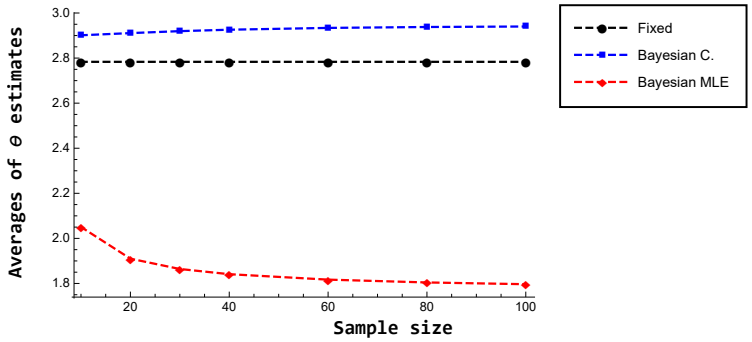


Figure 4.8.: Averages of θ Bayesian, Bayesian MLEs, and MLEs estimates versus sample size.

Table 4.8: MSEs of the Copula-based Bayesian, Jeffreys Bayesian, and maximum likelihood estimates of θ.

n	MSE of $\hat{\theta}$	MSE of $\hat{\theta}_{HT}$	MSE of $\hat{\theta}_{B.BSE}$
10	757.885	0.63582	0.0366226
20	390.787	0.801567	0.0387726
30	233.348	0.867125	0.0383609
40	161.632	0.902702	0.0382712
60	86.0931	0.942527	0.0388391
80	58.1365	0.964165	0.0399912
100	40.9529	0.977969	0.0412581

MLEs and Bayesian MLEs. Figure 4.9 visualizes only the Bayesian estimates in Table 4.8. Figure 4.9 shows the excellence of the Copula-based Bayesian estimates of θ compared to the Bayesian MLEs across different sample sizes.

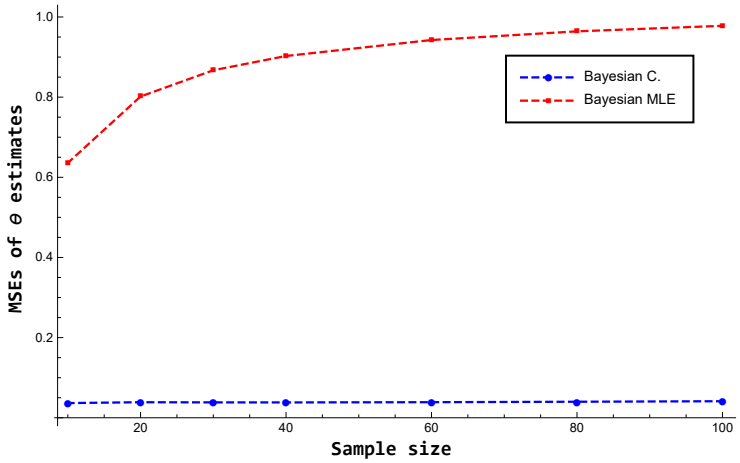


Figure 4.9.: MSEs of θ Bayesian estimates versus Bayesian MLEs sample size.

Thus far, we have shown that the Copula-based Bayesian estimates of the shape and scale parameters of the PLP intensity function are superior to the Bayesian estimates where only β is considered as an unknown random quantity and MLEs. We now evaluate the reliability function of the PLP (4.2.3), which is a function of the intensity function $V(t;\beta,\theta) = \frac{\beta}{\theta}\left(\frac{t}{\theta}\right)^{\beta-1}$. Therefore, evaluating the MLE and Bayesian estimates of the intensity functions compensates evaluating the same estimates of the reliability function. The relative efficiency (RE) was used in the evaluation process to compare the Copula-based Bayesian estimate of the intensity function with MLE and Bayesian MLE, respectively, which are given by:

$$RE(\hat{V}_{B.BSE}, \hat{V}_{MLE}) = \frac{IMSE(\hat{V}_{B.BSE})}{IMSE(\hat{V}_{MLE})} = \frac{\int_{-\infty}^{\infty}[\hat{V}_{B.BSE}(t) - V(t)]^2 dt}{\int_{-\infty}^{\infty}[\hat{V}_{MLE}(t) - V(t)]^2 dt}. \tag{4.3.1}$$

and

$$RE(\hat{V}_{B.BSE}, \hat{V}_{HT}^{J}) = \frac{IMSE(\hat{V}_{B.BSE})}{IMSE(\hat{V}_{HT}^{J})} = \frac{\int_{-\infty}^{\infty}[\hat{V}_{B.BSE}(t) - V(t)]^2 dt}{\int_{-\infty}^{\infty}[\hat{V}_{HT}^{J}(t) - V(t)]^2 dt}. \tag{4.3.2}$$

The RE measures the robustness of the intensity function using the Copula-based Bayesian estimates of β and θ ($\hat{V}_{B.BSE}$) compared to the intensity function using the MLEs of the same parameters ($\hat{V}_{MLE}$) and the intensity function using Jefferys Bayesian estimate of β and Bayesian MLE of θ ($\hat{V}_{HT}^{J}$).

$\hat{V}_{B.BSE}$ and $\hat{V}_{MLE}$ are considered equally effective when $RE = 1$. When $RE < 1$ and $RE > 1$, $\hat{V}_{B.BSE}$ is more and less effective, respectively, than $\hat{V}_{MLE}$. This interpretation is also applicable

when comparing $\hat{V}_{B.BSE}$ and $\hat{V}_{HT}^{J}$.

The MLE, Bayesian MLE, and Copula-based Bayesian estimates of the PLP intensity function, using the respective estimates of β and θ in Tables 4.5 and 4.7, are presented in Table 4.9.

Table 4.9: MSEs of the MLE and Bayesian estimates, under squared-error loss function, of β for different sample sizes.

n	$V(t)$	$\hat{V}_{MLE}$	$\hat{V}_{HT}^{J}$	$\hat{V}_{B.BSE}$
20	$0.31168 \cdot t^{-0.4561}$	$0.11017 \cdot t^{-0.2959}$	$0.3651 \cdot t^{-0.4932}$	$0.30451 \cdot t^{-0.4547}$
40	$0.31168 \cdot t^{-0.4561}$	$0.15165 \cdot t^{-0.3727}$	$0.37447 \cdot t^{-0.4882}$	$0.30391 \cdot t^{-0.453631}$
80	$0.31168 \cdot t^{-0.4561}$	$0.19135 \cdot t^{-0.4119}$	$0.39453 \cdot t^{-0.4561}$	$0.30326 \cdot t^{-0.453485}$
100	$0.31168 \cdot t^{-0.4561}$	$0.20360 \cdot t^{-0.4200}$	$0.38180 \cdot t^{-0.4832}$	$0.30312 \cdot t^{-0.453553}$

The REs of the MLE, Bayesian MLE, and Copula-based Bayesian estimates of the PLP intensity function are presented in Table 4.10.

Table 4.10: Relative efficiency of $\hat{V}_{B.BSE}$ compared to $\hat{V}_{MLE}$ and $\hat{V}_{HT}^{J}$.

n	$RE(\hat{V}_{BSE}, \hat{V}_{HT}^{J})$	$RE(\hat{V}_{BSE}, \hat{V}_{MLE})$
20	0.0019	0.0088
40	0.0052	0.0186
80	0.1030	0.0344
100	0.1190	0.0360

The Copula-based Bayesian estimate of the PLP intensity function ($\hat{V}_{B.BSE}$) is far more efficient than the MLE and the Bayesian MLE of the PLP intensity function, across small and large sample sizes. The REs also show a very small improvement in the MLE and the Bayesian MLE of the PLP intensity function as the sample size increases.

Furthermore, we generated 40 failure times from the PLP using values of 0.65 and 1.7441 for β and θ, respectively, and repeated the process $1,000$ times. Figure 4.10, below, shows the superiority of the Copula-based Bayesian estimate of the PLP intensity function compared to the MLE and Bayesian MLE of the PLP intensity function.

4.3.2 Using real data

We followed the algorithm that we used to generate the information in Figure 4.11 to obtain the Bayesian conditional reliability function for the real software reliability growth data given in Table 4.1.

Using the given real data, we computed the MLE, Bayesian MLE, and Copula-based Bayesian estimates of the PLP intensity function in order to obtain the respective estimates of the conditional

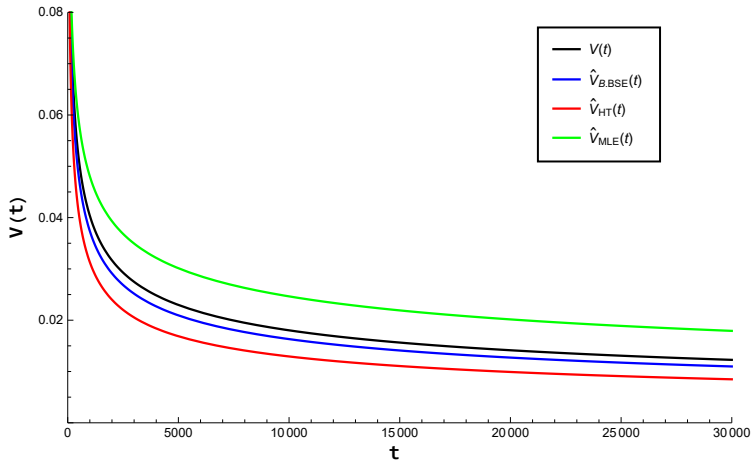


Figure 4.10.: MLE, Bayesian MLE, and Copula-based Bayesian estimates of the PLP intensity function for the true values 0.65 and 1.7441 of β and θ, respectively.

reliability function of the PLP.

The MLE, Jefferys Bayesian, and Copula-based Bayesian estimates of the PLP intensity function parameters using Crow failure times data are presented in Table 4.11.

Table 4.11: Parameters estimates to obtain the MLE, Bayesian MLE, and Copula-based Bayesian estimates of the PLP intensity function using Crow failure times

$\hat{\beta}_{MLE}$	$\hat{\theta}_{MLE}$	$\hat{\beta}^{J}_{HT}$	$\hat{\theta}^{J}_{HT}$	$\hat{\beta}_{B.BSE}$	$\hat{\theta}_{B.BSE}$
0.49	1.744	0.4893	1.732	0.527	2.592

The estimate of the key parameter β in the intensity function can be used in the software reliability's assessment. Although, all estimates in Table 4.11 indicate improvement in the software's reliability, the Copula-based Bayesian estimate shows less improvement compared to the other estimates. The MLE, Bayesian MLE, and Bayesian estimates of the PLP intensity function for the given real data are given by:

$$\hat{V}_{B.MLE}(t) = 0.373111 \cdot t^{-0.51},\ t > 0, \tag{4.3.2.1}$$

$$\hat{V}^{J}_{B.HT}(t) = 0.377756 \cdot t^{-0.512},\ t > 0, \tag{4.3.2.2}$$

and

$$\hat{V}_{B.BSE}(t) = 0.319025 \cdot t^{-0.473},\ t > 0. \tag{4.3.2.3}$$

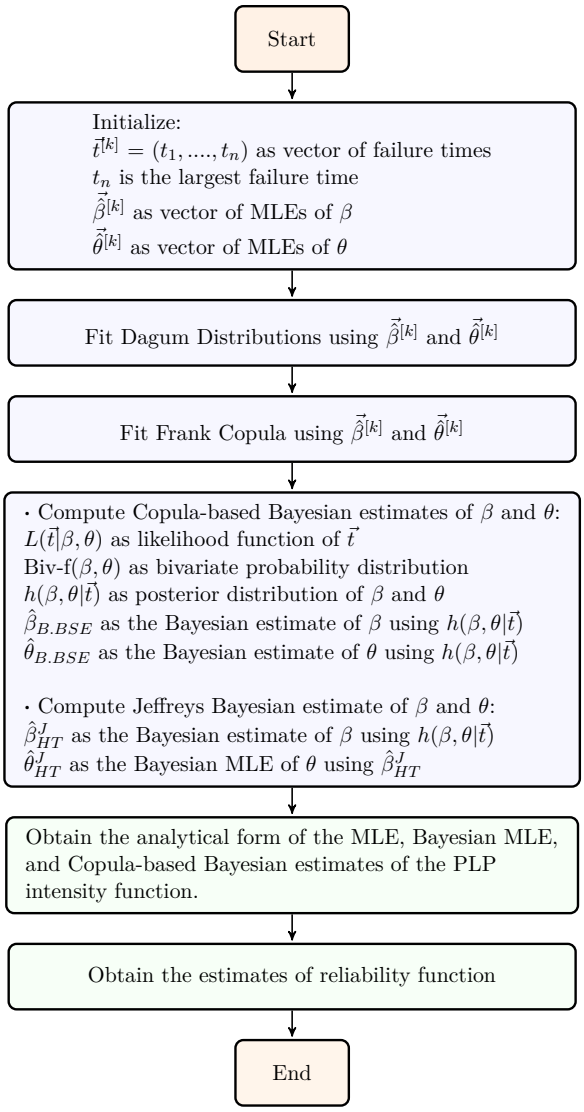


Figure 4.11.: Obtaining the MLE, Bayesian MLE, and Copula-based Bayesian estimates estimates of the conditional reliability function of the PLP using real data

Plotting the intensity function against time indicates the improvement status of the software reliability over time. A graphical display of $\hat{V}_{MLE}(t)$, $\hat{V}^{J}_{HT}(t)$, and $\hat{V}_{B.BSE}(t)$ is given below.

Based on Figure 4.12, the Copula-based Bayesian estimate of the intensity function shows a slower improvement in the software package reliability over time than would be estimated by MLE and Bayesian MLE. We recommend using the Copula-based Bayesian estimate of the intensity function since it provided better accuracy than the MLE and Bayesian MLE of the intensity function as a result of the simulation study in the previous section.

To obtain a Copula-based Bayesian estimate for the reliability function under the squared-error

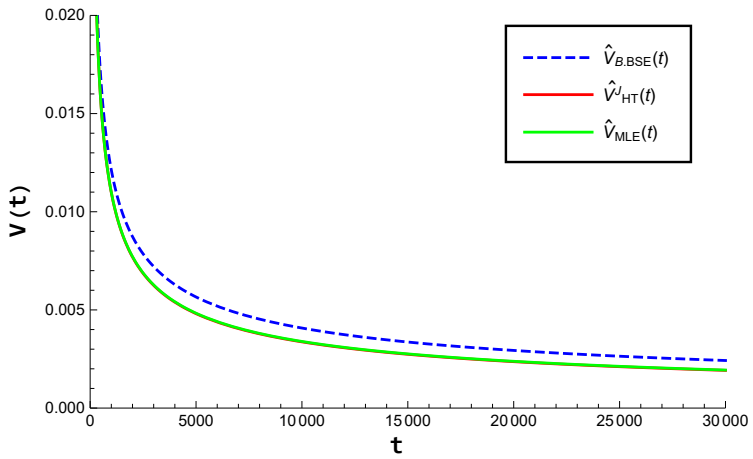


Figure 4.12.: MLE, Bayesian MLE, and Copula-based Bayesian estimates of the intensity function for the given real data in Table 4.1

loss function, we use the Copula-based Bayesian estimate for the intensity function. The analytical form for the corresponding Copula-based Bayesian reliability estimate, based on the real data, is given by:

$$\hat{R}_{B.BSE}(t_i|t_1,..,t_{i-1}) = \exp\left\{-0.319025\int_{t_{i-1}}^{t_i} x^{-0.473}dx\right\},\ t_i > t_{i-1} > 0. \tag{4.3.2.4}$$

The Copula-based Bayesian estimate of the conditional reliability of the software given that the last two failure times were $t_{39} = 3181$ and $t_{40} = 3256.3$ is approximately 59% whereas the MLE and Bayesian MLE of the conditional reliability are approximately 63%. Based on the simulation results, again, we recommend using the Copula-based Bayesian estimate of the conditional reliability of the PLP.

Thus, we demonstrated not only the applicability of the Copula-Based Bayesian analysis to the PLP, but also, using real data, the superiority of its performance compared to the MLE and Bayesian MLE of PLP intensity function, assuming the bivariate probability distribution of β and θ (Biv-f(β,θ)) is the prior knowledge of both parameters.

4.4 Contributions

The Power Law Process (PLP) is a commonly used model to analyze rarely occurring time-dependent events. In software reliability, the failure times of a software under development are

analyzed to assess the software reliability as a time-dependent function. The estimates of the PLP's parameters, β and θ, do not have closed forms [25, 26]. Therefore, there have been efforts to better estimate them, using frequency and Bayesian approaches [19, 97, 40, 42, 65, 5]. Based on the approximate maximum likelihood estimates (MLEs) of the parameters, β is a function of the largest failure time, and θ is a function of β. The dependancy among MLEs of β and θ has not been considered while developing Bayesian estimates of the PLP intensity function. We studied this dependency structure among the parameters, and based on that we developed a Copula- based Bayesian estimate of the PLP intensity function.

The first research result was the development of a bivariate probability distribution of β and θ using Copula theory. The developed probability density distribution was found to be a valid density function. It assists in capturing the bivariate probabilistic behavior of both parameters, and therefore helps to build better Bayesian estimates by choosing appropriate prior knowledge.

The second result was the development of an analytical Copula-based Bayesian analytical forms of the parameters β and θ, using the developed bivariate probability distribution as a bivariate prior of β and θ. We employed an extensive simulation study to compare the MLE, Jeffreys Bayesian where only β is considered an unknown random quantity, and Copula-based Bayesian estimates of both parameters. The Copula-based Bayesian estimates had excellent performance compared to the MLEs and Jefferys Bayesian estimates based on the mean square error criteria. Even for different sample sizes, similar results were achieved. We then compared the MLE, Bayesian MLE, and Copula-based Bayesian estimates of the PLP's intensity function, which resulted, as expected, in the Copula-based Bayesian intensity function outperforming the MLE and Bayesian MLE of the intensity function. The Bayesian conditional reliability function was then obtained.

Thus, based on this aspect of our analysis, we can conclude that the Copula-based Bayesian analysis approach, where both β and θ parameters are considered unknown random quantities, under the squared-error loss function, is superior to the maximum likelihood and Jefferys Bayesian, where only β parameter is considered an unknown random quantity, under the Higgins-Tsokos loss function, approaches in estimating the reliability function of the Power Law Process. The results of this study have the potential to contribute not only to the reliability analysis field but also to other fields that employ the Power Law Process.

Chapter 5
Probabilistic Comparisons of Commonly Used Computer Operating Systems' Vulnerabilities

In this chapter, we demonstrate the difficulty of performing parametric analysis of the computer operating systems vulnerability scores and rely on non-parametric method to proceed with the probabilistic analysis. Thus, we employ the kernel density approach to characterize the probabilistic behavior of computer operating systems vulnerability scores. We use the published vulnerability data in Common Vulnerabilities and Exposures and National Vulnerability Database databases.

The chapter is organized as follows, Section 2 describes the background: The National Institute of Standards and Technology (NIST) initiatives and related research. Section 3 presents the data and methodology. Section 4 presents results and discussion. Section 5 describes the contributions.

5.1 Introduction

Working backwards from end-users all the way to the software developers initially involved in the design and manufacture of software systems used in personal and business aspects of daily life, all parties are profoundly concerned with the information security and reliability components of software they build, use and trust. Recognizing that many threat sources exist and that they pose significant risks to demographic and financial data of all types, developers strive to include robust security and reliability features in the development process.

Software security breaches mostly happen when hackers identify and exploit known and unknown software vulnerabilities. The National Institute of Standards and Technology (NIST), under the U.S. Department of Commerce, defines system vulnerability as, "**Weakness in an information system, system security procedures, internal controls, or implementation that could be exploited or triggered by a threat source**". Clearly, when an operating system (OS) has several vulnerabilities or one severe vulnerability, it makes it less reliable and vice versa.

Efforts have been made to evaluate the vulnerabilities of systems, such as computer operating systems (OSes), to help establish and maintain security policies, one of which is developing a scoring

system to classify each system vulnerability. While scoring systems have been developed by vendors, the Common Vulnerability Scoring System (CVSS) is the commonly used scoring system. CVSS is under the custodial care of the **Forum of Incident Response and Security Teams** (**FIRST**) [62]. CVSS provides software developers and end-users with a framework to disseminate publicly published software vulnerabilities and thereby assist them in prioritizing their efforts related to the most significant impacts [87]. There are three versions of the CVSS, namely versions 1.0, 2.0, and 3.0., referred as CVSS v1, v2, and v3.

Starting in the early 2000s, researchers began to quantitatively analyze software vulnerabilities to better understand and predict the impact of newly discovered vulnerabilities. Prior to that, quantitative methods applied to software related data mostly focused on software reliability such as predicting reliability based on software failure data [26, 100, 39, 21, 9, 40, 97, 82, 83, 84]. Alhazmi and Malaiya [6] presented a figure to show the cumulative vulnerabilities of some Windows and UNIX computer operating systems. The following year, they proposed a time dependent model, similar to the reliability growth model, to monitor cumulative vulnerabilities over time. Jones, [52], proposed a method to identify and analyze software vulnerabilities that are publicly disclosed but not fixed. He used the publicly available Windows 2000 data for the period of January 1, 2001 to December 31, 2001, where his method depended on the daily vulnerability exposure (DVE). Ahmed, Al-Shaer, and Khan [1] proposed a framework to quantitatively evaluate network security by identifying security risk factors, predicting possible vulnerabilities, estimating their associated impact, and evaluating the security policy used in a network.

Vulnerability databases like Common Vulnerabilities and Exposures (CVE) and National Vulnerability Database (NVD) provide numerical and textual details for each published OS vulnerability. We refer to those details for each vulnerability as a vulnerability description. Modeling the textual details of the software vulnerabilities is an interest for some authors [70, 101, 92]. They showed the text description is a reliable source for modeling software vulnerability and proposed models to predict the severity of the vulnerabilities.

This study focuses on the published vulnerability scores of computer operating systems vulnerabilities. Therefore, the primary objectives of this study are to:

1. Develop the non-parametric analytical density estimate of the vulnerability scores of:

    1.1. Microsoft, Apple, and Linux computer operating systems combined.

    1.2. Microsoft, Apple, and Linux computer operating systems individually.

2. Obtain and compare probabilities and expected values of computer operating systems **low (0-3.9), medium (4-6.9),** and **high (7-10)** vulnerability scores.

We achieve the research objectives by identifying the probabilistic behavior of the computer operating systems vulnerability scores (CVSSs), and then calculate the subject probabilities. Throughout this research, the published vulnerability scores of computer operating systems vulnerabilities in Common Vulnerabilities and Exposures and National Vulnerability Database databases from 1999 to 2019 will be used.

5.2 Background

5.2.1 NIST initiatives: CVE, NVD, and CVSS

The National Institute of Standards and Technology (NIST), under the U.S. Department of Commerce, has several initiatives such as CVE, NVD, and CVSS.

The CVE was established in 1999, and can be seen as a vulnerabilities' dictionary, where each vulnerability is assigned a unique identification (ID) by the CVE Numbering Authorities (CNAs). CNAs are international organizations that are authorized to communicate and assign new vulnerabilities that are related to their products with a unique ID. The vulnerabilities' IDs are therefore made publicly available. The standards followed for assigning vulnerabilities' IDs and the process of participating as a member of the CNAs can be found on the CVE web-page, [24]. CVE is currently maintained and managed by MITRE, [23]. The vulnerability ID consists of three parts separated by dashes, namely, the abbreviation CVE, year, and four digis serial number. For example, CVE-2019-0772 is assigned to one of the Windows 10 vulnerabilities that was published on April 8, 2019.

The NVD [72] is a publicly and freely available comprehensive database of products' vulnerabilities. It is a product of the Computer Security Division, NIST, and sponsored by the National Cyber Security Division, Department of Homeland Security. NVD was established in 2000 under the name Internet –Categorization of Attacks Toolkit and abbreviation (ICAT). The NVD staff analyze the published CVEs, and collect the associated description, references, and any other related public data.

The CVSS is a framework to evaluate software vulnerabilities was developed by CVSS Special Interest Group (CVSS-SIG) under the FIRST, [33]. The CVSS has three versions [32]; CVSS v1, v2, and v3, and they were published in 2004, 2007, and 2015, respectively. Technology community

including organizations and individual specialists are using CVSS to assess software vulnerabilities and prioritize their remediation processes, [78]. The vulnerability score (VS) based on CVSS is a number between 0 and 10, where the vulnerability's severity increases as the number approaches to 10. There are three metrics used to compute the VS; namely, base, temporal, and environmental metrics. The VS using the base metrics is the emphasis of this research. It has 6 metrics; congregated into exploitability metrics (**Access Vector** (**AV**), **Access Complexity** (**AC**), and **Authentication** (**AU**)) and **impact metrics** (**Availability Impact** (**AI**), **Confidentiality Impact** (**CI**), and **Integrity Impact** (**II**)).

5.2.2 Related Research

To the best of our knowledge, the vulnerability scores of computer operating systems vulnerabilities have not been utilized to identify, if possible, the probability distribution function that characterize their probabilistic behavior.

5.3 Data and Methodology

5.3.1 CVSS v2

According to FIRST, the CVSS-SIG and others have identified major issues with the CVSS v1. One of the main issue is that it did not go through mass peer review across multiple organizations and industries, which led the team to develop and publish the CVSS v2, [99].

The VS under the CVSS v2 is the scope of this research because it is easily accessible to all of the disclosed vulnerabilities via CVEdetails, [104]. CVEdetails was developed by Serkan Özkan. It automatically collects vulnerability data from various sources as the NVD is the main source, and others as additional sources such as CWE, Exploit Database [88], Microsoft Security Bulletin, Metasploit, among others, [73]. The Beautiful Soup and Scrapy, [81, 43], libraries in Python were used to scrap the vulnerabilities of the Microsoft, Apple, and Linux OSes from 1999 to 2019.

The schematic diagram in Figure 1.1 shows the computation of the vulnerability base score based on CVSS v2.

5.3.2 Computer operating systems: Microsoft, Apple, and Linux

According to the NetMarketShare, [89], as of March 2019, the operating system market shares are 87.45%, 9.73%, and 2.16% for Microsoft, Apple, and Linux OSes, respectively. These represent

99.34% of the existing OSes. Therefore, they are selected for this research. The following table shows the versions of the Microsoft, Apple, and Linux OSes that are included in this research.

Table 5.1: OSes considered in this research and their family.

OS	Vendor	Family
Mac Os X	Apple	**Apple**
Mac Os X Server	Apple	**Apple**
Mac Os	Apple	**Apple**
Debian Linux	Debian	**Linux**
Linux Kernel	Linux	**Linux**
Ubuntu Linux	Canonical	**Linux**
Opensuse	Opensuse	**Linux**
Enterprise Linux	Redhat	**Linux**
Solaris	SUN	**Linux**
Fedora	Fedoraproject	**Linux**
Kernel	Linux	**Linux**
Linux Kernel-rt	Linux	**Linux**
Windows 7	Microsoft	**Microsoft**
Windows Vista	Microsoft	**Microsoft**
Windows 10	Microsoft	**Microsoft**
Windows Xp	Microsoft	**Microsoft**
Windows 8.1	Microsoft	**Microsoft**
Windows 2000	Microsoft	**Microsoft**
Windows	Microsoft	**Microsoft**
Windows 8	Microsoft	**Microsoft**
Windows 98	Microsoft	**Microsoft**
Windows 95	Microsoft	**Microsoft**

Table 5.1 shows that Microsoft has the largest number of OSes compares with the others. Linux family has various OSes that were developed by different developers as Linux is an open source tool that allows anyone to use it for operating system development. On the other hand, Apple has the smallest number of OSes due to its latest existence compared to others.

5.3.3 Kernel Density Estimation

There is no substitute for the powerful parametric approach, except when we have difficult/complex data structure where the assumption regarding the underline probability distribution is questionable or the subject distribution is not well-defined. In such cases, the use of a non-parametric approach like the kernel density estimation could be more appropriate. The non-parametric kernel estimation has been used in reliability estimation of a system failure times, among several other area of studies, [64].

Let v_1,v_2, ..., v_i represents independent and identical distributed random variables having a common probability density function $f(t)$, (PDF). The kernel density estimation of $f(t)$ is given by:

$$\hat{f}_n(v) = \frac{1}{nh}\sum_{i=1}^{n} K\Big(\frac{v - v_i}{h}\Big), \tag{5.3.3.1}$$

where $K(u)$ is the kernel function and h is a positive number called the bandwidth. The kernel function is required to meet the following conditions:

- $\int_{-\infty}^{+\infty}(K(u))du$ =1,
- $\int_{-\infty}^{+\infty} u \cdot (K(u))du$ =0,
  and
- $\int_{-\infty}^{+\infty} u^2 \cdot (K(u))du \geq 0$.

The following Table 5.2 shows the most commonly used kernel functions.

Table 5.2: Most commonly used kernels

Kernel functions $(K(u))$	Expression
Gaussian	$\frac{1}{\sqrt{2}} \exp\left(\frac{-u^2}{2}\right) I_R$
Epanechnikov	$\frac{3}{4}(1-u^2) I_{\{\mid u \mid \leq 1\}}$
Uniform	$\frac{1}{2} I_{\{\mid u \mid \leq 1\}}$
Triangular	$(1- \mid u \mid) I_{\{\mid u \mid \leq 1\}}$
Triweight	$\frac{35}{32}(1-u^2)^3 I_{\{\mid u \mid \leq 1\}}$
Biweight	$\frac{15}{16}(1-u^2)^2 I_{\{\mid u \mid \leq 1\}}$
Tricube	$\frac{70}{81}(1- \mid u \mid^3)^3 I_{\{\mid u \mid \leq 1\}}$
Cosine	$\frac{\pi}{4} \cos\left(\frac{\pi}{2}u\right)$

The kernel density estimate of the cumulative density function for a given set of data with a common PDF $f(t)$ is defined by:

$$\hat{F}_n(v) = \frac{1}{nh}\sum_{i=1}^{n} \int_{-\infty}^{x} K\Big(\frac{x - v_i}{h}\Big)dx. \tag{5.3.3.2}$$

It is clear that kernel density estimation of the PDF depends on the sample size, bandwidth, and the choice of the kernel function, $(K(u))$. We shall apply this method in estimating the PDF of the computer operating systems vulnerability scores.

5.4 Results and Discussions

5.4.1 Kernel Density Estimation of computer operating systems Vulnerability Scores

The objective in this subsection is to identify a probability distribution function of the computer operating systems vulnerability scores. Figure 5.1 below shows the histogram of all published OSes vulnerability scores from 1999 to 2019, where they range from 0 to 10.

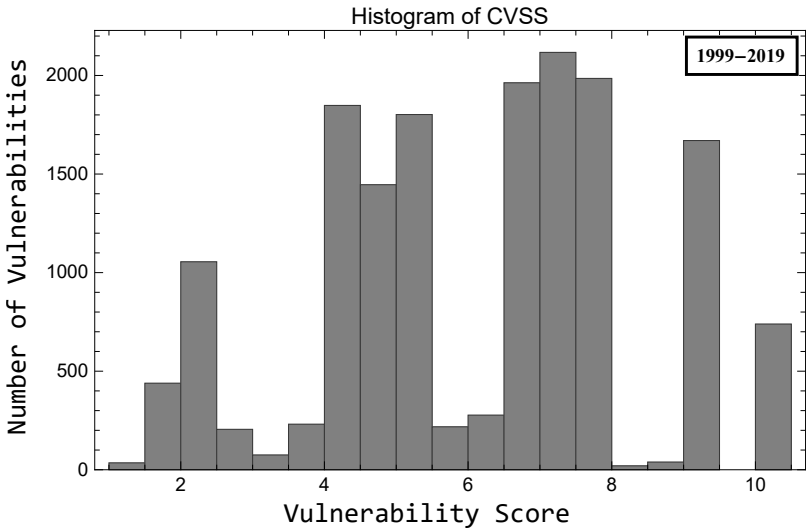


Figure 5.1.: Histogram of vulnerability scores collected from 1999 to 2019 for the Microsoft, Apple, and Linux OSes.

The histogram does not indicate that the vulnerability scores follow a Normal or any other well-defined probability distributions. Goodness of fit (GOF) tests were carried out to numerically test the assumption that the vulnerability scores follows a parametric PDF. Namely, Shapiro–Wilk and D'Agostino's K^2 GOF tests.

The numerical GOF tests confirm that the vulnerability scores do not follow any parametric probability distribution. Although, we have applied multiple transformation methods, with extensive search of the approximate underling distribution of the computer operating systems CVSSs, we were not able to find a well-defined probability distribution.

Therefore, the kernel density estimation approach was employed. It depends on the sample size, bandwidth, and the choice of the kernel function $(K(u))$. In this study, the optimal bandwidth

(h^*) and kernel function were chosen so as to minimize the asymptotic mean integrated squared error (AMISE). The simplified form of the AMISE is reduced to:

$$AMISE\left(\hat{f}(v)\right) = \frac{C(K)}{n \cdot h} + (\frac{1}{4} \cdot h^4 \cdot k_2^2 \cdot R\left(f^{(2)}(v)\right)), \tag{5.4.1.1}$$

where:

- C(K)= $\int (K(u))^2 du$.
- n: sample size.
- h: bandwidth.
- $k_2 = \int_{-\infty}^{+\infty} u^2 \cdot K(u) du$.
- $f^{(2)}(v)$ is the second derivative of $f(t)$.
- $R(f^{(2)}(v))= \int (f^{(2)}(v))^2 dv$.
- $h^*=\left[\frac{C(K)}{k_2^2 \cdot R\left(f^{(2)}(\beta)\right)}\right]^{1/5} \cdot n^{-1/5}$ (Silverman [90]).

AMISE depends on four entities, namely kernel function, bandwidth, sample size, and the target density f(v). We can control the bandwidth and the kernel function. By fixing the kernel function and using the optimal bandwidth (h^*), we obtain the optimal AMISE, [64]:

$$AMISE_{optima} = \frac{5}{4} \cdot (\sqrt{k_2} \cdot C(K))^{\frac{4}{5}} \cdot C(f^{(2)})^{\frac{1}{5}} \cdot n^{\frac{-4}{5}}. \tag{5.4.1.2}$$

The optimal kernel function is then chosen so that $AMISE_{optimal}$ is the minimum. Particularly, by minimizing $(\sqrt{k_2} \cdot C(K))$ without knowing the target PDF $f(v)$. The Epanechnikov kernel function has the minimum value of $AMISE_{optima}$ among the other kernel functions given in Table 5.2, which is given by:

$$K_{Epan.}(u) = \frac{3}{4}(1 - u^2), \qquad (\mid u \mid \leq 1). \tag{5.4.1.3}$$

Therefore, the kernel density estimates of the probability density function (PDF) and cumulative density function (CDF) of the computer operating systems vulnerability scores, $\hat{f}_n(v)$ and $\hat{F}_n(v)$, are given, respectively, by:

$$\hat{f}_n(v) = \frac{1}{(16164*0.278)} \sum_{i=1}^{16164} K_{Epan.}\left(\frac{v-v_i}{0.278}\right),$$
and
$$\hat{F}_n(v) = \frac{1}{(16164*0.278)} \sum_{i=1}^{16164} \int_{-\infty}^{x} K_{Epan.}\left(\frac{x-v_i}{0.278}\right)dx. \tag{5.4.1.4}$$

Figure 5.2 below shows the histogram of vulnerability scores (CVSSs) of the Microsoft, Apple, and Linux OSes combined. There are 16,164 vulnerability scores recorded from 1999 to March 15, 2019.

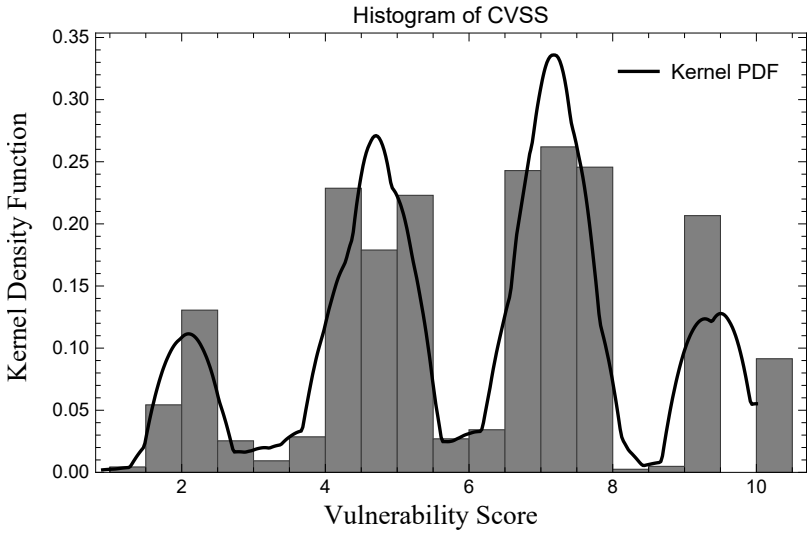


Figure 5.2.: Histogram of the computer operating systems vulnerability scores, along with the kernel density curve.

Figure 5.2 shows that the kernel density estimation captures the multimodality of of the computer operating systems vulnerability scores fairly well. By identifying the approximate probability distribution function of the computer operating systems vulnerability scores, one can calculate the probability that for a randomly selected operating system, will fall in **Low (0-3.9)**, **Medium (4-6.9)**, or **High (7-10)** vulnerability level, that will be useful for IT managers. We refer to those probabilities as $\hat{P}_{low}$, $\hat{P}_{medium}$, and $\hat{P}_{high}$, respectively. Using the form (5.3.3.1), the probabilities are given by:

$$\hat{P}_{low}(x) = \int_0^{3.99} \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x-v_i}{h}\right) dx,$$

$$\hat{P}_{medium}(x) = \int_4^{6.99} \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x-v_i}{h}\right) dx, \tag{5.4.1.5}$$

and

$$\hat{P}_{high}(x) = \int_7^{10} \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x-v_i}{h}\right) dx\,.$$

The following Figure 5.3 shows the histogram of the computer operating systems vulnerability scores, along with the curve of the kernel density estimation of the probability density function colored based on the vulnerability levels.

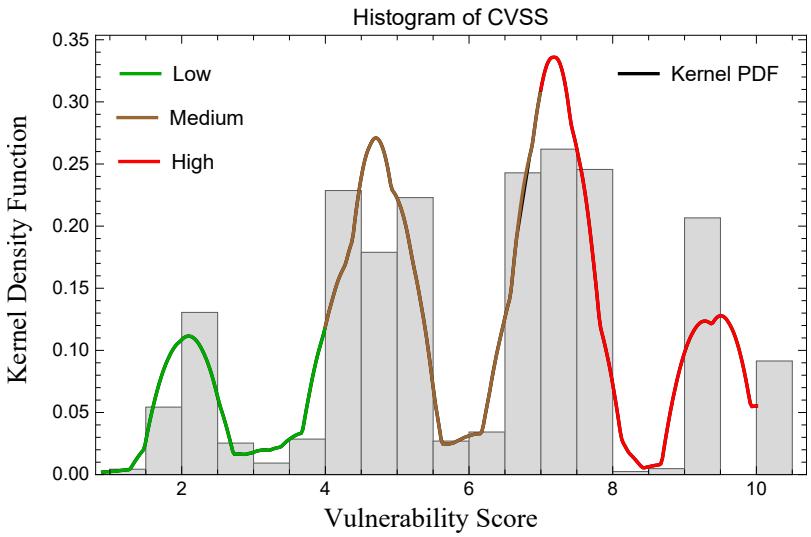


Figure 5.3.: Histogram of the computer operating systems vulnerability scores, along with the kernel density curve colored based on the vulnerability levels.

Figure 5.4 displays the cumulative density function of the computer operating systems vulnerability scores. We notice that the chance of a selected computer operating systems score falls into low level of vulnerability is much smaller than the given score falls into medium or high levels.

We proceed to calculate the subject probabilities using the analytical forms in equation (5.4.1.5). The results are presented by Table 5.3 below.

There is a higher chance for a randomly picked operating system vulnerability score to fall in the medium vulnerability level. Followed by falling into the high vulnerability level with a probability estimate of 0.40. With a probability estimate of 0.15, the randomly selected operating system

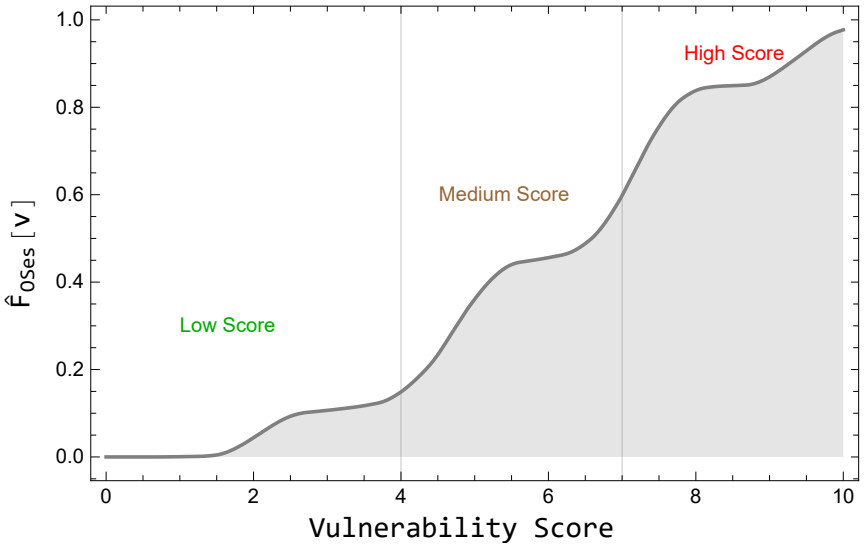


Figure 5.4.: Cumulative density function of the computer operating systems Vulnerability Scores (CVSSs).

Table 5.3: Probabilities that a randomly selected operating system vulnerability score, will fall in low (0-3.9), medium (4-6.9), and high (7-10) vulnerability level.

Probability	Probability
$\hat{P}_{low}$	0.15
$\hat{P}_{medium}$	0.45
$\hat{P}_{high}$	0.40

falls in the low vulnerability level. The expected and median values of the vulnerability scores are defined by:

$$\begin{aligned}&\text{Expected value} = E(v) = \int_0^{10} v \cdot \hat{f}_n(v)\, dv = \int_0^{10} v \cdot \tfrac{1}{nh} \sum_{i=1}^{n} K\left(\tfrac{v-v_i}{h}\right) dv = 6.1,\\ &\text{and}\\ &\text{Median value} = m_v, \text{ such that } \hat{P}(v < m_v) = \int_0^{m_v} \tfrac{1}{nh} \sum_{i=1}^{n} K\left(\tfrac{v-v_i}{h}\right) dv = \tfrac{1}{2}, = 6.8\,.\end{aligned} \tag{5.4.1.6}$$

The expected and median values of the computer operating systems vulnerabilities scores are 6.1 and 6.8, respectively. The expected and median values fall into the medium level of vulnerability severity scale provided by NVD.

Since the net market share of the Microsoft computer operating systems is very large comparing

to Linux and Apple computer operating systems, we shell obtain the kernel density estimation of Microsoft, Linux, and Apple computer operating systems individually. And proceed to calculate the probabilities in same previous manner.

5.4.1.1 Microsoft computer operating systems Vulnerability Scores

The Microsoft computer operating systems vulnerability scores were used to obtain kernel density estimates of the PDF ($f(v)$) and CDF ($F(v)$) of the given scores. They are given by:

$$\hat{f}_n(v) = \frac{1}{(5107*0.393)} \sum_{i=1}^{5107} K_{Epan.}\left(\frac{v-v_i}{0.393}\right),$$
$$\text{and} \tag{5.4.1.1}$$
$$\hat{F}_n(v) = \frac{1}{(5107*0.393)} \sum_{i=1}^{5107} \int_{-\infty}^{x} K_{Epan.}\left(\frac{x-v_i}{0.393}\right) dx.$$

Figure 5.5 shows the PDF curve of the Microsoft computer operating systems vulnerability scores.

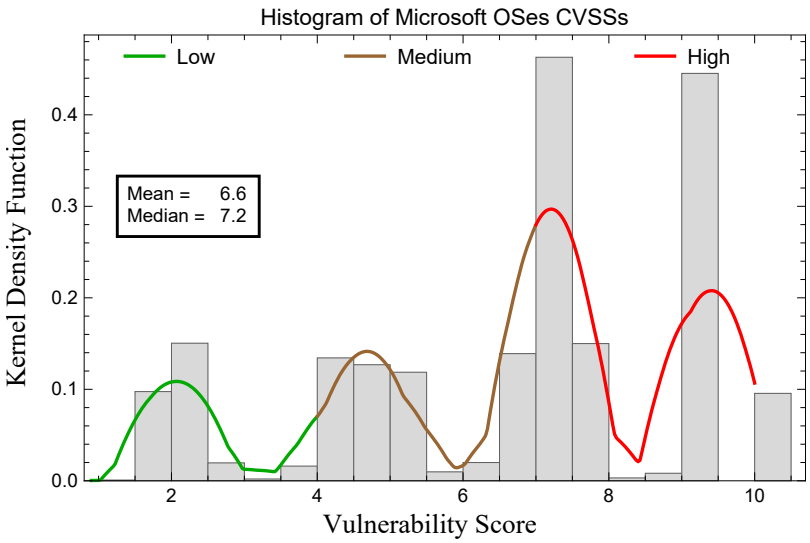


Figure 5.5.: Histogram of the Microsoft computer operating systems CVSSs, along with the kernel density curve colored based on the vulnerability levels.

The expected and median values, defined in equations (5.4.1.6), of the Microsoft computer operating systems vulnerability scores are 6.6 and 7.2, respectively. The expected value falls into the medium level of the vulnerability severity scale provided by NVD, whereas the median value falls into the high severity level, which both are higher than the expected and median values of all computer operating systems vulnerability scores.

The following Figure 5.6 shows the CDF of the Microsoft computer operating systems vulnerability scores.

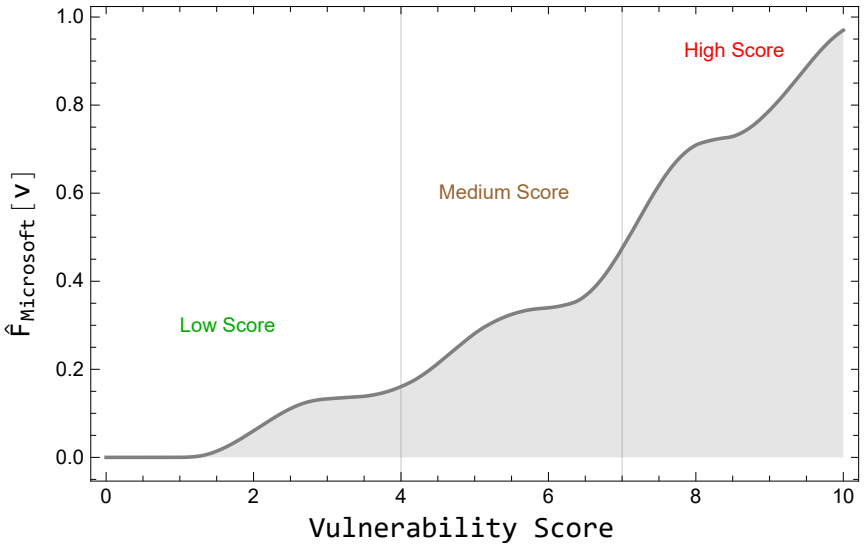


Figure 5.6.: Cumulative density function of the Microsoft computer operating systems vulnerability scores (CVSSs).

We notice from Figure 5.6 that the area under curve for the high vulnerability score level (6.9-10) is larger than the other areas under the curve. This indicates that if we selected randomly a Microsoft operating system, there is a higher probability it will fall in high level of severity. Table 5.4 presents the probability estimates if we randomly selected a Microsoft operating system vulnerability score, and it will fall into low, medium, and high level of severity, respectively.

Table 5.4: Probabilities that a randomly selected Microsoft operating system vulnerability score, will fall in Low (0-3.9), Medium (4-6.9), and High (7-10) vulnerability level.

Probability	Value
$\hat{P}_{low}$	0.16
$\hat{P}_{medium}$	0.31
$\hat{P}_{high}$	0.53

As expected, the results in Table 5.4 confirms that a Microsoft operating system vulnerability score is more likely to fall into high level of vulnerability severity if selected randomly.

5.4.1.2 Linux computer operating systems Vulnerability Scores

The Linux computer operating systems vulnerability scores were used to obtain kernel density estimates of the PDF ($f(v)$) and CDF ($F(v)$) of the given scores. They are given by:

$$\hat{f}_n(v) = \frac{1}{(8304*0.288)} \sum_{i=1}^{8304} K_{Epan.}\left(\frac{v-v_i}{0.288}\right),$$
$$\text{and} \tag{5.4.1.1}$$
$$\hat{F}_n(v) = \frac{1}{(8304*0.288)} \sum_{i=1}^{8304} \int_{-\infty}^{x} K_{Epan.}\left(\frac{x-v_i}{0.288}\right) dx.$$

Figure 5.7 shows the PDF curve of the Linux computer operating systems vulnerability scores.

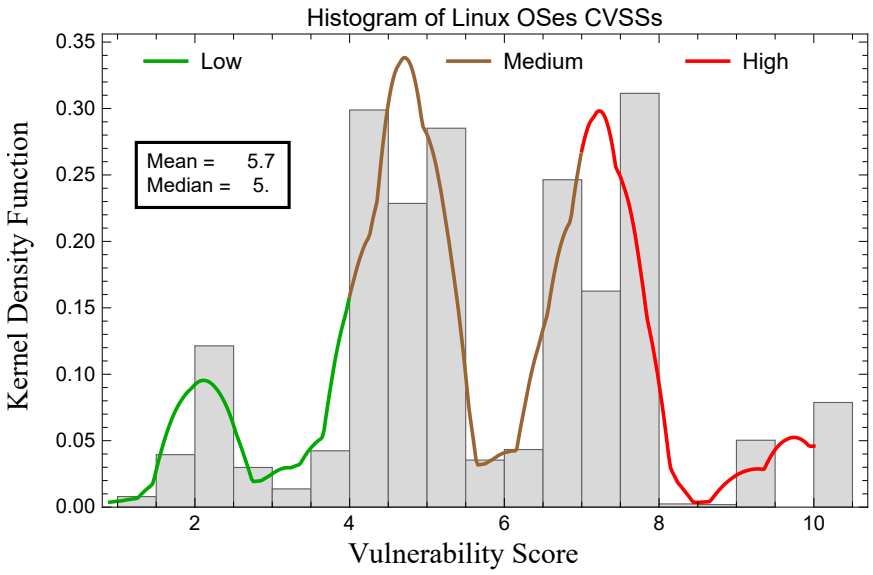


Figure 5.7.: Histogram of the Linux computer operating systems CVSSs, along with the kernel density curve colored based on the vulnerability levels.

The expected and median values, defined in equations (5.4.1.6), of the Linux computer operating systems vulnerability scores are 5.7 and 5, respectively. The expected and median values fall into the medium level of vulnerability severity scale provided by NVD, which are both are less than the expected and median values of all computer operating systems vulnerability scores.

Figure 5.8 shows the CDF of the Linux computer operating systems vulnerability scores.

We notice from Figure 5.8 that the area under curve for the medium vulnerability score level (4-6.9) is larger than the other areas under the curve. This indicates that if we selected randomly a Linux operating system, there is a higher probability it will fall in medium level of severity.

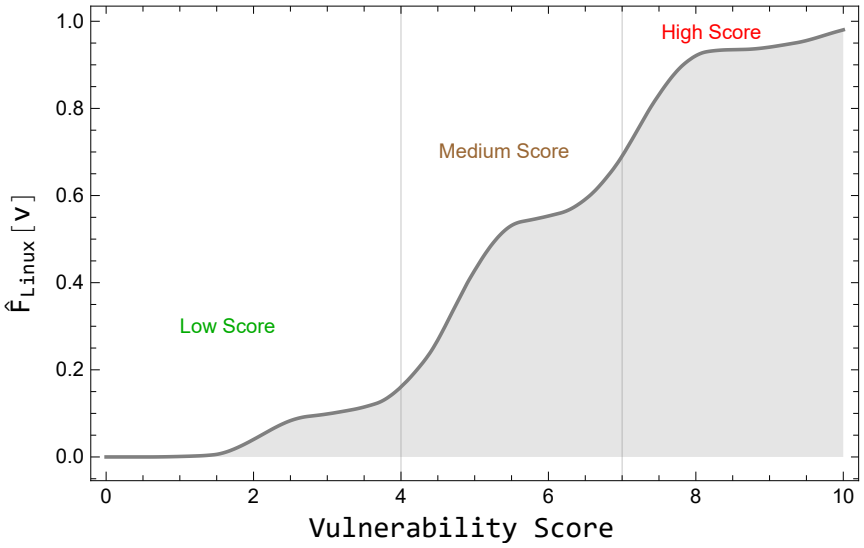


Figure 5.8.: Cumulative density function of the Linux computer operating systems vulnerability scores (CVSSs).

Table 5.5 presents the probability estimates if we randomly selected a Linux operating system vulnerability score, and it will fall into low, medium, and high level of severity, respectively.

Table 5.5: Probabilities that a randomly selected Linux operating system vulnerability score, will fall in Low (0-3.9), Medium (4-6.9), and High (7-10) vulnerability level.

Probability	Value
$\hat{P}_{low}$	0.16
$\hat{P}_{medium}$	0.53
$\hat{P}_{high}$	0.31

As expected, the results in Table 5.5 confirms that a Linux operating system vulnerability score is more likely to fall into medium level of vulnerability severity when selected randomly.

5.4.1.3 Apple computer operating systems Vulnerability Scores

The Apple computer operating systems vulnerability scores were used to obtain kernel density estimates of the PDF ($f(v)$) and CDF ($F(v)$) of the given scores. They are given by:

$$\hat{f}_n(v) = \frac{1}{(2753*0.341)} \sum_{i=1}^{2753} K_{Epan.}\left(\frac{v-v_i}{0.341}\right),$$
and
$$\hat{F}_n(v) = \frac{1}{(2753*0.341)} \sum_{i=1}^{2753} \int_{-\infty}^{x} K_{Epan.}\left(\frac{x-v_i}{0.341}\right)dx. \tag{5.4.1.1}$$

Figure 5.9 shows the PDF curve of the Apple computer operating systems vulnerability scores.

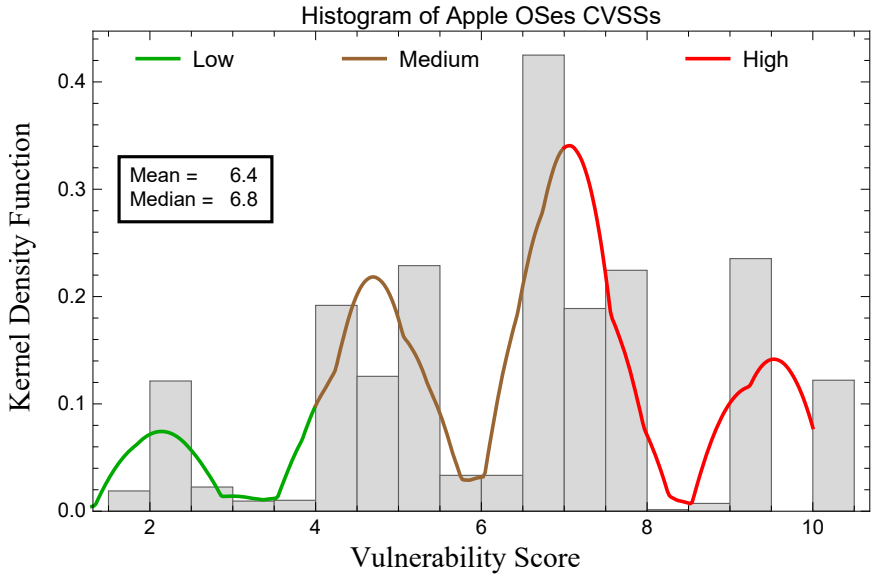


Figure 5.9.: Histogram of the Apple computer operating systems CVSSs, along with the kernel density curve colored based on the vulnerability levels.

The expected and median values, defined in equations (5.4.1.6), of the Apple computer operating systems vulnerability scores are 6.4 and 6.8, respectively. The expected and median values fall into the medium level of vulnerability severity scale provided by NVD. The expected value is higher than the expected value of all computer operating systems vulnerability scores.

The following Figure 5.10 shows the CDF of the Apple computer operating systems vulnerability scores.

We notice from Figure 5.10 that the area under curve for the low vulnerability score level (0-3.99) is less than the other areas under the curve. This indicates that if we selected randomly an Apple operating system vulnerability score, there is less probability it will fall in low level of severity. Table 5.6 presents the probability estimates if we randomly selected an Apple operating system vulnerability score, and it will fall into low, medium, and high level of severity, respectively.

The results in Table 5.6 shows that an Apple computer operating systems vulnerability score is

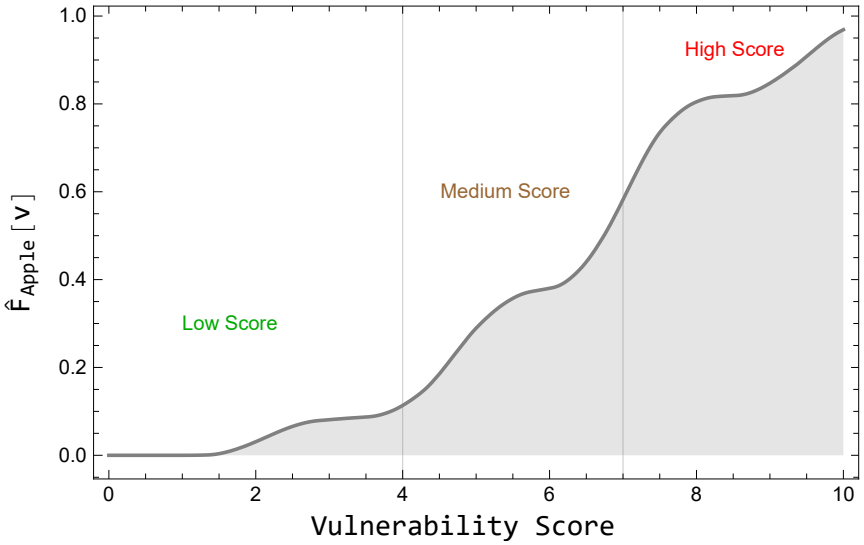


Figure 5.10.: Cumulative density function of the Apple computer operating systems vulnerability scores (CVSSs).

Table 5.6: Probabilities that a randomly selected Apple operating system vulnerability score, will fall in Low (0-3.9), Medium (4-6.9), and High (7-10) vulnerability level.

Probability	Value
$\hat{P}_{low}$	0.11
$\hat{P}_{medium}$	0.47
$\hat{P}_{high}$	0.42

more likely to fall into medium level of vulnerability severity when selected randomly.

Thus far, we obtained kernel estimate of the probability and cumulative density functions of all computer operating systems vulnerability scores. Considering the net market share for Microsoft, Linux, and Apple computer operating systems, we further obtained separate kernel estimates of the probability and cumulative density functions for Microsoft, Linux, and Apple computer operating systems vulnerability scores. Microsoft computer operating systems vulnerability scores has highest expected and median values. Followed by Apple and Linux computer operating systems vulnerability scores. Table 5.7 summarizes the probability estimates if we randomly selected Microsoft, Linux, or Apple operating system vulnerability score, and it will fall into low, medium, and high level of severity, respectively.

The results in Table 5.7 shows that an Microsoft computer operating systems vulnerability score is more likely to fall into High level of vulnerability severity when selected randomly. Followed by

Table 5.7: Summary of probabilities for a randomly selected operating system vulnerability score, will fall in Low (0-3.9), Medium (4-6.9), and High (7-10) vulnerability level.

computer operating systems	$\hat{P}_{low}$	$\hat{P}_{medium}$	$\hat{P}_{high}$
Microsoft	0.16	0.31	0.53
Linux	0.16	0.53	0.31
Apple	0.11	0.47	0.42
All	0.15	0.45	0.40

Apple and Linux computer operating systems. This suggests taking into consideration the type of computer operating systems in any further analysis of vulnerability scores.

5.5 Contributions

The security component of any operating systems is crucial since they store sensitive and confidential information. Computer operating systems (OSes) such as Microsoft, Apple, and Linux are the focus of this research. Specifically, the vulnerability scores of the OSes were collected from January 1999 to March 2019, and utilized to achieve the research objectives.

We used kernel density estimation, as a non-parametric approach, to identify the probability distribution function that characterizes probabilistic behavior of the computer operating systems vulnerability scores. It estimates the central tendency measures of the vulnerability scores and helps us better understand their behaviour probabilistically. Furthermore, we obtained kernel density estimates of the probability and cumulative density functions of Microsoft, Linux, and Apple computer operating systems vulnerability scores individually. It is noted that the vulnerability scores of Microsoft computer operating systems have higher expected and median values of vulnerabilities comparing to Linux and Apple computer operating systems. The probabilistic comparison of the Microsoft, Apple, and Linux computer operating systems identifies their strengths and weaknesses at the three severity levels of vulnerabilities.

The graphical figures of the kernel density estimates of the probability density functions and the cumulative distribution curves could help vendors and information technology managers increasing their understanding of the probabilistic behavior of their computer operating systems vulnerability scores. The resulting findings are important marketing tool in addition of given IT managers a process of decision making about modifying the subject computer operating system. This methodology and analysis could also be applicable to any software system.

Chapter 6

Effective Classification and Prediction Methods of Computer Operating Systems Vulnerabilities Subject to Risk Factors

In this chapter, we evaluate the effectiveness of common machine learning methods in classifying and predicting the computer operating systems vulnerability severity levels into **low (0-3.9), medium (4-6.9), or high (7-10) vulnerability severity level** and scores, respectively, subject to 13 risk factors that have never been used in this form of analysis. The risk factors are **Confidentiality Impact, Integrity Impact, Availability Impact, Access Complexity, Authentication, Gained Access, Family, Days, Year, NumAttacks, Frequency, Polarity, and Subjectivity**. We use the published vulnerability data in Common Vulnerabilities and Exposures and National Vulnerability Database databases. One article is accepted to be presented in the International Conference on Computer Applications & Information Security (ICCAIS'2020) conference [3] based on some findings of this Chapter.

The chapter is organized as follows, Section 2 describes the related research. Section 3 presents the data and methodology. Section 4 presents results and discussion. Section 5 describes the contributions.

6.1 Introduction

Information security is a major concern of not only software developers but also end-users. Software reliability and security are important components requiring evaluation during the development process, and monitoring for as long as the software is publicly available to users. Our high dependency on software packages for managing simple daily routines, such as storing our personal and financial information, require us to trust such software. Small and large businesses use software packages to operate and store their data as well.

Recent efforts utilized the Common Vulnerability Scoring System (CVSS) to develop probabilistic and statistical models to predict network security. Kaluarachchi, Tsokos, and Rajasooriya, [55], proposed a stochastic model, using a small schematic network system and structuring an attack

graph, that helps to measure the network security. The authors also proposed a statistical model to predict the expected path length and estimate, with a probability of approximately 1, the minimum number of steps to reach the target software in the network. Pokhrel and Tsokos, [76], proposed a stochastic predictive model, based on the Markovian process, to evaluate the risk to the entire network. Pokhrel, Rodrigo, and Tsokos, [75], proposed a time series based predictive model, using linear and non-linear approaches, to predict the number of vulnerabilities of a given operating system and thereby assist vendors in monitoring their OSes. Moreover, several authors [15, 31, 18, 30, 85] showed the usefulness and validity of machine learning techniques in the evaluation of software vulnerabilities, such as predicting the number of vulnerabilities and the time length until exploitation, if they are exploitable.

This study focuses on the numerical and textual details of computer operating systems vulnerabilities. Specifically, we evaluate the effectiveness of common machine learning methods in classifying and predicting the computer operating systems vulnerability severity levels (low (0-3.9), medium (4-6.9), and high (7-10) vulnerability severity level) and scores (range from 0 to 10), respectively, subject to 13 risk factors found in the National Vulnerability Datavase (NVD), namely **Confidentiality Impact, Integrity Impact, Availability Impact, Access Complexity, Authentication, Gained Access, Family, Days, Year, NumAttacks, Frequency, Polarity, and Subjectivity**. The impact of security breaches as a result of OSes vulnerability exploitation could be unpredictably large. For example, the exploitation of a Microsoft vulnerability (SMB MS17–010 in May 2017), forced several UK health-care providers to cancel doctors' appointments, including surgeries, and FedEx international shippers were also hit, among others [41]. This attack is known as **WannaCry**, and Microsoft published security updates for Windows XP, Windows 8, and Windows Server 2003 [38]. Therefore, the primary objectives of this study are to:

1. Develop a classification model to classify vulnerability severity level, subject to the 13 risk factors, of a given computer operating system vulnerability into low (0-3.9), medium (4-6.9), or high (7-10). Five methods are employed to select the best method, namely:

    1.1. Logistic Regression (LR).

    1.2. K nearest neighbors (kNN).

    1.3. Gaussian Naive Bayes (GNB).

    1.4. Random forest (RF).

    1.5. Adaptive Boosting (AdaBoost).

2. Develop a model, subject to the 13 risk factors, to classify whether a given computer operating system vulnerability will allow attackers to cause **Denial of Service** to the subject system.

3. Develop a prediction model to predict the vulnerability score of a given computer operating system vulnerability subject to the 13 risk factors for:

   3.1. Microsoft, Apple, and Linux computer operating systems combined.

   3.2. Microsoft, Apple, and Linux computer operating systems individually.

4. Develop a ranking process to rank the contribution of the 13 risk factors in predicting the computer operating systems vulnerability scores.

The first research objective is achieved by utilizing common machine learning methods to develop classification models to classify the vulnerability severity level of a given computer operating system vulnerability into low (0-3.9), medium (4-6.9), or high (7-10), subject to the 13 risk factors. The second research objective is achieved by utilizing a Random Forest classification method, subject to the 13 risk factors, to classify whether a given vulnerability would cause a **Denial of Service** to the computer operating system. The third research objective is achieved by utilizing a Random Forest regression method, subject to the 13 risk factors, to predict computer operating systems vulnerability scores. The fourth objective is achieved by utilizing a Random Forest regression method to develop the ranking process subject to the 13 risk factors. Throughout this research, we use the published vulnerability data of computer operating systems in Common Vulnerabilities and Exposures and National Vulnerability Database databases from 1999 to 2019.

6.2 Related Research

To the best of our knowledge, the 13 risk factors, namely **Confidentiality Impact, Integrity Impact, Availability Impact, Access Complexity, Authentication, Gained Access, Family, Days, Year, NumAttacks, Frequency, Polarity, and Subjectivity**, have not been used to develop analytical models to classify and predict the computer operating systems vulnerability severity levels into low (0-3.9), medium (4-6.9), or high (7-10) vulnerability severity level and scores, respectively. However, the following are some research studies that utilized NVD and/or Common Vulnerabilities and Exposures (CVE) databases and machine learning methods to develop analytical models for vulnerabilities' prediction and management purposes.

Bozorgi, Saul, Savage, and Voelker [15] used the Open Source Vulnerability Database (OSVDB) and the CVE database to find how likely a vulnerability could be exploited and how soon. They extracted a very large number of variables, many of which are binary variables derived using a bag-of-words model representing whether certain words exist in each text field (i.e. name of attack types). Their data showed that 73% of the vulnerabilities were classified as "exploited" which contradicts the data collected in [7, 68, 30]. By using linear support vector machines (SVMs), they were able to reach a false positive rate of 12.5%, with approximately 90% accuracy.

Zhang, Caragea, and Ou [103] carried out some machine learning methods, among other methods, using NVD to develop a predictive model that predicts time to next vulnerability. The software considered in their study were Linux and Linux OSes and Mozilla web browser. The machine learning methods were radial basis function (RBF) network, sequential minimal optimization (SMO), multilayer perceptron (MLP), and simple logistic. The raw input variables were software's name, vulnerability's published time, software's version, and software's CVSS. Despite their claim that better prediction capability can be achieved by transforming the software's version variable to be the distance between two different versions, the false positive rates of their predictive models are high (about 40%) with approximately accuracy 70% using SMO which is their best performance.

Edkrantz and Said [30] studied the vulnerabilities from NVD up to May 2015. They applied Support Vector Machines (SVM), K-Nearest-Neighbors (KNN), Naive Bayes, and Random Forests ML methods to predict exploit status of vulnerabilities.The variable space considered includes CVSS scores and parameters, CWE variables, length of the text summary, words, vendors, and references. Considering the binary classification, the SVM with Radial Basis Function (RBF) kernel has the best performance, among others, with prediction accuracy, precision, and recall of nearly 83%. They also commented that their prediction accuracy is less than the accuracy presented in Bozorgi, Saul, Savage, and Voelker study [15] because the OSVDB and the extracted large number of variables were not available to them.

6.3 Data and Methodology

6.3.1 Research Data

The research data is comprised of original data, namely **Confidentiality Impact, Integrity Impact, Availability Impact, Access Complexity, Authentication, Gained Access**, collected for Microsoft, Apple, and Linux computer operating systems vulnerabilities from the CVEdetails

site. The operating systems names list is presented in Table 5.1. The CVEdetails site's variables and their description are in Appendix A. Additionally, we include the following data that are computed and collected using the original data in the CVEdetails site:

- **Year**: The year that the OSes vulnerability was published.
- **Days**: Number of days between the publish date and the most recent update date [1] of the vulnerability.
- **Frequency**: An indicator variable; whether the vulnerability affects single or multiple OSes.
- **NumAttacks**: Number of vulnerability types defined in CVEdetails site.
- **Family**: The family of each OS (Table 5.1)
- **Polarity**: The polarity of the textual summary of the vulnerability; a score that ranges from $[-1, 1]$.
- **Subjectivity**: The subjectivity of the textual summary of the vulnerability; a score that ranges from $[0, 1]$.

Natural Language Processing (NLP) was used to compute the polarity and subjectivity of the summary text of each vulnerability. Note that we assume the polarity score is nearly 0, where any score with negative or positive signs indicates the negativity and positivity toward the description of the vulnerability. Also, we assume the subjectivity score is nearly 0, where any score near 0 or 1 indicates the objectivity and subjectivity toward the description of the vulnerability.

For example, for the vulnerability (CVE-2018-0751) that was published in January 4, 2018, and its description is given by Table 6.1.

The vulnerability (CVE-2018-0751) is a weakness in Microsoft products; Windows 10 and Windows 8.1 computer operating systems. It was published with a summary text; "**The Windows Kernel API in Windows 8.1 and RT 8.1, Windows Server 2012 and R2, Windows 10 Gold, 1511, 1607, 1703 and 1709, Windows Server 2016 and Windows Server, version 1709 allows an elevation of privilege vulnerability due to the way the Kernel API enforces permissions, aka "Windows Elevation of Privilege Vulnerability". This CVE ID is unique from CVE-2018-0752.**"

[1] The update date is up to the collection date of the research data. March 15, 2019

Table 6.1: The collected variables for the vulnerability CVE-2018-0751

Variable	Value	Variable	Value
CVE ID Number	CVE-2018-0751	Days	299
Family	Microsoft	Vuln level	low
CVSS Score	3.6	Year	2018
Confidentiality Impact	Partial	Polarity	0.125
Integrity Impact	Partial	Subjectivity	0.6875
Availibility Impact	None	Frequency	1
Access Complexity	Low	Denial Of Service	0
Authentication	Not required	Gained Access	None
NumAttacks	0	- -	- -

We collected the same data for all computer operating systems vulnerabilities included in this research.

In what follows we introduce the five machine learning methods that are selected to develop classification models for Microsoft, Apple, and Linux operating systems vulnerabilities.

6.3.2 Machine Learning (ML)

The machine learning (ML) classification methods were selected because the research response variables are categorical, such as the vulnerability severity levels and types. Logistic Regression (LR), K nearest neighbors (kNN), Gaussian Naive Bayes (GNB), Random forest (RF), and Adaptive Boosting (AdaBoost) ML methods are used to estimate an operating system vulnerability severity level given its vulnerability description. Moreover, the RF regression method is used to estimate an operating system vulnerability score given its vulnerability description. Next, the definitions and classification process of these methods is presented. The response (dependent) variable, for binary class, $y^{(i)}$ takes binary values $\{0, 1\}$ whereas the n explanatory (risk) variables $\{x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\}$, takes $i = 1, 2, ..., N$. The sample (i.e. vulnerabilities) size is N=N_{tr} (training dataset)+N_{ts} (testing dataset). The full data can be expressed as $\{(\mathbf{x}_j^{(i)}, y^{(i)})\}_{i=1}^{N}$, where $j = 1, 2, ..., n$. By developing a predictive model, we estimate the parameters $\{\beta_j\}$ that link the explanatory variables to the response variable using the training dataset, then evaluate the model using the testing (hidden) dataset.

In what follows we present an overview of the first machine learning method; namely, Logistic Regression.

6.3.2.1 Logistic Regression (LR)

In LR, each variable should have a single value for each class, where variables' dependency doesn't prevent applying it. Logistic sigmoid, $S(\mathbf{x})$, is a function used to ensure that the model outcome is bounded between zero and one, and is given by:

$$S(\mathbf{x}) = \frac{1}{1+\exp\{-\beta_j^T \cdot \mathbf{x}\}}\,. \tag{6.3.2.1.1}$$

The analytical form of LR is given by:

$$Log\frac{P(x)}{1-P(x)} = \boldsymbol{\beta^T} \cdot \mathbf{x}\,. \tag{6.3.2.1.2}$$

Numerical optimization methods like the Newton-Raphson method are used to obtain approximate maximum likelihood estimates (MLEs) for the coefficients $\{\hat{\beta}_j\}$ of each class. The conditional probabilities of each class (c) given the testing dataset ($P(C_c|\mathbf{x})$) are computed and then assigned to the class of the one that gives the maximum probability, that is:

$$P(C_c|\mathbf{x}) = \begin{cases} \frac{\exp\{\boldsymbol{\beta_c^T}\cdot\mathbf{x}\}}{1+\sum_{j=1}^{c-1}\exp\{\boldsymbol{\beta_j^T}\cdot\mathbf{x}\}}, & c = 1,2,..,C-1 \\ \frac{1}{1+\sum_{j=1}^{c-1}\exp\{\boldsymbol{\beta_j^T}\cdot\mathbf{x}\}}, & c = C \end{cases}\,. \tag{6.3.2.1.3}$$

Since the probabilities sums to one, we only need to calculate k-1 probabilities, where the last one is one minus the summation of the rest of the probabilities, [46].

In what follows we present an overview of the second machine learning method; namely, K nearest neighbors.

6.3.2.2 K nearest neighbors (KNN)

KNN is considered a simple and intuitive method. It stores all available cases from the training set, then for each testing data point with a predetermined K value, say K=k, for now, it chooses the k nearest data points from the training set, and classifies this data point to the class that the majority of data points belong to. To avoid the problem of ties, it is practically recommended to choose an odd number for k, [13]. Therefore, KNN is a classification method based on similarity measure (i.e. distance functions). There are three commonly used distance functions when considering

continuous variables:

$$\begin{aligned}&\text{Euclidean distance} : \sqrt{\textstyle\sum_{j=1}^{N_{tr}}(x_i - x_i^*)^2},\\ &\text{Manhattan distance} : \textstyle\sum_{j=1}^{N_{tr}} |x_i - x_i^*|,\\ &\text{and}\\ &\text{Minkowski distance of order p} : (\textstyle\sum_{j=1}^{N_{tr}}(|x_i - x_i^*|)^p)^{1/p}\,.\end{aligned} \tag{6.3.2.2.1}$$

KNN can also be applied when considering categorical variables using the Hamming distance. It measures the number of instances in which corresponding symbols are different in two strings of equal length, and is given by:

$$\text{Hamming distance } D_H : \sum_{j=1}^{N_{tr}} |x_i - x_i^*|,\ \text{if } x = x^*, \text{ then } D_H = 0,\ \text{otherwise } D_H = 1\,. \tag{6.3.2.2.2}$$

In what follows we present an overview of the third machine learning method; namely, Gaussian Naive Bayes.

6.3.2.3 Gaussian Naive Bayes (GNB)

GNB is a classification method that is based on Bayes rule ($P(A|B) = \frac{P(A \cap B)}{P(B)}$, for any event A and B). GNB has a strong assumption that variables are independent given class label, and is given by:

$$P(x_j|C_c) = \prod_{j=1}^{n} P(x_{ji}|C_c)\,. \tag{6.3.2.3.1}$$

The general Bayes theorem in this regard is given by:

$$P(c|x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}) = \frac{\prod_{j=1}^{n} P(x_{ji}|c) \cdot P(c)}{P(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)})}\,. \tag{6.3.2.3.2}$$

In addition, GNB assumes normality for each continuous variable which is hard to maintain in real data (i.e. vulnerability), and therefore makes it to have lower performance. We have shown in Chapter 5 that the operating systems vulnerability data do not follow Normal distribution, but GNB is selected because it is commonly used machine learning method. The MLEs of the model

parameters are used to calculate the conditional probabilities and class assignment is based on the highest probability, [46].

In what follows we present an overview of the fourth machine learning method; namely, Random forest.

6.3.2.4 Random Forests (RF)

Leo Breiman proposed a bootstrap aggregation scheme that builds a large collection of decision trees (classification/regression trees), where at each tree the final decision is made after considering the results from those trees. He named this scheme Random Forests.

For the rest of the present study, we refer to a collection of tree-structured classifiers as $\{T(\mathbf{x}, \Theta_k)\}$.

Prior to introducing the classification process of Random Forest method, we shall briefly introduce Decision Tree machine learning method in the following subsection.

6.3.2.4.1 Decision Tree

A typical decision tree model is presented by Figure 6.1.

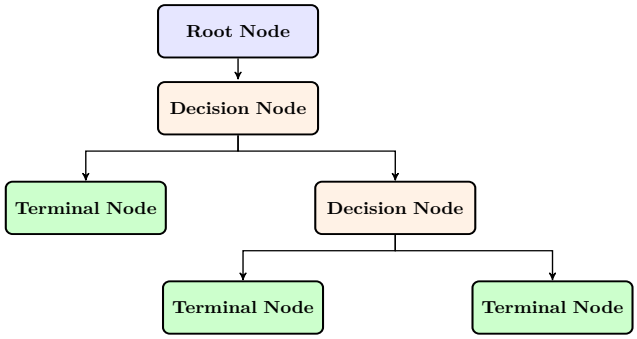


Figure 6.1.: Decision Tree model

The root node in decision tree represents the whole dataset, whereas splitting is the division process of a node into sub-nodes. The decision node is a sub-node that is divided (split) into another sub-nodes. A node is called terminal node when there is not further division. Pruning is the opposite of splitting; when we remove existing nodes in the tree.

Decision tree divides the data space into non-overlapping sub-spaces (regions), which represented by the terminal nodes of the trees [46]. Each region (R_j) is assigned a constant value c_j, which represents the result of the following decision rule:

$$\forall\, x \in R_j \Rightarrow f(x) = c_j\,. \tag{6.3.2.4.1}$$

That is, for a given region j (R_j), then the constant response can be modeled as the average of all responses in this region (node). That is,

$$\hat{f}(x) = \frac{1}{n_j} \sum_{j=1}^{J} c_j \cdot I_{(x \in R_j)}, \tag{6.3.2.4.2}$$

where n_j representa the number of observations in region j.

Note that the problem becomes finding the splitting attributable variable, the split value s, and when to stop splitting (i.e. the tree shape). Minimizing the sum of squares ($\sum_i y_i - f(x_i)$) in a regression tree is a practical criterion for finding the subject values, [46].

The regression and classifications trees, in contrast to Discriminant Analysis, provide a non-parametric data driven model for each region (i.e. class label) [17].

When our data consists of p attributable variables and a continuous response variable, then we proceed to grow a regression tree. Let $(y_i, \mathbf{x}_i)$ represents the data with $i = 1, \cdots, N$ observations and $\mathbf{x}_i = (x_{i1}, \cdots, \mathbf{x}_{ip})$. To decide on a splitting attribute x_{ij} and a split value s, a greedy method is one way to find the best pair (j, s) that minimizes the sum of squares ($\sum_i y_i - f(x_i)$). The data then is divided by using the calculated pair (j, s), and same process can be applied on the resulting regions.

Growing a large regression tree might cause overfitting the data. One way to avoid that is to set a minimum node size (i.e. say 7), then prune the regression tree by applying the cost-complexity pruning technique [17, 46]. Let:

$$\begin{aligned} N_m &= \text{ number of } x_i \in R_m, \\ \hat{c}_m &= \tfrac{1}{N_m} \textstyle\sum_{x_i \in R_m} y_i, \\ Q_m(T) &= \tfrac{1}{N_m} \textstyle\sum_{x_i \in R_m} (y_i - \hat{c}_m)^2, \end{aligned} \tag{6.3.2.4.3}$$

then the cost-complexity criterion is defined by:

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m \cdot Q_m(T) + \alpha |T|, \tag{6.3.2.4.4}$$

where $|T|$ is the total number of terminal nodes in the regression tree. The key concept of applying cost-complexity pruning technique is to find $T_\alpha \subseteq T_0$ to minimize the $C_\alpha(T)$. If α is fixed to 0, then the resulting regression tree will be the full tree T_0.

When our data consists of p attributable variables and a categorical response variable, we grow a classification tree. To decide on a splitting attribute x_{ij} and a split value s, a greedy method is one way to find the best pair (j, s) that minimizes the Gini index, is given by:

$$\sum_{k \neq k^*} \hat{p}_{mk} \cdot \hat{p}_{mk^*} = \sum_{k=1}^{K} \hat{p}_{mk} \cdot (1 - \hat{p}_{mk}), \tag{6.3.2.4.5}$$

where $\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I_{y_i = k}$ is the propotion of a class k instances in node m, and I is an indicator function.

Decision tree classification of a new instance might have a high variance which is a major drawback. This might be caused by having a simple tree-structured and instability of the split technique. Also, any error in the first split will cause an increase of the error until the last split. However, bootstrap aggregation scheme, which is adapted by Random Forests method, should effectively solve this problem, [16, 46].

In the following subsection we introduce a detailed overview of Random Forests classification procedure.

6.3.2.4.2 Random Forests method

Breiman defined Random Forests as, "A random forest is a classifier consisting of a collection of tree-structured classifiers $\{h(\mathbf{x}, \Theta_k), k = 1, ...\}$ where the $\{\Theta_k\}$ are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input $\mathbf{x}$".

We proceed to list a typical process of a Random Forests procedure [46, 16], that is,

1. For $i = 1, \cdots, I$:

   1.1. Form a subset of the training data by drawing randomly with replacement (i.e. bootstrapping).

   1.2. Grow a classification/ regression tree out of each subset (i.e. the bootstrapped sample), by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.

      i. Select m variables at random from the p variables. $m = p/3$, is recommended, [16].

      ii. Pick the best variable/split-point among the m variables using a greedy algorithm.

      iii. Split the node into two daughter nodes.

2. Store the ensemble of trees $\{T_i\}_1^I$.

3. Make a prediction at a new point x:

   3.1. Regression: $\hat{f}_{rf}^I(x) = \frac{1}{I}\sum_{i=1}^I T_i(x)$.

   3.2. Classification: $\hat{C}_{rf}^I(x)$=majority vote of $\{\hat{C}_i(x)\}_1^I$, where $\hat{C}_i(x)$ is the class prediction of the i^{th} random-forest tree.

Thus, Random Forests can be formally defined as a large collection of randomized base decision trees $\{T_n(\mathbf{X}, \Theta_m, \mathcal{D}_n), m \geq 1\}$, where $\Theta_1, \Theta_2, \cdots$ are identically and independently distributed outputs of the randomizing variable Θ [12].

When structuring a decision tree, the randomizing variable Θ is used to determine what attributable variable to split and the split position. Θ is assumed independent of the attributable variables $\mathbf{X}$ and the training data $\mathcal{D}_n$, [12].

In the regression case, each randomized tree $T_n(\mathbf{X}, \Theta_m)$ calculates the average of the response that corresponds to the randomly selected attributable variables, and is given by, [12]:

$$T_n(\mathbf{X}, \Theta_m) = \frac{\sum_{i=1}^n Y_i \cdot I_{[\mathbf{X}_i \in A_n(\mathbf{X}, \Theta)]}}{\sum_{i=1}^n I_{[\mathbf{X}_i \in A_n(\mathbf{X}, \Theta)]}} \cdot I_{[\mathrm{E}_n(\mathbf{X}, \Theta)]}, \tag{6.3.2.4.6}$$

where $A_n(\mathbf{X}, \Theta)$ represents rectangular cell of the random partition containing $\mathbf{X}$. The event $\mathrm{E}_n(\mathbf{X}, \Theta)$ is defined by $[\sum_{i=1}^n I_{[\mathbf{X}_i \in A_n(\mathbf{X}, \Theta)]} \neq 0]$, which estimates the empty cells by 0, [12].

According to Biau [12], by taking the expectation of equation 6.3.2.4.6 with respect to the parameter Θ, the Random Forests regression model ($\bar{T}_n(\mathbf{X})$) takes the following form:

$$\bar{T}_n(\mathbf{X}) = \mathop{\mathbf{E}}_{\Theta}[T_n(\mathbf{X}, \Theta_m)] = \mathop{\mathbf{E}}_{\Theta}[\frac{\sum_{i=1}^n Y_i \cdot I_{[\mathbf{X}_i \in A_n(\mathbf{X}, \Theta)]}}{\sum_{i=1}^n I_{[\mathbf{X}_i \in A_n(\mathbf{X}, \Theta)]}} \cdot I_{[\mathrm{E}_n(\mathbf{X}, \Theta)]}]. \tag{6.3.2.4.7}$$

Based on the Strong Law of Large Numbers and the tree structure, Breiman showed that Random Forests do not overfit as we add more decision trees [16].

Random Forests can produce a generalization performance assessment (i.e. prediction accuracy) using Out Of Bag (OOB) observations. Those observations are in the original dataset but are not in the subsets at each bootstrapping step.

Another advantage of the Random Forests is the variable importance. For each variable, the split criterion in each tree helps measuring its importance to the prediction improvement, and then accumulate it over all trees in the forest [46]. Therefore, the process of finding the best pair (j, s)

that reduces the sum squares error (or reduces the Gini index in the classification case), can be used as an index to rank the relevancy (**importance**) of the variables to the prediction task.

In what follows we present an overview of the fifth machine learning method; namely, Adaptive Boosting.

6.3.2.5 Adaptive Boosting (AdaBoost)

Boosting in machine learning is a way of combining several weak classifiers into one strong classifier. AdaBoost was introduced by Freund and Schapire, [36] in 1996. It is considered an iterative method, where it starts from a weak classifier (i.e. low classification performance). Then apply it several times where each time the training data points receive a certain weigh. Adaboost classification process and the resulting algorithm can be summarized as follows:

- For a given training dataset with size m, the training data points are assigned a weight; starting with uniform weight $D_1(i) = \frac{1}{N_{tr}}$ for $i = 1, 2, ..., N_{tr}$.

- Apply the weak classifier to the weighted training dataset.

- Update the weight of the training data points by giving the mis-classified data points more weight, as an attempt to correct the errors (i.e. for the classifier to perform better in correctly classifying them).

- The final classifier (H(x)) consists of the linear combination of t weak classifiers, given by:

$$H(x) = Sign(\sum_{t=1}^{T} \alpha_t h_t(x)) \,, \tag{6.3.2.5.1}$$

  where α_1,α_2,...,α_t weight the contribution of each respective weak classifier $h_t(x)$.

  Each weak classifier will be given a weight, and will be given a positive sign when it is accurate, minus sign when it is not accurate (i.e. less than 50%), and zero when its accuracy is 50%.

- The classification decision is based on the sign of the summation in previous step, that is, for -1 and +1 outcome.

- Note that α_t is based on the classifier error (ε_t), that is,

$$\alpha_t = \frac{1}{2} \cdot ln(\frac{1-\varepsilon_t}{\varepsilon_t}) \,. \tag{6.3.2.5.2}$$

Note that Adaboost makes the first update for the weight after computing α_1, that is,

$$D_2(i) = \frac{D_1(i) \exp\{-\alpha_1 y_i h_1(x_i)\}}{Z_1}, \tag{6.3.2.5.3}$$

where Z_1 is the summation of all weights.

Thus far, we have introduced the five selected machine learning methods for classifying the severity level of a new computer systems vulnerability. In what follows we present an overview of evaluation metrics that will be used to evaluate the classification and regression machine learning models.

6.3.3 Evaluation metrics

There are four evaluation metrics will be used for each of the classification and regression models. In the following subsections, we state some definitions and analytical forms that are used for the evaluation metrics.

6.3.3.1 Confusion matrix: Accuracy, Precision, Recall, and F-measure

Classification models, like other analytic models, need to be evaluated for their performance and effectiveness before they are used. There are several evaluation metrics for classification purposes, one of which is the confusion matrix, displayed in Table 6.2 below.

Table 6.2: Confusion matrix in cyber attack context for binary classification

Attack status	Classified as positive	Classified as negative
Positive	true positive (tp)	false negative (fn)
Negative	false positive (fp)	true negative (tn)

Table 6.2 shows the confusion matrix in cyber attack context for binary classification. The true postive (tp) is when the attack is correctly positive and classified as positive whereas the false

negative (fn)is when the attack is incorrectly classified as negative. The correctly negative attack is classified as positive (false positive) whereas it is true negative (tn), the negative attack is correctly classified. These metrics provides more insight about the performance of the ML classification methods than just the accuracy (A) which only shows the correctly classified attacks over all benign (negative) and attack (positive) cases, [91, 48]. In addition, there are useful metrics using the confusion matrix, namely **precision**, **recall**, **receiver operating characteristic (ROC)** curve, and **F–measure**. Table 6.3 has the analytical forms of the evaluation metrics considering a multi-class classification.

Table 6.3: Analytical form for each evaluation metric, where c is the total number of classes.

Metric	Analytical form
Average Accuracy (AA)	$\frac{\sum_{i=1}^{c} \frac{tp_i+tn_i}{tp_i+fp_i+fn_i+tn_i}}{c}$
Average Precision (AP)	$\frac{\sum_{i=1}^{c} \frac{tp_i}{tp_i+fp_i}}{c}$
Average Recall (AR)	$\frac{\sum_{i=1}^{c} \frac{tp_i}{tp_i+fn_i}}{c}$
Average F–score (AF–score)	$(1+\beta^2) \cdot \frac{AP \cdot AR}{(\beta^2 \cdot AP)+AR}$

Precision (P) is defined as the number of correctly positive attacks that are classified as positive attacks (tp) over the number of correctly positive and negative attacks that are classified as positive (tp + fp). Recall (R) is defined as the number of correctly positive attacks that are classified as positive attacks (tp) over the number of correctly positive that are classified as positive and negative (tp + fn), it provides an insight of the effectiveness of the ML method in correctly classifying the positive attacks. The ROC curve is defined by $\frac{1}{2}(\frac{tn}{tp+fn} + \frac{tn}{tp+fn})$. It is used to evaluate the ML method; ability to avoid false positive and false negative classifications. F–measure (or F–score) is the harmonic average of precision and recall where it balances between them, and considers them as perfect values when it reaches 1. It is defined by $(1+\beta^2) \cdot \frac{P \cdot R}{(\beta^2 \cdot P)+R}$. When $\beta^2 = 1$, it means the precision and recall are given the same weight when computing the F–score and would be called F_1–score. These evaluation metrics can be generalized for multi-class classification except for the ROC curve which is only for binary classification, [59, 91].

In addition, the time (T) that ML classification methods consume to train and test the predictive models can be included in the previous metrics.

6.3.3.2 Explained Variance, R-squared, Adjusted R-squared, and Root Mean Squared Error

Considering the statistical regression modeling, there are commonly used evaluation metrics such as the Explained Variance (EV), R-squared (R^2), Adjusted R-squared (R_a^2), and Root Mean Squared Error ($RMSE$). Table 6.4 below presents the analytical forms of these evaluation metrics.

Table 6.4: Analytical form for each evaluation metric considered in statistical regression modeling

Metric	Analytical form
Explained Variance (EV)	$1 - \frac{VAR[\hat{y}-y]}{VAR[y]}$
R-squared (R^2)	$1 - \frac{\frac{1}{N}\sqrt{\sum_{j=1}^{N}(y_j-\hat{y}_j)^2}}{\frac{1}{N}\sqrt{\sum_{j=1}^{N}(y_j-\bar{y}_j)^2}}$
Adjusted R-squared (R_a^2)	$1 - [\frac{(1-R^2)(N-1)}{N-n-1}]$
Root Mean Squared Error ($RMSE$)	$\sqrt{\frac{1}{2}\sum_{j=1}^{N}(y_j - \hat{y}_j)^2}$

Note that these evaluation metrics compare the ground truth (hidden testing) vulnerability scores to the predicted ones.

6.3.3.3 K-fold Cross Validation

Usually, during the process of building the predictive analytical models, the dataset is split randomly into training (usually 70-80% of the data) dataset and the rest is testing dataset. Even though each data point based on this split will have the same chance to be in the training and testing datasets, the models will only be trained using the training dataset. To limit the ML classification methods from over-fitting, it is highly recommended that one uses the K-fold cross validation method [13]. It is basically repeating training and testing the models K- times in a systemic manner. For example, in the case of a 10-fold cross validation, the dataset is split 10 times. The first time, the first 10% compose of the testing dataset and the rest are the training dataset. The second time, the second 10% compose of the testing dataset and the rest are the training dataset, and so on until the final time, where the last 10% compose of the testing dataset and the rest are the training dataset. This method assures that each data point will be in the training dataset K-1 times and only one time in the testing dataset, and eventually each data point will be in the testing dataset.

6.4 Results and Discussions

6.4.1 Modeling vulnerability severity levels

The CVSS v2 base score is classified into three levels, namely Low (0-3.9), Medium (4-6.9), and High (7-10). The high level means the vulnerability has highest severity status. There were 16,164 published OSes vulnerabilities to be considered (shown in Table 5.1) by March 15, 2019. The vulnerability severity levels for each OS are presented in Figure 6.2, below:

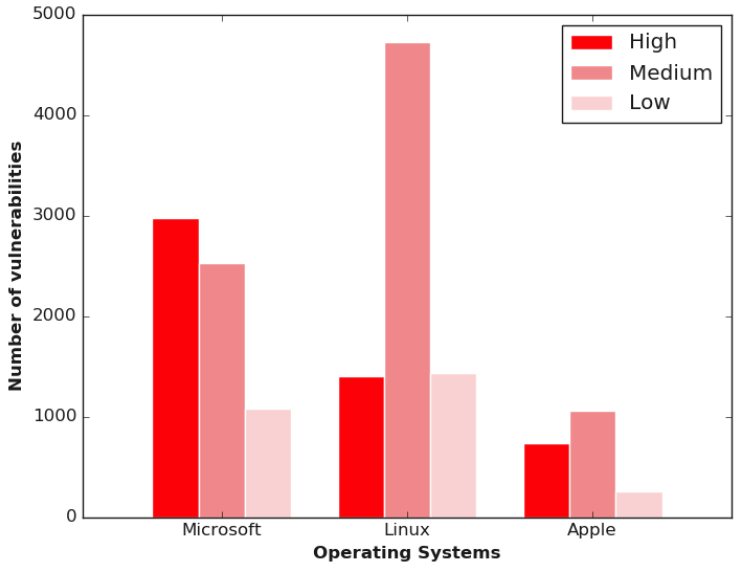


Figure 6.2.: Frequency of each vulnerability severity level within computer operating systems families

The Microsoft OSes have the largest number of high vulnerabilities levels, followed bt Linux and Apple OSes, respectively. The Linux OSes have the largest number of medium venerabilities, followed by Microsoft and Apple OSes, respectively. The Apple OSes have the lowest number of low vulnerabilities compared to Microsoft and Linux OSes.

6.4.1.1 Classification results

The ML classification methods were ran to build models, subject to the 13 risk factors, that classify the severity level of a given operating system vulnerability.

A data driven LR model was developed where Newton-Raphson method was used to obtain approximate maximum likelihood estimates of its coefficients ($\hat{\beta}_j$), and is given by:

$$\hat{LR}(\mathbf{x}) = Log\frac{P(x)}{1-P(x)} = \hat{\boldsymbol{\beta}}^{\boldsymbol{T}} \cdot \mathbf{x}\,. \tag{6.4.1.1}$$

The approximate maximum likelihood estimates of the coefficients (β_j) of the LR model for each vulnerability severity level are presented in Table 6.5 below. The results of the evaluation metrics of the developed classification model are presented in Tables 6.7 and 6.8.

Table 6.5: Approximate maximum likelihood estimates of the LR models' coefficients ($\hat{\beta}_j$) for each vulnerability severity level

Variable	Coefficient	Low level	Medium level	High level
Intercept	$\hat{\beta}_0$	118.001	-31.366	-86.635
Year	$\hat{\beta}_1$	-0.059	0.015	0.044
Days	$\hat{\beta}_2$	0.000	0.000	0.000
NumAttacks	$\hat{\beta}_3$	-0.449	0.008	0.441
Polarity	$\hat{\beta}_4$	3.246	0.059	-3.305
Subjectivity	$\hat{\beta}_5$	-0.819	-0.849	1.668
Confidentiality Impact	$\hat{\beta}_6$	1.122	-0.209	-0.913
Integrity Impact	$\hat{\beta}_7$	-0.528	0.486	0.043
Availibility Impact	$\hat{\beta}_8$	0.363	0.178	-0.541
Access Complexity	$\hat{\beta}_9$	-0.294	0.688	-0.393
Authentication	$\hat{\beta}_{10}$	0.784	0.342	-1.127
Gained Access	$\hat{\beta}_{11}$	-0.756	0.187	0.569
Family	$\hat{\beta}_{12}$	0.312	-0.271	-0.040
Frequency	$\hat{\beta}_{13}$	-0.166	-0.127	0.293

A data driven KNN model was developed where the Minkowski distance function in equation (6.3.2.2.1) and k=3 neighbors were chosen after an extensive search to maximize the accuracy rate for classifying an OS vulnerability severity level. The results of the evaluation metrics of the developed classification model are presented in Tables 6.7 and 6.8.

GNB assumes the variables in the present study are independently and identically distributed. The operating systems vulnerability data do not follow any well-defined probability distribution. However, we proceeded to develop a data driven GNB model because it is a commonly used method. The approximate maximum likelihood estimates of the mean and variance of each variable are presented in Table 6.6 below. They are used to develop the GNB analytical model.

GNB analytical model is given by equation (6.3.2.3.2). The results of the evaluation metrics of the developed classification model are presented in Tables 6.7 and 6.8.

RF method was applied with $T = 200$ trees after an extensive search to maximize the accuracy rate for classifying an OS vulnerability severity level. A data driven RF model was developed

Table 6.6: Approximate maximum likelihood estimates of the mean (μ) and variance (σ) of each variable within each vulnerability severity level.

Variable	Low level		Medium level		High level	
	$\hat{\mu}$	$\hat{\sigma}$	$\hat{\mu}$	$\hat{\sigma}$	$\hat{\mu}$	$\hat{\sigma}$
Year	2013.18	26.77	2013.59	20.37	2012.70	22.35
Days	1360.39	3112082.91	1246.59	2413027.49	1709.93	2885410.36
NumAttacks	0.86	0.33	1.18	0.92	1.44	0.91
Polarity	0.04	0.02	-0.02	0.01	-0.03	0.01
Subjectivity	0.40	0.08	0.34	0.04	0.36	0.04
Confidentiality Impact	1.60	0.24	1.36	0.44	0.50	0.68
Integrity Impact	1.21	0.17	1.35	0.38	0.51	0.68
Availibility Impact	1.27	0.20	1.32	0.61	0.43	0.67
Access Complexity	1.19	0.38	1.47	0.32	1.24	0.26
Authentication	1.09	0.10	1.07	0.07	1.02	0.02
Gained Access	1.00	0.00	1.03	0.06	0.91	0.18
Family	1.22	0.44	0.99	0.38	1.30	0.53
Frequency	0.53	0.25	0.50	0.25	0.63	0.24

where equation (6.3.2.4.7) presents its analytical form . The results of the evaluation metrics of the developed classification model are presented in Tables 6.7 and 6.8.

AdaBoost method was applied with RF as a weak classifier, which reveals a better performance with the vulnerability data comparing to other weak classifiers such as Decision Tree and GNB methods. A data driven AdaBoost model was developed where its analytical form is presented by equation (6.3.2.5.1). The results of the evaluation metrics of the developed classification model are presented in Tables 6.7 and 6.8 below.

Table 6.7 presents the performance of the data driven classification models based on the precision, recall, F_1–score, and accuracy.

Table 6.7: Performance of ML methods in classifying the OSes vulnerabilities' levels.

Classifier	Precision	Recall	F_1–score	Accuracy
Logistic Regression	0.71	0.71	0.7	0.71
K nearest neighbors	0.77	0.77	0.77	0.77
Gaussian Naive Bayes	0.69	0.67	0.68	0.67
Random Forest	**0.93**	**0.93**	**0.93**	**0.93**
Adaptive Boosting	0.92	0.92	0.92	0.92

The RF and Adaboost classification methods outperform the LR, KNN, and GNB, where their evaluation metrics values were at 93% and 92%, respectively. In addition, Table 6.8 shows the number of mis-classified vulnerabilities, training times, and testing times.

Table 6.8: Performance of ML methods in classifying the OSes vulnerabilities' levels based on the miss-classified, training and testing times.

Classifier	Miss-Classified	Training Time	Testing Time
Logistic Regression	1414	0:00:02.409	0:00:00.000
K nearest neighbors	1114	0:00:00.018	0:00:00.104
Gaussian Naive Bayes	1579	0:00:00.004	0:00:00.001
Random Forest	**328**	**0:00:01.343**	**0:00:00.150**
Adaptive Boosting	367	0:00:06.126	0:00:00.571

The RF has the highest accuracy and therefore smallest number of miss-classified vulnerabilities. The second best classifier was the Adaboost method with the influence of the RF method. Although the training and testing times for all ML classification methods are very small, the Adaboost method consumed more time than the others.

The 10-fold cross validation was also applied to the ML classification methods to investigate whether the classifier overestimated the vulnerability severity levels. The box-plots of the results appear in Figure 6.3.

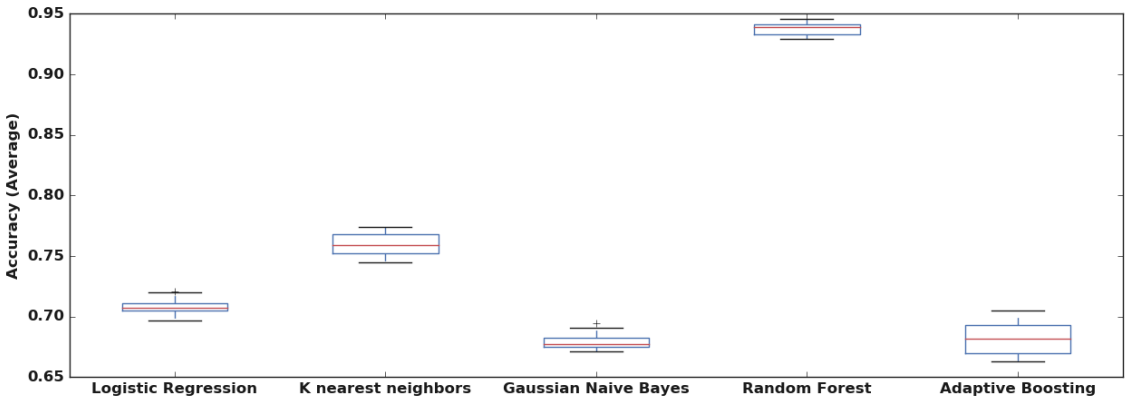


Figure 6.3.: 10-fold cross validation of the ML classification methods

It can be seen that LR, KNN, GNB, and RF have maintained similar performance compared to their respective performance in randomly splitting the training and testing datasets, whereas Adaboost steadily worsened, during the 10-fold cross validation, in classifying the vulnerability severity levels. The box-plots of Figure 6.3 do not indicate any outliers and all results for each method were around the average accuracy values. The Adaboost classification method clearly was over-fitting in classifying the vulnerability severity levels. The average accuracy of the RF classification method is nearly 94%, which is the highest among all methods, which suggests that it is the best method to classify vulnerability severity levels among other methods in the present

study.

6.4.1.2 Denial of Service Attack

The exploitability of any newly discovered OS vulnerability is an important aspect that specialists try to discover in order to prepare a remediation plan. **Denial of Service** attack is a commonly known attack and building a classification model would assist in preventing it. CVEdetails provides information about whether the published OS vulnerability allows attackers to cause denial of service. Using this information, the RF classification method was applied subject to the 13 risk factors. The precision, recall, F_1–score, and accuracy were 93% in classifying whether the newly discovered OS vulnerability would allow attackers to cause denial of service to the system.

6.4.2 Statistical Modeling of Vulnerability Scores

The vulnerability scores (CVSS v2) range from 0 to 10, and it is the continuous response variable in the present study. The Random Forest (RF) method outperforms the other ML classification methods when modeling the vulnerability severity levels. It is also a regression method, and commonly used as a regression method especially when the assumptions of the powerful parametric methods such as the Multiple Linear Regression are not met.

In what follows we develop data driven RF regression models to predict the expected vulnerability scores of Microsoft, Apple, and Linux OSes, combined and individually, vulnerabilities based on the vulnerability 13 risk factors presented in section 6.3.1. .

6.4.2.1 Vulnerability Scores of the Microsoft, Apple, and Linux OSes

The number of the vulnerabilities (from 1999 to 2019) for each OSes' family are different. Table 6.9, below, presents the summary statistics of the vulnerability scores for the Microsoft, Apple, and Linux OSes.

Table 6.9: Summary statistics for Microsoft, Apple, and Linux OSes vulnerability scores.

Family	Count	Mean	Std. Deviation	25^{th} Percentile	50^{th} Percentile	75^{th} Percentile
Microsoft	5107	6.59	2.43	4.9	7.2	9.3
Linux	8304	5.68	1.96	4.3	5	7.2
Apple	2753	6.40	2.11	5	6.8	7.5

Linux OSes have the largest number of vulnerabilities, which may due to the fact that Linux

is an open source where it can be used by multiple vendors to build their own OSes. Yet, it has the smallest mean, standard deviation, and percentiles ($25^{th}, 50^{th}$, and 75^{th}). Vulnerabilities scores of the Windows OSes have the second largest number of vulnerabilities, and have the largest mean, standard deviation, and percentiles (50^{th} and 75^{th}). On the other hand, Apple OSes has the smallest number of vulnerabilities, and second largest mean, standard deviation, and percentiles (50^{th} and 75^{th}).

The RF regression method was performed using all vulnerability scores of the Microsoft, Apple, and Linux OSes vulnerabilities. The data driven RF predictive model is given by:

$$\bar{T}_{200}(\mathbf{X}) = \mathop{\mathbf{E}}_{\Theta}[T_{200}(\mathbf{X}, \Theta_m)] = \mathop{\mathbf{E}}_{\Theta}[\frac{\sum_{i=1}^{n} Y_i \cdot I_{[\mathbf{X}_i \in A_{200}(\mathbf{X},\Theta)]}}{\sum_{i=1}^{n} I_{[\mathbf{X}_i \in A_{200}(\mathbf{X},\Theta)]}} \cdot I_{[\mathrm{E}_{200}(\mathbf{X},\Theta)]}], \tag{6.4.2.1}$$

where $T_{200}(\mathbf{X}, \Theta_m)$ are randomized trees and $A_{200}(\mathbf{X}, \Theta)$ represents rectangular cell of the random partition containing $\mathbf{X}$. A sample of the resulting estimated vulnerability scores of computer operating systems vulnerabilities that we have obtained using the proposed model, along with their respective actual scores is given in Table 6.10.

Table 6.10: A sample of the estimated vulnerability scores using the proposed RF model and the actual respective vulnerability scores of computer operating systems vulnerabilities.

Number	Estimated score	Actual score
1	5.59	6.4
2	4.71	4.6
3	4.90	4.9
4	6.77	6.8
5	7.56	6.9
6	6.73	6.8
7	4.72	5
8	6.79	6.8
9	4.22	5.1
10	7.07	6.8
11	4.71	5

Table 6.11 shows the evaluation metrics of the predictive model using the testing (unseen) vulnerability scores against the predicted vulnerability scores.

The evaluation metrics of the RF regression model provide evidence of its powerful performance. The amount of the variance that is explained by the model is 90% and the residual is nearly zero. The R-squared and the Adjusted R-squared are approximately 90% and equal.

The developed data driven RF regression model is useful in predicting the expected vulnerability

Table 6.11: Evaluation metrics of the Random Forest regression model of predicting the vulnerability scores of Microsoft, Apple, and Linux OSes vulnerabilities combined.

Metrics	Value
Explained Variance (EV)	0.90
R-squared (R^2)	0.90
Adjusted R-squared (R^2_a)	0.90
Root Mean Squared Error ($RMSE$)	0.69
Residual	0.0099

scores of Microsoft, Apple, and Linux OSes vulnerabilities. IT managers can utilize the predictive model to assess and prioritize their operating systems vulnerabilities.

6.4.2.2 Ranks of The Risk Variables

The RF regression method calculates the variables importance during the modeling process, which assists in ranking these variables and prioritizing efforts to fix the OSes vulnerabilities. The variables importance are shown in Figure 6.4.

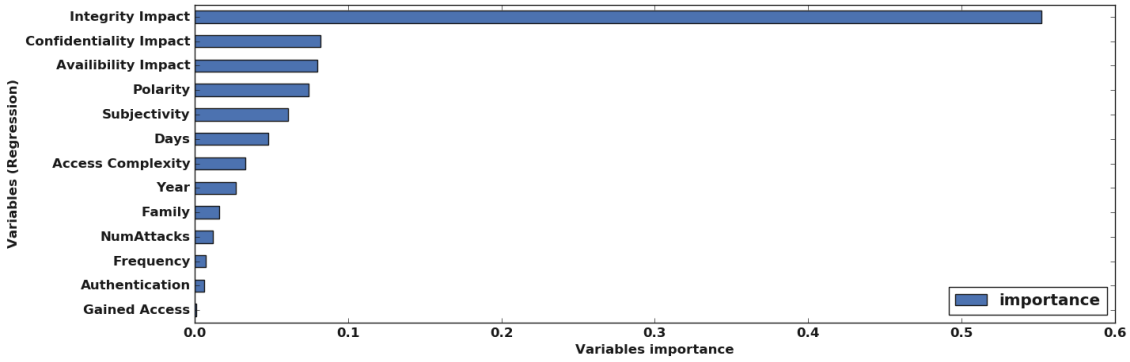


Figure 6.4.: Importance of variables in estimating the vulnerability scores using Random Forest regression method.

It is seen that the OSes integrity, confidentiality, and availability impacts are the three most important variables in predicting the vulnerability scores. The next three important variables are the introduced variables that were not considered in other related studies, namely Polarity, Subjectivity, and Days. Gained access and Authentication are the least important variables, respectively.

Table 6.12 below summarized and present ranks of the risk variables that contributed in estimating the vulnerability scores of Microsoft, Apple, and Linux OSes.

Thus far, we have developed a predictive model to estimate the vulnerability score for a given vulnerability description of an computer operating system subject to the 13 risk factors. As seen

Table 6.12: Ranks of risk variables along with their contribution in estimating the vulnerability scores using Random Forest regression method.

Factor	Rank	Contribution
Integrity Impact	1	0.552
Confidentiality Impact	2	0.082
Availability Impact	3	0.079
Polarity	4	0.074
Subjectivity	5	0.061
Days	6	0.048
Access Complexity	7	0.033
Year	8	0.027
Family	9	0.016
NumAttacks	10	0.012
Frequency	11	0.007
Authentication	12	0.006
Gained Access	13	0.002

in Table 6.9, the number of vulnerabilities and mean values are different for each operating system family, we proceed to apply Kruskal-Wallis test to examine whether the median values of Microsoft, Apple, and Linux OSes are statistically equal. The result of the Kruskal-Wallis test reveals that the Microsoft, Apple, and Linux OSes vulnerability scores come from different probability distributions with p-value = 0.00. Even though the family of the OSes is included in the previous RF regression model, we proceed to develop a data driven RF regression model for each OSes family.

6.4.2.3 Statistical Models of Microsoft, Apple, and Linux OSes Vulnerability Scores

In this section, we proceed to develop RF regression models for the vulnerability scores of Microsoft, Apple, and Linux OSes individually. The RF regression models take the analytical form given in equation (6.3.2.4.7).

A sample of the resulting estimated vulnerability scores of computer operating systems vulnerabilities that we have obtained using the proposed model for each OS family, along with their respective actual scores is given in Table 6.13.

Table 6.14 shows the evaluation metrics of the RF regression models of each OSes family.

The evaluation metrics were different for each of Microsoft, Apple, and Linux OSes. Although for our proposed statistical models both R-squared and the Adjusted R-squared are approximately equal, Microsoft OSes statistical model is the best in explaining the total response (vulnerability score) variation. The residuals of the proposed models are very small which indicate a good quality

Table 6.13: A sample of the estimated (Est.) vulnerability scores using the proposed RF models and the actual (Act.) respective vulnerability scores of Microsoft, Apple, and Linux OSes Vulnerabilities individually.

OS	Microsoft		Apple		Linux	
	Est. score	Act. score	Est. score	Act. score	Est. score	Act. score
1	2.13	2.1	2.90	2.1	7.44	7.8
2	4.96	5	4.70	4.6	7.67	6.8
3	7.26	7.2	4.65	5	4.78	4.7
4	5.75	7.1	6.94	6.8	6.07	6.8
5	7.51	7.6	9.28	9.3	4.74	4.7
6	5.76	5.8	3.33	2.1	5.46	4.4
7	2.14	2.1	7.74	10	3.03	2.1
8	9.83	10	4.93	4.9	6.42	6.8
9	8.17	8.3	4.92	5	8.89	9.3
10	9.17	9.3	6.14	4.6	2.34	5
11	4.94	5	3.83	4.3	7.66	7.2

Table 6.14: Evaluation metrics of the Random Forest regression model for Microsoft, Apple, and Linux OSes individually.

Metrics	Microsoft OSes	Apple OSes	Linux OSes
Explained Variance (EV)	0.96	0.87	0.84
R-squared (R^2)	0.96	0.87	0.84
Adjusted R-squared (R_a^2)	0.96	0.87	0.84
Root Mean Squared Error ($RMSE$)	0.47	0.73	0.79
Residual	-0.0043	-0.0402	0.0011

models.

Noticeably, the predictive model for the vulnerability scores of the Microsoft OSes performs better than the predictive model for the vulnerabilities of Microsoft, Apple, and Linux OSes combined. Followed by the predictive models for the vulnerability scores of the Apple and Linux OSes.

6.5 Contributions

Today, information is stored in systems where security is an important component, along with reliability and availability. Computer operating systems (OSes) such as Microsoft, Apple, and Linux are the focus of this research. Specifically, the vulnerabilities of the OSes were collected from January 1999 to March 2019, and utilized to achieve the research objectives.

We tested several models, namely, Logistic Regression, K nearest neighbors, Gaussian Naive Bayes, Random Forests, and Adaboost to identify the most accurate in predicting a) vulnerability severity level (low (0-3.9), medium (4-6.9), or high (7-10)) b) vulnerability score and c) how likely

it will be for attackers to cause denial of service to the OS. Random Forest Machine Learning was the best model in identifying (predicting) the key factors mentioned with 93% precision, recall, F1-score and most importantly accuracy. The Adaboost algorithm had 92% precision performing the same tasks, however, it consumed more time in the training and testing stages. The other methods evaluated for the similar task, K-nearest neighbor, Logistic Regression, and Gaussian Naive Bayes, did not perform very well, having evaluation metrics of 77% or less.

Additionally, we developed a Random Forest regression model subject to 13 risk factors for each OS family. These predictive models are used to estimate the vulnerability score of Microsoft, Apple, and Linux OSes, individually. These models gave excellent results.

The developed classification and predictive models will assist not only vendors and information technology specialists, but also end-users, in managing and understanding the impact of the unfixed and newly discovered vulnerabilities of their computer operating systems. Also, identifying the ranks of the risk variables for these vulnerabilities assists them in prioritizing the remediation process.

Chapter 7

Parametric and Non-parametric Modeling of a Computer Network Reliability

In this chapter, we continue previous efforts and model, parametrically and non-parametrically, the expected path lengths for a hacker to compromise a computer network system and the minimum number of steps that an attacker needs to hack the network with a very high probability. We use the statistical models provided by Rajasooriya and Tsokos [80], and utilize the National Vulnerability Database (NVD) to calculate the expected path length and minimum number of steps that an attacker will need to hack the network with a very high probability. Moreover, we consider the number of steps necessary to hack the computer network system as the failure time for a certain vulnerability. That is, once the network has been hacked, it has been failed.

We proceed to identify the probability distribution functions that characterize the probabilistic behaviors of the expected path lengths and the minimum number of steps for a hacker to compromise the computer network, respectively. Furthermore, we develop parametric and non-parametric reliability analysis of the network system.

The chapter is organized as follows, Section 2 we review the calculation of the expected path length and minimum number of steps to hack a given computer network system. Section 3 presents the data and methodology. Section 4 presents results and discussion. Section 5 describes the contributions.

7.1 Introductions

The reliability and security of a computer network is very crucial matter. Any attempt to hack into a network with stored information will place the system at substantial risk. Most of government entities, private industries, banks, hospitals, and schools have computer networks connected to servers that store their sensitive information. One way to prevent system breaches is to analyze the hackers behaviour and understand advanced techniques they will use during cybercrime activities.

In the last 10 years American consumers dramatically increased their online presence and negative consequences followed repeated massive data breaches. By current standards [34], breaches are

defined as any "compromise of security" that leads to "**loss, alteration, unauthorized disclosure of, or access to protected data**." Daniel Funke in his very recent article for PolitiFact [37] categorizes data breaches in three ways. First, caused by external actors, i.e., hackers stealing personal/financial information with malicious intent. Second, those which occurred by accident – i.e., a company – holding sensitive information leaks it by mistake via less than fully-competent workers, for example. These breaches are likely to go undiscovered for some time. Third, those caused by the third parties, i.e., external contractors who are not familiar with security measures and standards of the company with whom they contract or work with.

The size of a breach is measured by the number of records exposed. The highest one to date is Yahoo breach, publicized in 2017, of personal data of all its 3 billion customers [37]. It is also important to condider the kinds of data exposed, with personal data and medical records being the most sought, since they are more difficult to replace, and can be used in forging insurance claims. While the past few years we saw an increase in size and frequency of data breaches, Funke states that in the US there is not a single federal law that requires notifying impacted consumers about a data breach. Instead, there is a "patchwork" of 50 state laws, but not each state has in place a procedure that requires organizations that suffer a data breach to give notice to attorney generals. Not all states have laws in place that require the government to share the occurrence of data breaches publicly. Also, there is a discrepancy among the states in their views about what data is considered "sensitive". All these facts makes it virtually impossible to estimate the real economic and business impact caused by data breaches.

Therefore, maintaining a highly secured and reliable computer network is a very important matter. In fact, understanding attackers behaviors may prevent data breaches, or at least complicate the hacking process.

Kaluarachchi and Tsokos [53] utilized the National Vulnerability Database, using a computer network system with a host centric attack graph. The Markov process was then applied by using vulnerability information of each system in the network to understand the behavior of network topology. They also developed a nonlinear statistical model to estimate the expected path length of hacking the computer network as a function of systems vulnerability scores using the Markovian iteration process.

Kaluarachchi and Tsokos [53] also obtained the absorbing matrix for each set of systems vulnerabilities in the network, which identifies the minimum number of steps that the attacker is required to compromise the network with a very high probability. Using these computations, Kaluarachchi

and Tsokos [53] developed another statistical model to predict the minimum number of steps needed for the attacker to reach the goal state with very high probability , considering a given network system.

Product reliability is the probability that it will perform according to predetermined standards over a period of time [14, 22]. Products can be hardware or software, however, our focus in this research is on software products such as computer operating systems.

On the other hand, we have a "network reliability" term which was first introduced by Shannon and Moore [66]. They used the term in their study of the relay circuits reliability. The failure of a relay contact to close and the failure of a contact to open are the only considered errors that cause relay circuits to fail. They defined network reliability as "the probability of the network being closed, when the individual contacts each have probability p of being closed." Since then, the "network reliability" term has been used and continued to be used in several fields.

In the cybersecurity field, the hack event can be considered as a death or failure event of a particular computer system. Therefore, the reliability of a software system can be defined as the probability that the it will not be hacked with respect to a specified period of time (steps). Similarly, we define a reliability of computer network as the probability that it will not be hacked until a predetermined period of time (steps) is reached. In this research, we are considering the network system provided by Kaluarachchi and Tsokos [53], where a computer network is considered hacked if and only if the last computer in the network system is hacked, assuming the hacking event started from the first computer.

To our knowledge, very little if any reliability analysis of computer network system based on their vulnerabilities has been conducted on the subject area. Thus, our reliability study is an initial aspect of research in the field of cybersecurity and hopefully will motivate researchers to conduct such studies.

When a company employs the Rajasooriya and Tsokos network system and their statistical models, where vulnerability information for each computer in the network is available, they can estimate the expected path length for an attacker to hack their network and the minimum number of steps that an attacker will need to successfully hack the network system, with a very high probability. However, alternating the systems in the same network system using different vulnerability information will normally result in a different expected path length and minimum number of steps for an attacker to hack the network with a very high probability. This study is based on a specific computer network system, however, the methodologies and analysis are applicable to more general

computer network system and a desired one. As a continuation of previous efforts, we aim to model, parametrically and non-parametrically, the expected path lengths for a hacker to compromise a computer network and the minimum number of steps that an attacker needs to hack the network with a very high probability and different vulnerability information. We use the statistical models provided by the Rajasooriya and Tsokos [80], and utilize the National Vulnerability Database (NVD) to calculate the expected path length and minimum number of steps that an attacker will need to hack the network with a very high probability.

Moreover, for a given network system that consists of two computers, with three vulnerabilities, v_1, v_2, v_3. We download operating systems vulnerabilities from NVD. Then, we randomly and without replacement select a set of three vulnerabilities and calculated its expected path length. This random selection process is repeated $2,980$ times and thus, we have $2,980$ expected path length values. In addition, we randomly and without replacement select a set of three vulnerabilities and calculated its minimum number of steps for a hacker to compromise the network system with a very high probability. This random selection process is repeated $2,953$ times and thus, we have $2,953$ minimum number of steps values.

Furthermore, we proceed to identify the probability distribution functions that characterize the probabilistic behaviors of the expected path lengths for a hacker to compromise the computer network and the minimum number of steps that an attacker needs to hack the computer network with a very high probability, respectively. Furthermore, we develop parametric and non-parametric reliability functions of the network system.

The parametric and non-parametric modeling will assist cybersecurity specialists not only in monitoring the security status but also in the reliability status of a network, especially in estimating the probability that a randomly selected set of vulnerabilities in the network system will have an expected path length that falls between two desired time units. Similar usefulness applies to the minimum number of steps that an attacker needs to hack the network system with a very high probability. More importantly, estimating the reliability of computer network is very important to the IT manager of a given enterprise.

7.2 Calculation of the Expected Path Length and Minimum Number of Steps to Hack a Given Computer System

Kaluarachchi and Tsokos [53] proposed statistical models to evaluate system's security, based on vulnerabilities information at the attack path's sequences. They used the the Common Vulnerability

Scoring System (CVSS), which is classified into three levels, namely **Low (0-3.9), Medium (4-6.9), and High (7-10)**. The high level means the vulnerability has highest severity status. To begin with, and to have a better picture of the network topology, proceeded with "small case" scenario and investigate 3 basic levels of vulnerability; namely Low, Medium and High. The authors illustrated their concept with the attack graph they designed, that includes all possibilities for the attacker to achieve his/her target – the absorption state (the goal state). Their model is data driven and is being characterized by homogeneous Markovian process.

Given below is a breif review of the Markov Process and the Kaluarachchi and Tsokos analytical model.

7.2.1 Markov Chain and Transition Probabilities

Let S be a discrete set. A Markov chain ($X = \{X_n, n \geq 0\}$) is a sequence of random variables X_0, X_1, ... taking values in S with the property that:

$$P(X_{n+1} = j | X_0 = x_0, ..., X_{n-1} = x_{n-1}, X_n = i) = P(X_{n+1} = j | X_n = i), \tag{7.2.1.1}$$

for all x_0, .., x_{n-1}, i, j $\in S$, and $n \geq 0$. The set S is the space of the Markov chain. Markov chain is time-homogeneous if the probabilities in 7.2.1.1 do not depend on n, [28]. The transition probabilities (P_{ij}) for Markov chain can be defined by $P_{ij} = P(X_{n+1} = j | X_n = i)$.

A stochastic matrix is a square matrix P, which satisfies:

- $P_{ij} \in [0, 1]$ for all i and j,
  and
- for each row i, $\sum_j P_{ij} = 1$.

7.2.2 Transient States

Let P be the transition probability matrix for the Markov chain X_n. With a probability of 1, a state i is named transient state if the chain visits it only a finite number of times. The Markov chain is an absorbing Markov chain if we have at least one absorbing state where we can possibly reach it from any other state by a finite number of steps.

7.2.3 Prediction of Expected Path Length (EPL)

The transition matrix for an absorbing Markov chain [79, 54, 57, 71, 50] has the following canonical form:

$$P = \begin{bmatrix} \mathbf{Q} & \mathbf{R} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}, \tag{7.2.3.1}$$

where $\mathbf{Q}$ is a t-by-t matrix (t-number of transient states), $\mathbf{R}$ is a nonzero t-by-r matrix for absorbing states, $\mathbf{0}$ is an r-by-t zero matrix, and $\mathbf{I}$ is the r-by-r identity matrix. The probability of a chain will be absorbed in an absorbing Markov chain is always 1, therefore we have: $\mathbf{Q^n} \to 0, \text{ as } n \to \infty$.

This implies that all the eigenvalues of $\mathbf{Q}$ have absolute values strictly less than 1 [28]. Then $(\mathbf{I} - \mathbf{Q})$ is an invertible matrix as follows:

$$\mathbf{M} = (\mathbf{I} - \mathbf{Q})^{-1} = 1 + \mathbf{Q} + \mathbf{Q^2} + \mathbf{Q^3} + \dots \quad . \tag{7.2.3.2}$$

Matrix $\mathbf{M}$ is the fundamental matrix of P. The matrix elements, $\mathbf{M}_{i,j}$, give the expected number of visits to state i when the chain started at state j. Hence, to compute the expected number of steps that started at state j before the chain enters a recurrent class, we sum $\mathbf{M}_{i,j}$ over all transient states i [53].

7.2.4 Prediction of the Minimum Number of Steps

After feeding the transition probability matrix P with actual vulnerability scores (CVSS), the authors obtained the minimum required number of steps to achieve the goal state. For this, Kaluarachchi and Tsokos use the matrices P^2, P^3, P^4,... up to k steps P^k. In this process, the absorbing states are achieved and k is considered the number of steps to compromise the network security. They investigated this process by changing the CVSS scores and calculating for each combination of V_1, V_2, and V_3 in order to find the minimum number of steps that the attacker need to reach the goal state with a very high probability.

7.2.5 Network Topology

To illustrate their approach, the authors considered a computer network that is shown by Figure 7.1 below.

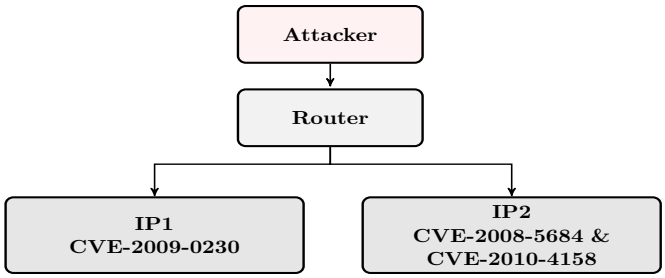


Figure 7.1.: Network Topology- model [53]

The network has two service hosts IP 1, IP 2 and an attacker's workstation. Attacker connects to each of the servers via a central router. Server IP1 has one vulnerability, V_1, while server IP2 has 2 vulnerabilities, V_2 and V_3. Table 7.1 presents the base scores of the vulnerabilities in the network system.

Table 7.1: Base scores of vulnerabilities in the network system

Vulnerability	Base Score
V_1 (CVE-2009-0230)	9
V_2 (CVE-2008-5684)	5
V_3 (CVE-2010-4158)	2.1

Their computer network can be generalized to any desired computer network system.

7.2.6 Host Centric Attack graph

The attack graph the Kaluarachchi and Tsokos model has the attacker state, several nodes representing vulnerabilities, and at least one absorption (goal) state – the node which will be exploited by the attacker. Reaching this node completes the attack path. Figure 7.2 presents the adjacent host centric graph.

The adjacent host centric graph illustrates the case when an attacker reaches the goal (absorbing) state only by exploiting vulnerability V_2. Number "10" on the graph means maximum vulnerability score which allows the attacker the best chance to exploit the vulnerability.

7.2.7 Transition Matrix of the Attack Graph

The states of the network system are: s_1 – **the attacker**, s_2- (**IP1, 1**), s_3 - (IP 2,1) and s_4 - (**IP2,2**), the goal state. In this study the authors use CVSS scores for each vulnerability presented

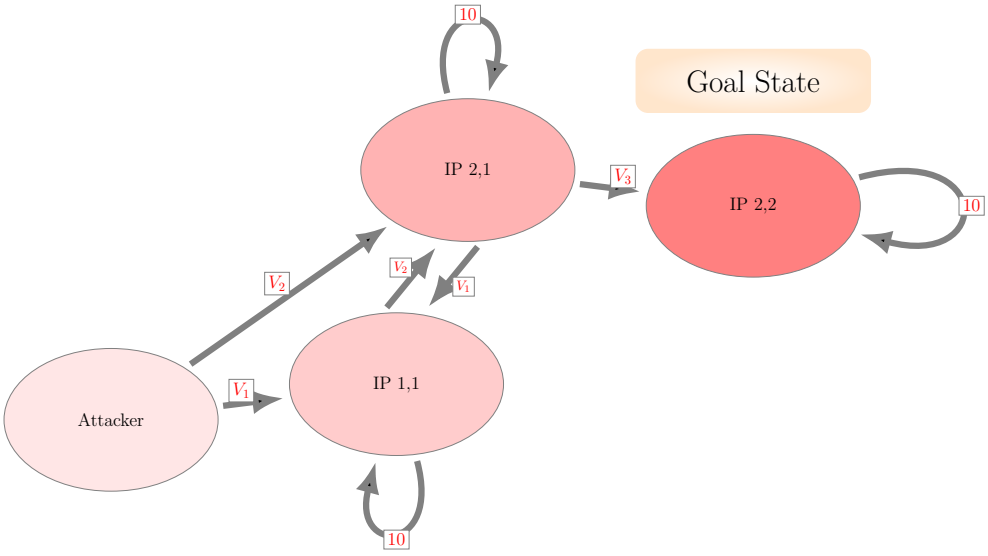


Figure 7.2.: Host centric attack graph [53].

in the attack graph. By normalizing these scores for each transition between states the authors obtained estimates for transition probabilities as follows:

$$\mathbf{P}_{i,j} = \frac{v_j}{\sum_{k=1}^{n} v_k}, \tag{7.2.7.1}$$

where $\mathbf{P}_{i,j}$ is the probability that the attacker will explore the vulnerability in state "i" while being in state "j" of the attack graph; n is the number of edges from state "i" to state "j"; v_j is CVSS score for vulnerability in state "j".

These separate probabilities help to obtain the absorbing probability matrix P that lists all the probabilities of any possible single step attack, i.e. it shows the reachability of the attacker in 1-step distance. The authors then obtain normalised (weighted) value of exploiting each vulnerability from a given state to another state by dividing the vulnerability score by summing of all outgoing vulnerability values. For example, weighted value of exploiting V_1 from s_1 to s_2 is $\frac{V_1}{V_1+V_2}$, and the weighted value of exploiting V_2 from s_1 to s_3 is $\frac{V_2}{V_1+V_2}$, and so on. Eventually, the entire Adjacency Matrix of of attack graph is obtained and is given by:

$$P = \begin{bmatrix} \frac{0}{V_1+V_2} & \frac{V_1}{V_1+V_2} & \frac{V_2}{V_1+V_2} & \frac{0}{V_1+V_2} \\ \frac{0}{10+V_2} & \frac{10}{10+V_2} & \frac{V_2}{10+V_2} & \frac{0}{10+V_2} \\ \frac{0}{V_1+V_3+10} & \frac{V_1}{V_1+V_3+10} & \frac{10}{V_1+V_3+10} & \frac{V_3}{V_1+V_3+10} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{7.2.7.2}$$

7.2.8 Statistical Model to Predict the Expected Path Length

The V_1, V_2 and V_3, in the transition probability matrix, are replaced by the actual CVSS scores, and numerical values in P are obtained. By utilizing the vulnerabilities information given in Table 7.1, we can obtain the transition probability matrix, P, of the attack graph as follows:

$$P = \begin{bmatrix} 0 & 0.6429 & 0.3571 & 0 \\ 0 & 0.6667 & 0.3333 & 0 \\ 0 & 0.4265 & 0.4739 & 0.0995 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{7.2.8.1}$$

We can use P to answer some important questions in security. Each value in the transition probability matrix P represents a probability estimate to exploits the related vulnerability in the network system in first step. For example, the 0.6429 represents the probability that the attacker explots V_1 in first step, which means moving from s_1 to s_2 in the attack graph. The probability estimate that the attacker exploits V_3 and therefore break the network system (reaching the goal state) is 0.0995, as he/she moves from s_3 to s_4.

The authors investigated this process by changing the CVSS score and calculating for each combination of V_1, V_2 and V_3. Then they calculated the expected path Length (i.e. expected number of steps) that the attacker will take to breach the network security starting from the initial state [53, 54] in the attack graph. Using this calculations the authors developed additional statistical models that predicts the expected path length of the attacker. The best statistical mode, in terms of R^2 (coefficient of determination), R^2_{adj} (R^2-adjusted), is presented in Table 7.2.

Table 7.2: Statistical models to estimate the expected path length as a function of network vulnerabilities [53]

Model Equation	R^2	R^2_{adj}
$Y(V_1, V_2, V_3) = 70.62 + 5.338V_1 - 14.108V_2$ $-33.144V_3 - 0.041V_1V_2 + 0.942V_1V_3 + 3.943V_2V_3$ $-0.015V_1^2 + 0.864V_2^2 + 1.814V_3^2 - 0.35V_1V_2V_3$	0.943	0.9378

Here, $Y(V_1, V_2, V_3)$ is the expected path Length for the attacker to exploit the network security (reaching the goal state) starting from the initial state in the attack graph, and (V_1, V_2, and V_3) are the vulnerability scores of the systems in the subject network. The R^2 and R^2_{adj} values shows the high quality of the model. The accounts for the interaction between all 3 vulnerabilities, which helps

the security specialists to evaluate the network security not only using the individual vulnerabilities but their interactions.

In addition, Kaluarachchi and Tsokos [54] utilized the R^2 criteria to rank the attributable variables (vulnerabilities), along with the significant interactions according to their contribution to estimate the expected path length, and is given by Table 7.3.

Table 7.3: Ranking of the variables according to their contribution [54].

Parameter	Rank
V_3^2	1
V_3	2
V_2	3
V_2^2	4
V_2V_3	5
$V_1V_2V_3$	6
V_1	7
V_1V_3	8
V_1V_2	9
V_1^2	10

The third vulnerability in quadratic form (V_3^2) is contributing the most to the estimate of the expected path length, followed by the third vulnerability (V_3). The quadratic form of first vulnerability (V_1^2) is contributing the least to the subject model.

7.2.9 Statistical Model to Predict the Minimum Number of Steps

The authors investigated this process by changing the CVSS scores and calculating the minimum number of steps that the attacker will need to breach the network security with a very high probability [53, 54], and then developed statistical models by using this calculations to predict the minimum number of steps to reach the goal state with a very high probability. The best statistical mode, in terms of R^2 (coefficient of determination), R^2_{adj} (R^2-adjusted), is presented in Table 7.4.

Table 7.4: Statistical models to estimate the minimum number of steps to reach the goal state with a very high probability as a function of network vulnerabilities [53]

Model Equation	R^2	R^2_{adj}
$Y(V_1, V_2, V_3) = 689.84 + 51.177V_1 - 138.815V_2$ $-328.093V_3 - 0.3626V_1V_2 + 9.29V_1V_3 + 39.114V_2V_3$ $-0.084V_1^2 + 8.479V_2^2 + 17.96V_3^2 - 3.47V_1V_2V_3$	0.9428	0.9376

Here, $Y(V_1, V_2, V_3)$ is the minimum number of steps for the attacker to reach the goal state with a

very high probability, and (V_1, V_2, and V_3) are the vulnerability scores of the systems in the subject network. The R^2 and R^2_{adj} values shows the quality of the model. The model also accounts for the interaction between all 3 vulnerabilities, which assists the security administrators to evaluate the network security status with the vulnerabilities interactions.

7.3 Data and Methodology of the Present Study

Common Vulnerabilities and Exposures (CVE) and National Vulnerability Database (NVD) databases were utlized to collect each published operating systems vulnerability from 1999-2015. In this section, we show the procedures of employing the statistical models of the expected time for a hacker to exploit the network system and the minimum number of steps for a hacker to exploit the network with a very high probability, respectively.

7.3.1 Calculation of Expected Path Length

The expected time for the network system to be compromised was calculated using the CVE and NVD databases and the statistical model provided by Kaluarachchi and Tsokos [53]. Figure 7.3 shows the complete procedure.

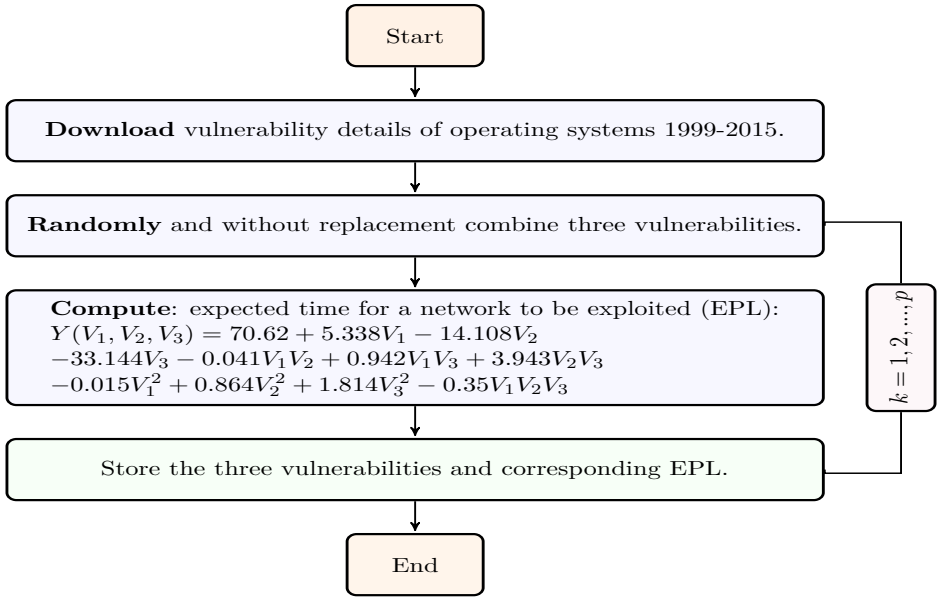


Figure 7.3.: Data collection procedure and computation of expected path length for a network to be exploited.

A sample of the resulting data that we have obtained and will be used in the present study is given in Table 7.5.

7.3.2 Calculation of Minimum Number of Steps

The minimum number of steps for a hacker to compromise the network system with a very high probability was calculated using the CVE and NVD databases and the statistical model provided by Kaluarachchi and Tsokos [53]. Figure 7.4 shows the complete procedure.

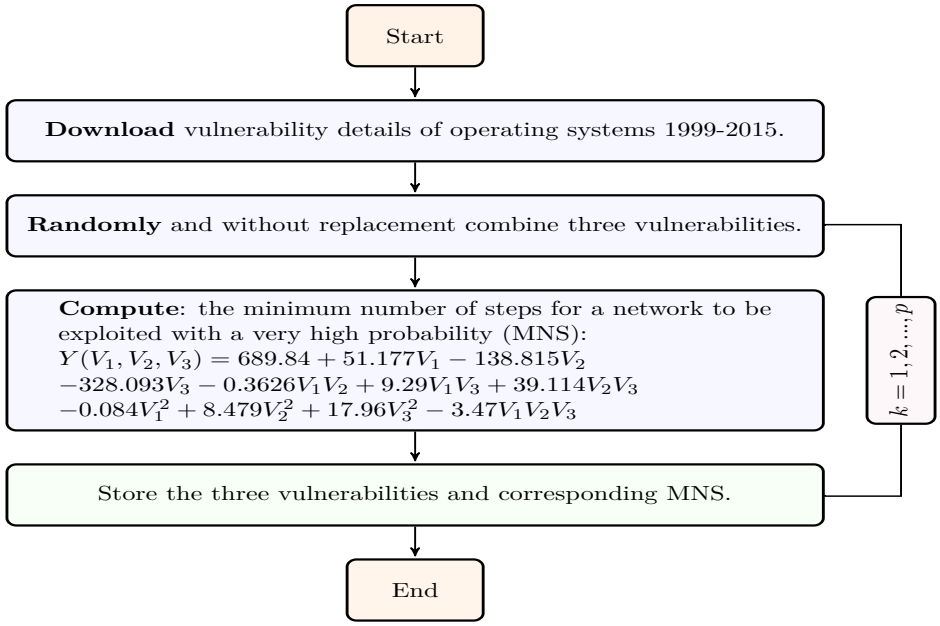


Figure 7.4.: Data collection procedure and computation of the minimum number of steps for a network to be exploited with a very high probability.

A sample of the resulting data that we have obtained and will be used in the present study is given in Table 7.5 below.

Table 7.5: A sample of sets of vulnerability scores, along with the estimates of the expected path length and minimum number of steps to hack the computer network system

Set	$CVSS_1$	$CVSS_2$	$CVSS_3$	EPL	MNS
1	7.5	5	5	7.24	57.38
2	4.3	4.3	5	0.15	91.54
3	9.3	4.3	5	10.86	359.19
4	10	4.9	10	37.97	1130.52
5	2.1	7.5	9.3	114.72	388.93
6	5	7.2	7.1	40.34	325.54
7	7.5	4.9	9.3	34.34	400.16
8	6.8	10	5.4	41.67	713.20
9	2.1	7.5	7.5	72.65	38.01
10	4.9	2.1	9.3	5.27	667.33
11	6.8	6.8	10	68.65	42.44
12	6.6	4	4.7	5.72	156.03
13	6.6	5.5	7.2	17.08	584.97
14	6.8	7.2	9.3	60.31	70.04
15	7.2	4.3	6.8	8.55	157.62
16	6.8	7.5	5	17.19	1003.03
17	4	7.1	10	102.15	338.12

7.3.3 Methodology

Parametric and non-parametric analysis involve identifying the probabilistic behavior of the expected path length for a hacker to compromise the network system, and the minimum number of steps it would take him/her to hack the network with a very high probability.

We proceed, if possible, to define the best probability density function (PDF) that characterize the probabilistic behavior of the hacking time. Anderson-Darling, Cramérvon-Mises, and Kolmogorov-Smirnov goodness of fit tests will be used to idenify the subject PDF. Each hypothesis test have a null hypothesis that the underlying probabilistic behavior of the data is explained by the target distribution, at 0.05 level of significance.

Sometimes, our attempts do not result in finding a well-defined PDF that fits the data in hand. In such cases, we look into alternative approaches, non-parametric approach, like the kernel density estimation.

Let x_1,x_2, ..., x_i represents independent and identical distributed random variables having a common probability density function (PDF), $f(x)$. The kernel density estimation of $f(x)$ is given by:

$$\hat{f}_n(x) = \frac{1}{nh}\sum_{i=1}^{n} K\Big(\frac{x - x_i}{h}\Big), \tag{7.3.3.1}$$

where $K(u)$ is the kernel function and h is a positive number called the bandwidth. The kernel function is usually required to meet the following conditions:

- $\int_{-\infty}^{+\infty}(K(u))du$ =1,
- $\int_{-\infty}^{+\infty} u \cdot (K(u))du$ =0,
  and,
- $\int_{-\infty}^{+\infty} u^2 \cdot (K(u))du \geq 0$.

Table 7.6 shows the most commonly used kernel functions.

The kernel density estimate of the cumulative density function for the given random variables with a common PDF, $f(x)$, is defined by:

$$\hat{F}_n(x) = \frac{1}{nh}\sum_{i=1}^{n} \int_{-\infty}^{y} K\Big(\frac{y - x_i}{h}\Big)dy. \tag{7.3.3.2}$$

Table 7.6: Most commonly used kernels

Kernel functions ($K(u)$)	Expression
Gaussian	$\frac{1}{\sqrt{2}} \exp\left(\frac{-u^2}{2}\right) I_R$
Epanechnikov	$\frac{3}{4}(1-u^2) I_{\{\mid u \mid \leq 1\}}$
Uniform	$\frac{1}{2} I_{\{\mid u \mid \leq 1\}}$
Triangular	$(1- \mid u \mid) I_{\{\mid u \mid \leq 1\}}$
Triweight	$\frac{35}{32}(1-u^2)^3 I_{\{\mid u \mid \leq 1\}}$
Biweight	$\frac{15}{16}(1-u^2)^2 I_{\{\mid u \mid \leq 1\}}$
Tricube	$\frac{70}{81}(1- \mid u \mid^3)^3 I_{\{\mid u \mid \leq 1\}}$
Cosine	$\frac{\pi}{4} \cos\left(\frac{\pi}{2} u\right)$

It is observed that kernel density estimation depends on the sample size, bandwidth, and the choice of the kernel function ($K(u)$). We show the selection procedure in the result section.

7.4 Results and Discussions

7.4.1 Parametric Modeling of the Expected Path Length

As described in section 7.3.1, the expected path length for the network system to be hacked was calculated using the statistical model provided by Kaluarachchi and Tsokos [53]. By the end of the calculation process, the total number of expected path length was $2,980$. Figure 7.5 shows a histogram of the data.

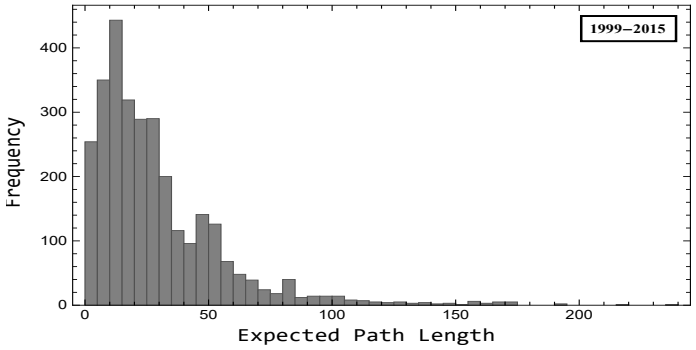


Figure 7.5.: Histogram of the expected path length of the network system to be compromised

The histogram of the expected path length data for hacking the network system indicates skewness to the right. After an extensive search for parametric probability distributions, the goodness of fit tests did not result in any significance finding for the best probability distribution to fit the expected path length data. Common data transformation techniques were also applied to the original data, but still we couldn't find a probability distribution that best fits the data.

Furthermore, we proceeded to apply the goodness of fit tests of the expected path length data against a mixture of parametric probability distributions. A mixture of Gamma (α,β) and Log-Normal (μ,σ) probability distributions was a good fit to the data. The Anderson-Darling and Cramérvon-Mises goodness of fit tests confirmed the fit of the mixture of the mentioned probability distributions, with p-values = 0.10 and 0.13, respectively.

The PDF of a mixture of the two probability distributions can be written by:

$$f(x) = \sum_{i=1}^{2} w_i \cdot f_i(x), \tag{7.4.1.1}$$

where:

- w_i: weights such that $w_i \geq 0$ and $\sum w_i$=1,

and

- $f_1(x)$ and $f_2(x)$ are PDFs of Gamma and LogNormal probability distributions, respectively.

Thus, the PDF of the mixture of Gamma and LogNormal probability distributions that approximately fit the expected path length data ($\hat{f}_{EPL}(x)$) for a hacker to compromise the network system is simplified and given by:

$$\hat{f}_{EPL}(x) = \begin{cases} \frac{e^{-\frac{x}{\beta}} w_1 \beta^{-\alpha} x^{\alpha-1}}{(w_1+w_2)\Gamma(\alpha)} + \frac{e^{-\frac{(\log(x)-\mu)^2}{2\sigma^2}} w_2}{\sqrt{2\pi}(w_1+w_2)\sigma x}, & x > 0 \\ 0, & \text{Otherwise} \end{cases}, \tag{7.4.1.2}$$

where α and β represents the shape and scale parameters of Gamma probability distribution, respectively, and μ and σ represents the mean and standard deviation of LogNormal probability distribution, respectively. Whereas w_1 and w_2 are the weighting parameters of the mixture of the two probability distributions. The cumulative density function (CDF) of the mixture of Gamma and LogNormal distributions is simplified and given by:

$$\hat{F}_{EPL}(x) = \begin{cases} \frac{w_2 \cdot \text{Erfc}\left(\frac{\mu - \log(x)}{\sqrt{2}\sigma}\right)}{2(w_1+w_2)} + \frac{w_1 Q\left(\alpha, 0, \frac{x}{\beta}\right)}{w_1+w_2}, & x > 0 \\ 0, & \text{Otherwise} \end{cases}, \tag{7.4.1.3}$$

where:

- Erfc() is the complementary error function; $\text{Erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-z^2} dz$,

and

- Q() is the regularized incomplete gamma function; $Q(\alpha, 0, \frac{x}{\beta}) = \frac{\gamma(\alpha, \frac{x}{\beta})}{\Gamma(\alpha)}$.

Table 7.7 presents the approximate maximum likelihood estimates (MLEs) of the $\hat{f}_{EPL}(x)$'s parameters.

Table 7.7: Approximate maximum likelihood estimates (MLEs) of the $\hat{f}_{EPL}(x)$'s parameters.

Parameter	MLE
w_1	0.16
w_2	0.84
α	1.59
β	5.08
μ	3.24
σ	0.75

Figure 7.6 shows the PDF of the mixture Gamma and LogNormal (Mixture PDF) probability distributions curve along with the histogram of the expected path length data.

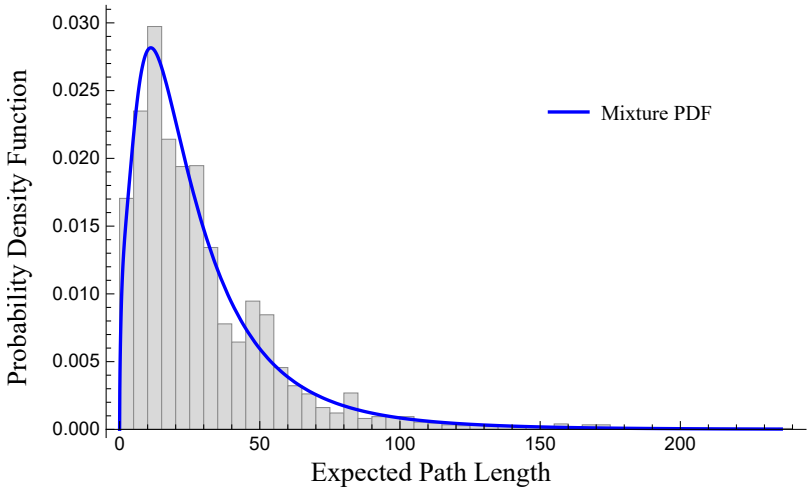


Figure 7.6.: Histogram of the the expected path length data, along with the mixture PDFs.

It can be seen that the mixture of Gamma and LogNormal probability distributions fit the expected path length data very well.

The defined PDF of the mixture of the subject probability distributions can assist us to answer some important questions in cybersecurity analysis. For example, what is the probability of the selected network system with a random set of vulnerabilities that will have an expected path length for an attacker to hack the network between 80 and 90 days (i.e., any time unit)? The answer is simply by integrating the $\hat{f}_{EPL}(x)$ from 80 to 90 after evaluating it at the approximate maximum likelihood estimates of the parameters in Table 7.7; namely α, β, μ, σ, w_1, and w_2, and given by:

$$\int_{80}^{90} 0.0134 \cdot e^{-0.197x} x^{0.592} + \frac{0.445 \cdot e^{-0.889(\log(x)-3.237)^2}}{x} dx = 0.014 \tag{7.4.1.4}$$

Therefore, the estimate of the required probability is $\hat{p} = 0.014$.

Figure 7.7 shows the CDF of the mixture Gamma and LogNormal (Mixture PDF) probability distributions curve of the estimated expected path length data.

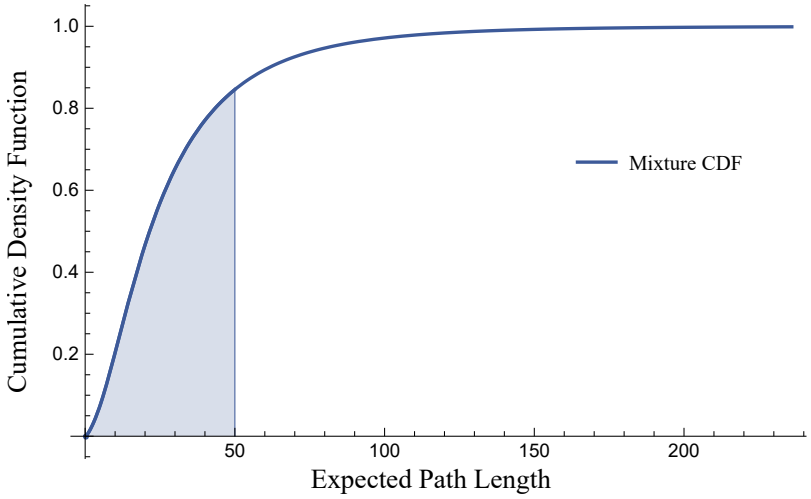


Figure 7.7.: The mixture cumulative density function of the expected path length data.

The CDF of the mixture of the subject probability distributions of the expected path length data is very useful in computing the cumulative probabilities. For example, the shaded area under the CDF curve (in Figure 7.7) represents approximately $\hat{p}$=0.846, which is the cumulative probability of the selected network system with a random set of vulnerabilities that will have an expected path length for an attacker to hack the network from the first time unit t=1 up to the 50^{th} time unit.

7.4.2 Non-Parametric Modeling The Expected Path Length

Parametric modeling of data at hand is a very powerful analysis compared to non-parametric modeling, especially when the underline probability distribution is correctly identified. To eliminate

any possibilities that the goodness of fit tests might misclassify the probability distribution, and to give cybersecurity specialists more options on modeling the expected path length data, we proceed to apply the kernel density estimation as non-parametric approach.

The kernel density estimation depends on the sample size, bandwidth, and the choice of the kernel function ($K(u)$). In this study, the optimal bandwidth (h^*) and kernel function were chosen to minimize the asymptotic mean integrated squared error (AMISE). The simplified form of the AMISE is reduced to:

$$AMISE\left(\hat{f}(x)\right) = \frac{C(K)}{n \cdot h} + (\frac{1}{4} \cdot h^4 \cdot k_2^2 \cdot R\left(f^{(2)}(x)\right)) \tag{7.4.2.1}$$

where:

- C(K)= $\int (K(u))^2 du$.
- n: sample size.
- h: bandwidth.
- $k_2 = \int_{-\infty}^{+\infty} u^2 \cdot K(u) du$.
- $f^{(2)}(x)$ is the second derivative of $f(x)$.
- $R(f^{(2)}(x))= \int (f^{(2)}(x))^2 dv$.
- $h^*=\left[\frac{C(K)}{k_2^2 \cdot R\left(f^{(2)}(\beta)\right)}\right]^{1/5} \cdot n^{-1/5}$ (Silverman [90]).

AMISE depends on four entities, namely kernel function, bandwidth, sample size, and the target density $f(x)$. We can control the bandwidth and the kernel function. By fixing the kernel function and using the optimal bandwidth (h^*), we obtain the optimal AMISE [64]:

$$AMISE_{optima} = \frac{5}{4} \cdot (\sqrt{k_2} \cdot C(K))^{\frac{4}{5}} \cdot C(f^{(2)})^{\frac{1}{5}} \cdot n^{\frac{-4}{5}}. \tag{7.4.2.2}$$

The optimal kernel function is then chosen so that $AMISE_{optimal}$ is the minimum. Particularly, by minimizing $(\sqrt{k_2} \cdot C(K))$ without knowing the target PDF $f(x)$. The Epanechnikov kernel function has the minimum value among other kernel functions (Table 7.6), which is given by:

$$K_{Epan.}(u) = \frac{3}{4}(1 - u^2), \qquad (\mid u \mid \leq 1). \tag{7.4.2.3}$$

Therefore, the kernel density estimates of the probability density function (PDF) and cumulative density function (CDF) of the expected path length data for the network system to be hacked, $\hat{f}_n(x)$ and $\hat{F}_n(x)$, respectively, are given by:

$$\begin{aligned} &\hat{f}_n(x) = \frac{1}{(2980*3.53)} \sum_{i=1}^{2980} K_{Epan.}\left(\frac{x-x_i}{3.53}\right), \\ &\text{and} \\ &\hat{F}_n(x) = \frac{1}{(2980*3.53)} \sum_{i=1}^{2980} \int_{-\infty}^{x} K_{Epan.}\left(\frac{x-v_i}{3.53}\right) dx. \end{aligned} \tag{7.4.2.4}$$

Figure 7.8 shows the histogram of the expected path length data for the network system to be hacked, along with curves of the mixture and kernel PDFs. There are $2,980$ expected path length computed.

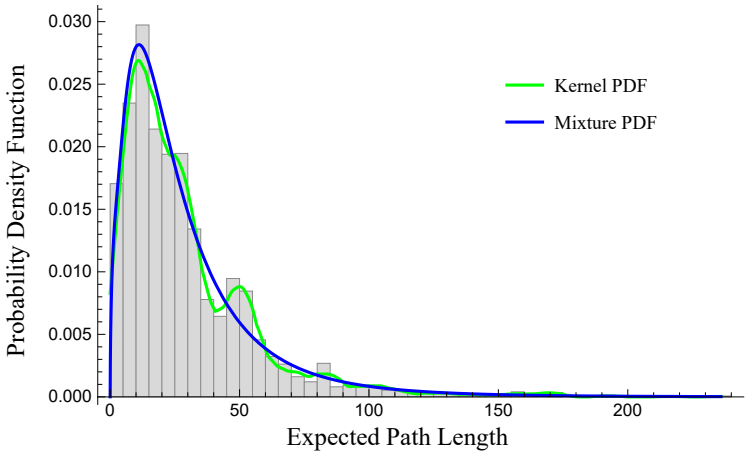


Figure 7.8.: Histogram of the expected path length, along with the mixture and kernel PDF curves.

We can see that both of the mixture and kernel PDFs are a very good fit to the expected path length data. The parametric modeling approach that resulted in defining the mixture of the subject probability distributions is highly recommended. The kernel density estimate approach gives the cybersecurity administrator another approach to better characterise the probabilistic behaviour of the expected path length data of their network system, especially when it is not possible to define a parametric probability distribution.

7.4.3 Parametric Modeling of The Minimum Number of Steps

As described in section 7.3.1, the minimum number of steps for the network system to be hacked with a very high probability was calculated using the statistical model provided by Kaluarachchi and Tsokos [53]. By the end of the calculation process, the total number of expected path length was $2,953$. Figure 7.9 shows the histogram of the data.

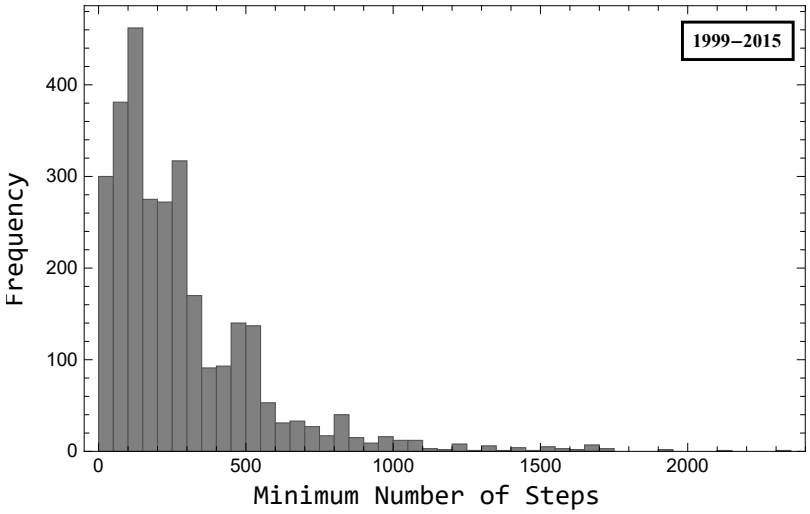


Figure 7.9.: Histogram of the minimum number of steps to hack the network system with a very high probability.

The histogram of the minimum number of steps data for hacking the network system with a very high probability indicates skewness to the right. After an extensive search in parametric probability distributions, the goodness of fit tests did not result in any significance in finding the best probability distribution to fit the minimum number of steps data (failure). Then, we proceeded to apply the goodness of fit tests of the minimum number of steps data against a mixture of parametric probability distributions. Again, a mixture of Gamma (α,β) and LogNormal (μ,σ) probability distributions was significant to fit the data. The Anderson-Darling and Cramérvon-Mises goodness of fit tests confirmed the fit of the mixture of mentioned distributions, with p-value $= 0.16$ and 0.19, respectively.

The PDF and CDF of the mixture of Gamma and LogNormal distributions that approximately fit the minimum number of steps data ($\hat{f}_{MNS}(x)$) for a hacker to compromise the network system with a very high probability is given by equations 7.4.1.2 and 7.4.1.3, respectively.

Table 7.8 presents the approximate maximum likelihood estimates (MLEs) of the $\hat{f}_{EPL}(x)$'s parameters.

Table 7.8: Maximum likelihood estimates (MLEs) of the $\hat{f}_{MNS}(x)$'s parameters.

Parameter	MLE
w_1	0.81
w_2	0.19
α	1.57
β	131.13
μ	6.23
σ	0.75

Figure 7.10 shows the PDF of the mixture Gamma and LogNormal (Mixture PDF) probability distributions curve along with the histogram of the minimum number of steps data.

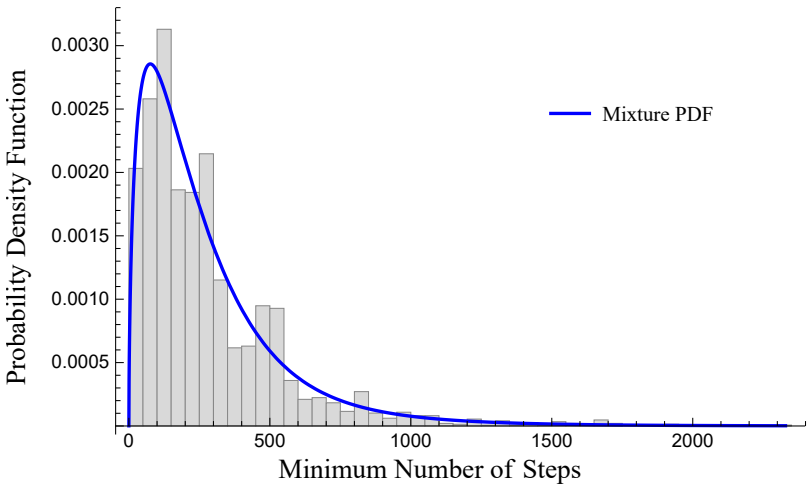


Figure 7.10.: Histogram of the the minimum number of steps data, along with the mixture PDF.

It can be seen that the mixture of Gamma and LogNormal probability distributions fit the minimum number of steps data well.

We can define the hack time of a computer network as a failure time. With reference to that, we consider the minimum number of steps (or any time unit such as days) for an attacker to hack the network system with a very high probability as the number of steps (or number of days) until a death or failure event occurs for the network with with a very high probability.

Thus, we can construct a reliability function of such computer network system. The reliability function of the minimum number of steps for an attacker to hack the network with a very high probability or the minimum number of time unit till a failure event of the computer network system

occurs with a very high probability is given by:

$$\hat{R}_{MNS}(x) = \int_x^{\infty} \hat{f}_{MNS}(z)dz = \begin{cases} \frac{w_1}{w_1+w_2} + \frac{w_2}{w_1+w_2} = 1, & x \leq 0\,, \\ \frac{w_2 \text{Erfc}\left(-\frac{\mu-\log(x)}{\sqrt{2}\sigma}\right)}{2(w_1+w_2)} + \frac{w_1 Q\left(\alpha, \frac{x}{\beta}\right)}{w_1+w_2}, & x > 0, \end{cases} \tag{7.4.3.1}$$

where:

- the probability of not being hacked past time 0 is $\frac{w_1}{w_1+w_2} + \frac{w_2}{w_1+w_2} = 1$,
- Erfc() is the complementary error function; $\text{Erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} e^{-z^2} dz$,

and

- Q() is the regularized incomplete gamma function; $Q(\alpha, \frac{x}{\beta}) = \frac{\Gamma(\alpha, \frac{x}{\beta})}{\Gamma(\alpha)}$.

The visualization of the reliability function of the minimum number of steps that an attacker needs to compromise the network or the time till the failure of the network system with a very high probability is given by Figure 7.11.

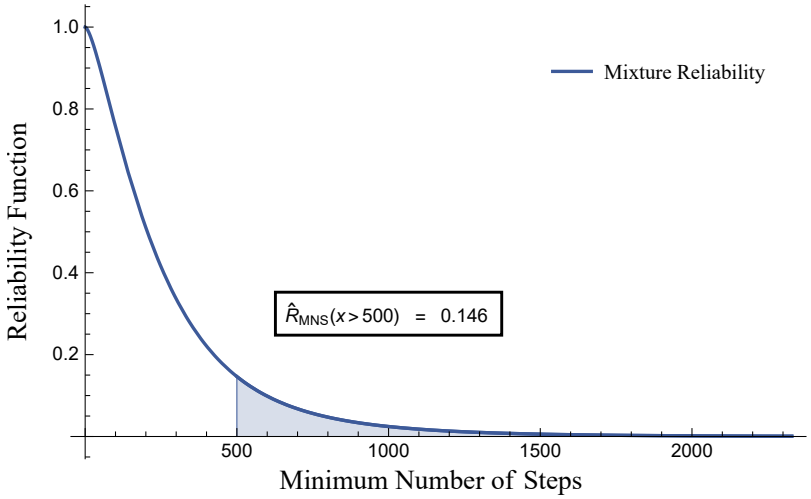


Figure 7.11.: Reliability function of the minimum number of steps to hack the network with a very high probability.

The reliability function of the mixture of the subject probability distributions is very useful to cybersecurity administrators. For example, the shaded area under the curve in Figure 7.11 represents the probability estimate ($\hat{R}_{MNS}(x > 500)$=0.146) of the selected network system with a random set of vulnerabilities that will not be hacked after 500 unit of time with very high probability. Again, this reliability function is for a given computer network system, however, the methodology and analysis are applicable to any desired computer network system.

7.4.4 Non-Parametric Modeling of the Minimum Number of Steps

The parametric data analysis is highly recommended when the underlying probability distribution is appropriately identified. In real world problems such as cybersecurity data, researchers might not be able to find the probability distribution that characterizes the probabilistic behaviour of their subject phenomenon. Therefore, we provide them with a secondary approach by employing the kernel density estimation (non-parametric modeling) of the minimum number of steps for an attacker to hack the network system with a very high probability.

Again, the minimum number of steps for an attacker to hack the network system with a very high probability can be defined as the minimum number of time unit till a failure event of the computer network system occurs with a very high probability.

The kernel density estimates of the probability density function (PDF) and cumulative density function (CDF) of the minimum number of steps data of hacking the network system with a very high probability , $\hat{f}_{MNS.n}(x)$ and $\hat{F}_{MNS.n}(x)$, respectively, are developed based on the Epanechnikov kernel function (optimal kernel) and the optimal bandwidth (h^*) [90].

$$\begin{aligned} &\hat{f}_{MNS.n}(x) = \tfrac{1}{(2953*35.2)} \textstyle\sum_{i=1}^{2953} K_{Epan.}\left(\tfrac{x-x_i}{35.2}\right), \\ &\text{and} \\ &\hat{F}_{MNS.n}(x) = \tfrac{1}{(2953*35.2)} \textstyle\sum_{i=1}^{2953} \int_{-\infty}^{x} K_{Epan.}\left(\tfrac{x-v_i}{35.2}\right)dx. \end{aligned} \tag{7.4.4.1}$$

Figure 7.12 shows the histogram of the minimum number of steps data of the network system to be hacked with a very high probability, along with curves of the kernel density estimate of the their PDF ($\hat{f}_{mns.n}(x)$) and the mixture PDF of Gamma (α,β) and LogNormal (μ,σ) probability distributions. There are $2,953$ minimum number of steps computed.

We can see that the kernel and mixture PDFs are a good fit to the the minimum number of steps data (Figure 7.12).

The kernel estimate of the reliability function of the minimum number of steps that an attacker needs to compromise the network with a very high probability ($\hat{R}_{MNS.n}(x)$) is given by:

$$\hat{R}_{MNS.n}(x) = 1 - \hat{F}_{MNS.n}(x) = 1 - (\frac{1}{(2953 * 35.2)} \sum_{i=1}^{2953} \int_{-\infty}^{x} K_{Epan.}\left(\frac{x - v_i}{35.2}\right)dx). \tag{7.4.4.2}$$

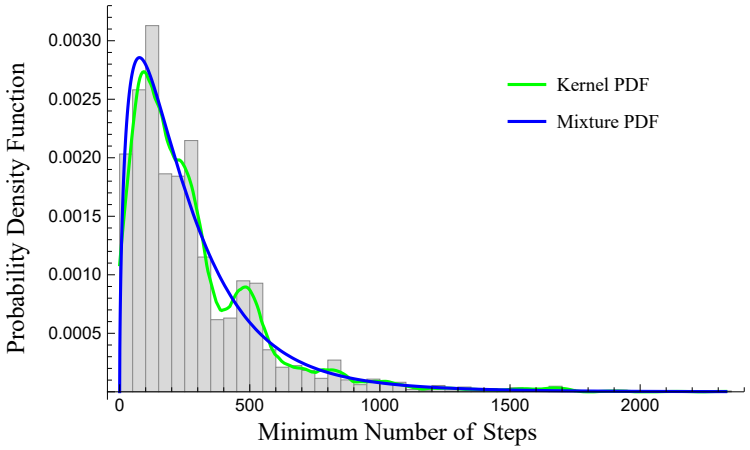


Figure 7.12.: Histogram of the minimum number of steps of hacking the network system with a very high probability, along with the mixture and kernel PDF curves.

Figure 7.13 shows the $\hat{R}_{mns.n}(x)$'s plot against the minimum number of steps.

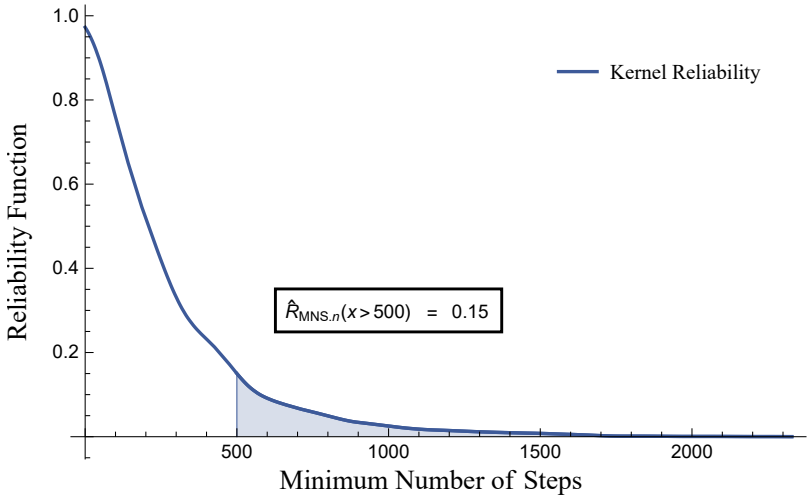


Figure 7.13.: kernel estimate of the reliability function of the minimum number of steps to hack the network with a very high probability.

The shaded area under the kernel reliability function curve in Figure 7.13 represents the probability estimate ($\hat{R}_{MNS}(x > 500)$=0.15) of the selected network system with a random set of vulnerabilities that will not be hacked after 500^{th} unit of time with very high probability. This reliability estimates is approximately same estimate resulted from the reliability function of the mixture of the subject probability distributions.

7.5 Contribution

In the present study, we developed a process of obtaining the reliability of a particular statistical network system that was previously studied by Kaluarachchi and Tsokos [53]. Moreover, we utilized two structured statistical models to both simulate expected path length and minimum number of steps data to hack a computer network system with a very high probability. Here we consider the minimum number of steps to hack our computer network system as the failure time, that is, if our network system is hacked, it fails us.

Then, we performed a parametric reliability analysis, and additionally a non-parametric reliability analysis. An IT manager can utilize the parametric reliability analysis of a given computer network system for maintenance purposes and other administrative management strategies. In cases where the computer network is more complicated and one can not do parametric analysis, then they can use the given procedure for non-parametric reliability analysis.

Although this analysis is data driven based on a computer network system of two systems and three vulnerabilities, it can be generalized to a computer network with 3,..,k systems. However, the analytical procedure and methodology remains the same.

Chapter 8

Future Research

We are considering future research in three areas. First is to apply Copula theory in healthcare systems. Second is to apply Power Law Process in financial systems. The third is the application of bootstrapping techniques in cybersecurity.

8.1 Applying Copula Theory in Healthcare Systems

Our primary objective is to apply the Copula theory in the investigation of Bayesian analysis in healthcare systems where parametric analysis indicates that the system we are analyzing has been characterized by a probability density function with two parameters that are dependent on each other, because they are estimated from the same information. Therefore, if we want to perform Bayesian analysis of a given healthcare system, we consider that both parameters are behaving as random variables, then we need to identify the bivariate probability distribution that characterizes probabilistic behaviors. Thus, once we obtain the Bayesian estimates we would expect them to be superior to the parametric estimates. Parametric estimates always provide the best results unless we are able to use Bayesian analysis by virtue of meeting the appropriate justifications. If we use real data in Bayesian analysis, we should be able to structure the bivariate probability distribution and thereby obtain more accurate estimates for both parameters behaving as random variables. There are a number of healthcare systems, particularly in the field of oncology research where this approach should be applicable.

8.2 Applying Power Law Process in Financial Systems

In the near future, we want to apply the Power Law Process in the evaluation of financial portfolio management performance by a financial advisor. One could evaluate the advisor as a function of time (i.e. week), and consider whether he makes money, loses money, or breaks even over consecutive time intervals. We could evaluate and track the portfolio behavior by calculating the

beta factor, which becomes the advisor's evaluation metric. Applying the PLP to a given financial system would not only be useful from a personal perspective, but anyone could use it to evaluate their portfolio managers.

8.3 Applying Bootstrapping Techniques in Cybersecurity

According to NetMarketShare, [89], as of March 2019, the operating system market shares are 87.45%, 9.73%, and 2.16% for Microsoft, Apple, and Linux computer operating systems, respectively. However, based on the National Vulnerability Database, Linux has the largest number of vulnerabilities, followed by Microsoft and Apple computer operating systems. Bootstapping techniques can be used to balance the number of vulnerabilities per computer operating system to a level where they are approximately equal. Performing comparison analysis and evaluating difference analysis of these three, now comparable operating systems vulnerabilities, will allow greater interpretation usefulness.

References

[1] M. S. Ahmed, E. Al-Shaer, and L. Khan. A novel quantitative approach for measuring network security. In *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*, pages 1957–1965, April 2008.

[2] Freeh Alenezi and Chris. Tsokos. Bayesian reliability analysis of the power law process with respect to the higgins-tsokos loss function for modeling software failure times, 2020.

[3] Freeh Alenezi and Chris Tsokos. Machine learning approach to predict computer operating systems vulnerabilities. In *2020 3rd International Conference on Computer Applications & Information Security (ICCAIS)*, page (in press). IEEE, (in press).

[4] Freeh N. Alenezi and Chris P. Tsokos. Bayesian reliability analysis of the power law process with respect to the higgins-tsokos loss function for modeling software failure times. *Submitted for publication*, dec 2018.

[5] Freeh N. Alenezi and Chris P Tsokos. The effectiveness of the squared error and higgins-tsokos loss functions on the bayesian reliability analysis of software failure times under the power law process. *Engineering*, 5(03):28, 2019.

[6] O. H. Alhazmi and Y. K. Malaiya. Quantitative vulnerability assessment of systems software. In *Annual Reliability and Maintainability Symposium, 2005. Proceedings.*, pages 615–620, Jan 2005.

[7] Luca Allodi and Fabio Massacci. Comparing vulnerability severity and exploits using case-control studies. *ACM Trans. Inf. Syst. Secur.*, 17(1):1:1–1:20, August 2014.

[8] Harold E Ascher. Weibull distribution vs'weibull process'. In *Annual Reliability and Maintainability Symposium, Philadelphia, Pa*, pages 426–431, 1981.

[9] Lee Bain and Max Engelhardt. *Statistical Analysis of Reliability and Life Testing Models.* Marcel-Dekker, New York, NY, USA, 1991.

[10] Shaul K Bar-Lev, Idit Lavi, and Benjamin Reiser. Bayesian inference for the power law process. *Annals of the Institute of Statistical Mathematics*, 44(4):623–639, 1992.

[11] James O Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer, 1985.

[12] GÃŠrard Biau. Analysis of a random forests model. *Journal of Machine Learning Research*, 13(Apr):1063–1095, 2012.

[13] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[14] Ronald Blank. *The Basics of Reliability*. SteinerBooks, 2004.

[15] Mehran Bozorgi, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker. Beyond heuristics: Learning to classify vulnerabilities and predict exploits. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 105–114, New York, NY, USA, 2010. ACM.

[16] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[17] Leo Breiman, Jerome Friedman, Richard Olshen, and Charles Stone. Classification and regression trees. wadsworth int. *Group*, 37(15):237–251, 1984.

[18] Benjamin L. Bullough, Anna K. Yanchenko, Christopher L. Smith, and Joseph R. Zipkin. Predicting exploitation of disclosed software vulnerabilities using open-source data. In *Proceedings of the 3rd ACM on International Workshop on Security And Privacy Analytics*, IWSPA '17, pages 45–53, New York, NY, USA, 2017. ACM.

[19] R Calabria, M Guida, and G Pulcini. Some modified maximum likelihood estimators for the weibull process. *Reliability Engineering & System Safety*, 23(1):51–58, 1988.

[20] R Calabria, M Guida, and G Pulcini. Bayes estimation of prediction intervals for a power law process. *Communications in Statistics-Theory and Methods*, 19(8):3023–3035, 1990.

[21] R. Calabria, M. Guida, and G. Pulcini. A bayes procedure for estimation of current system reliability. *IEEE Transactions on Reliability*, 41(4):616–620, 1992.

[22] George C Canavos and Chris P Tsokos. A study of an ordinary and empirical bayes approach to reliability estimation in the gamma life testing model. *Proceedings of IEEE Symposium on Reliability*, 1971.

[23] The MITRE Corporation. https://www.mitre.org.

[24] The MITRE Corporation. Common vulnerabilities and exposures. https://cve.mitre.org/index.html.

[25] Larry H. Crow. Reliability analysis for complex, repairable systems. *Proc. Twentieth Conf. Design of Experiments*, page 741–754, 1974.

[26] Larry H. Crow. Tracking reliability growth. *Proc. Twentieth Conf. Design of Experiments*, page 741–754, 1975.

[27] Daryl J Daley and David Vere-Jones. Conditional intensities and likelihoods. *An Introduction to the Theory of Point Processes: Volume I: Elementary Theory and Methods*, pages 211–287, 2003.

[28] Robert P Dobrow. *Introduction to stochastic processes with R.* John Wiley & Sons, 2016.

[29] J. T. Duane. Learning curve approach to reliability monitoring. *IEEE Transactions on Aerospace*, 2(2):563–566, 1964.

[30] Michel Edkrantz and Alan Said. Predicting cyber vulnerability exploits with machine learning. In *SCAI*, pages 48–57, 2015.

[31] A. Feutrill, D. Ranathunga, Y. Yarom, and M. Roughan. The effect of common vulnerability scoring system metrics on vulnerability exploit delay. In *2018 Sixth International Symposium on Computing and Networking (CANDAR)*, pages 1–10, Nov 2018.

[32] FIRST. Common vulnerability scoring system. https://www.first.org/cvss.

[33] FIRST. Forum of incident response and security teams. https://www.first.org.

[34] International Organization for Standardization. International organization for standardization. https://www.iso.org/home.html.

[35] Maurice J Frank. On the simultaneous associativity off (x, y) andx+y- f (x, y). *Aequationes mathematicae*, 19(1):194–226, 1979.

[36] Yoav Freund, Robert E Schapire, et al. Experiments with a new boosting algorithm. In *icml*, volume 96, pages 148–156. Citeseer, 1996.

[37] Daniel Funke. How common are data breaches — and what can you do about them? https://www.tampabay.com/news/business/2019/09/24/politifact-how-common-are-data-breaches-and-what-can-you-do-about-them/.

[38] Steven Furnell and David Emm. The abc of ransomware protection. *Computer Fraud & Security*, 2017(10):5 – 11, 2017.

[39] Amrit L. Goel and Kazu Okumoto. Time-dependent error-detection rate model for software reliability and other performance measures. *IEEE Transactions on Reliability*, R-28(3):206–211, aug 1979.

[40] Amrit L. Goel and Kazu Okumoto. Bayesian inference for the weibull process with applications to assessing software reliability growth and predicting software failures. *Proc. Sixteenth Symp. Interface*, R-28(3):206–211, 1984.

[41] Anatoliy Gorbenko, Alexander Romanovsky, Olga Tarasyuk, and Oleksandr Biloborodov. From analysing operating system vulnerabilities to designing multiversion intrusion–tolerance architectures. *IEEE Transactions on Reliability*, 2019.

[42] M Guida, R Calabria, and G Pulcini. Bayes inference for a non-homogeneous poisson process with power intensity law (reliability). *IEEE Transactions on Reliability*, 38(5):603–609, 1989.

[43] Gábor László Hajba. *Using Beautiful Soup*, pages 41–96. Apress, Berkeley, CA, 2018.

[44] Norman Harris. Repairable systems reliability: Modelling, inference, misconceptions and their causes, harold ascher and harry feingold, marcel dekker inc., 1984. *Quality and Reliability Engineering International*, 1(2):140–141, apr 1985.

[45] Bruce W. Turnbull Hasan Abu-Libdeh and Larry C. Clark. Estimation of failure intensity for the weibull process. *Biometrics*, 46:1017–1034, Dec 1990.

[46] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The elements of statistical learning: data mining, inference, and prediction, springer series in statistics, 2009.

[47] J.J. Higgins and C.P. Tsokos. A study of the effect of the loss function on bayes estimates of failure intensity, MTBF, and reliability. *Applied Mathematics and Computation*, 6(2):145–166, mar 1980.

[48] Mohammad Hossin and MN Sulaiman. A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process*, 5(2):1, 2015.

[49] Wolfram Research, Inc. Mathematica, Version 11.3. Champaign, IL, 2018.

[50] Sushil Jajodia and Steven Noel. Advanced cyber attack modeling analysis and visualization. Technical report, GEORGE MASON UNIV FAIRFAX VA, 2010.

[51] Harold Jeffreys. An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 186(1007):453–461, sep 1946.

[52] J. R. Jones. Estimating software vulnerabilities. *IEEE Security Privacy*, 5(4):28–32, July 2007.

[53] Pubudu Kalpani Kaluarachchi. Cybersecurity: Stochastic analysis and modelling of vulnerabilities to determine the network security and attackers behavior. *Graduate Theses and Dissertations.*, 2017.

[54] Pubudu Kalpani Kaluarachchi, Chris P Tsokos, and Sasith M Rajasooriya. Cybersecurity: a statistical predictive model for the expected path length. *Journal of information Security*, 7(03):112, 2016.

[55] Pubudu Kalpani Kaluarachchi, Chris P Tsokos, and Sasith M Rajasooriya. Cybersecurity: a statistical predictive model for the expected path length. *Journal of information Security*, 7(03):112, 2016.

[56] Le M. Kieu. *Analytical Modelling of Point Process and Application to Transportation.* Springer, New York, NY, USA, 2018.

[57] Phongphun Kijsanayothin. *Network security modeling with intelligent and complexity analysis.* PhD thesis, Texas Tech University, 2010.

[58] M. Kimura, T. Toyota, and S. Yamada. Economic analysis of software release problems with warranty cost and reliability requirement. *Reliability Engineering & System Safety*, 66(1):49–55, oct 1999.

[59] Nicolas Lachiche and Peter A Flach. Improving accuracy and cost of two-class and multi-class probabilistic classifiers using roc curves. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 416–423, 2003.

[60] Marco Raberto Enrico Scalas Linda Ponta, Mailan Trinh and Silvano Cincotti. Modeling non-stationarities in high-frequency financial time series. *Physica A: Statistical Mechanics and its Applications*, 521:173 – 196, May 2019.

[61] Liang Luo, Liudong Xing, and Gregory Levitin. Optimizing dynamic survivability and security of replicated data in cloud systems under co-residence attacks. *Reliability Engineering & System Safety*, sep 2018.

[62] Peter Mell, Karen Ann Kent, and Sasha Romanosky. *The common vulnerability scoring system (CVSS) and its applicability to federal agency systems.* Citeseer, 2007.

[63] Aditya Krishna Menon and Young Lee. Predicting short-term public transport demand via inhomogeneous poisson processes. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, pages 2207–2210, New York, NY, USA, 2017. ACM.

[64] Branko Miladinovic and Chris P. Tsokos. Ordinary, bayes, empirical bayes, and nonparametric reliability analysis for the modified gumbel failure model. *Nonlinear Analysis: Theory, Methods & Applications*, 71(12):e1426 – e1436, 2009.

[65] Carlos A. Molinares and Chris P. Tsokos. Bayesian reliability approach to the power law process with sensitive analysis to prior selection. *International Journal of Reliability, Quality and Safety Engineering*, 20(01):1350004, feb 2013.

[66] Edward F Moore and Claude E Shannon. Reliable circuits using less reliable relays. *Journal of the Franklin Institute*, 262(3):191–208, 1956.

[67] Yazdan Movahedi, Michel Cukier, Ambrose Andongabo, and Ilir Gashi. Cluster-based vulnerability assessment of operating systems and web browsers. *Computing*, sep 2018.

[68] Kartik Nayak, Daniel Marino, Petros Efstathopoulos, and Tudor Dumitraş. Some vulnerabilities are different than others. In *International Workshop on Recent Advances in Intrusion Detection*, pages 426–446. Springer, 2014.

[69] Roger B Nelsen. *An introduction to copulas*. Springer Science & Business Media, 2007.

[70] S. Neuhaus and T. Zimmermann. Security trend analysis with cve topic models. In *2010 IEEE 21st International Symposium on Software Reliability Engineering*, pages 111–120, Nov 2010.

[71] Steven Noel, Michael Jacobs, Pramod Kalapa, and Sushil Jajodia. Multiple coordinated views for network attack graphs. In *IEEE Workshop on Visualization for Computer Security, 2005.(VizSEC 05).*, pages 99–106. IEEE, 2005.

[72] National Institute of Standards and Technology. National vulnerability database. `https://nvd.nist.gov`.

[73] Serkan Özkan. Cve details. *Retrieved*, 16:2017, 2017.

[74] Robert Piessens and Maria Branders. A note on the optimal addition of abscissas to quadrature formulas of gauss and lobatto type. *Mathematics of Computation*, 28(125):135–139, 1974.

[75] Nawa Raj Pokhrel, Hansapani Rodrigo, and Chris P Tsokos. Cybersecurity: Time series predictive modeling of vulnerabilities of desktop operating system using linear and non-linear approach. *Journal of Information Security*, 8(04):362, 2017.

[76] Nawa Raj Pokhrel and Chris P Tsokos. Cybersecurity: A stochastic predictive model to determine overall network security risk using markovian process. *Journal of Information Security*, 8(02):91, 2017.

[77] G. Qi, G. Pan, S. Li, Z. Wu, D. Zhang, L. Sun, and L. T. Yang. How long a passenger waits for a vacant taxi – large-scale taxi trace mining for smart cities. In *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*, pages 1029–1036, Aug 2013.

[78] S. Radack and R. Kuhn. Managing security: The security content automation protocol. *IT Professional*, 13(1):9–11, Jan 2011.

[79] Sasith M Rajasooriya, Chris P Tsokos, and Pubudu Kalpani Kaluarachchi. Cyber security: Nonlinear stochastic models for predicting the exploitability. *Journal of information Security*, 8(02):125–140, 2017.

[80] Sasith Maduranga Rajasooriya. Cybersecurity: Probabilistic behavior of vulnerability and life cycle. *Graduate Theses and Dissertations.*, 2017.

[81] Leonard Richardson. Beautiful soup documentation. *April*, 2007.

[82] Steven E. Rigdon and Asit P. Basu. The effect of assuming a homogeneous poisson process when the true process is a power law process. *Journal of Quality Technology*, 22(2):111–117, 1989.

[83] Steven E. Rigdon and Asit P. Basu. The power law process: A model for the reliability of repairable systems. *Journal of Quality Technology*, 21(4):251–260, 1989.

[84] Steven E. Rigdon and Asit P. Basu. Estimating the intensity function of a power law process at the current time: time truncated case. *Communications in Statistics - Simulation and Computation*, 19(3):1079–1104, 1990.

[85] Carl Sabottke, Octavian Suciu, and Tudor Dumitraș. Vulnerability disclosure in the age of social media: exploiting twitter for predicting real-world exploits. In *24th {USENIX} Security Symposium ({USENIX} Security 15)*, pages 1041–1056, 2015.

[86] Hamid R. Sayarshad and Joseph Y. J. Chow. Survey and empirical evaluation of nonhomogeneous arrival process models with taxi data. *Journal of Advanced Transportation*, 50(7):1275–1294, 2016.

[87] Karen Scarfone and Peter Mell. An analysis of cvss version 2 vulnerability scoring. In *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*, ESEM '09, pages 516–525, Washington, DC, USA, 2009. IEEE Computer Society.

[88] Offensive Security. Exploit database. `https://www.exploit-db.com/`.

[89] Net Market Share. Operating system market share. `https://www.netmarketshare.com/`.

[90] Bernard. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, London, 1986.

[91] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427–437, 2009.

[92] Georgios Spanos, Lefteris Angelis, and Dimitrios Toloudis. Assessment of vulnerability severity using text mining. In *Proceedings of the 21st Pan-Hellenic Conference on Informatics*, PCI 2017, pages 49:1–49:6, New York, NY, USA, 2017. ACM.

[93] Anna But Tommi Härkänen and Jari Haukka. Non-parametric bayesian intensity model: Exploring time-to-event data on two time scales. *Scandinavian Journal of Statistics*, 44(3):798–814, June 2017.

[94] Chris Tsokos and I. Shimi. *The theory and applications of reliability with emphasis on Bayesian and nonparametric methods*, volume 2. Academic Press, 1977.

[95] Chris Tsokos and I. Shimi. *The theory and applications of reliability with emphasis on Bayesian and nonparametric methods*, volume 1. Academic Press, 1977.

[96] Chris P Tsokos. *Reliability Growth: Non-homogeneous Poisson Process*. CRD Press, 1995.

[97] Chris P. Tsokos and A. N. V. Rao. Estimation of failure intensity for the weibull process. *Reliab. Eng. Syst. Safety*, 45:271–275, 1994.

[98] Chris P. Tsokos and Yong Xu. Non-homogenous poisson process for evaluating stage i & II ductal breast cancer treatment. *Journal of Modern Applied Statistical Methods*, 10(2):646–655, nov 2011.

[99] Chensi Wu, Tao Wen, and Yuqing Zhang. A revised cvss-based system to improve the dispersion of vulnerability risk scores. *Science China Information Sciences*, 62(3):39102, Sep 2018.

[100] Shigeru Yamada, Mitsuru Ohba, and Shunji Osaki. S-shaped reliability growth modeling for software error detection. *IEEE Transactions on Reliability*, R-32(5):475–484, dec 1983.

[101] Y. Yamamoto, D. Miyamoto, and M. Nakayama. Text-mining approach for estimating vulnerability score. In *2015 4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)*, pages 67–73, Nov 2015.

[102] Dequan Yue, Guoxi Zhao, and Wuyi Yue. Analysis of a multi-server queueing-inventory system with non-homogeneous poisson arrivals. In *Proceedings of the 11th International Conference on Queueing Theory and Network Applications*, QTNA '16, pages 7:1–7:5, New York, NY, USA, 2016. ACM.

[103] Su Zhang, Doina Caragea, and Xinming Ou. An empirical study on using the national vulnerability database to predict software vulnerabilities. In Abdelkader Hameurlain, Stephen W. Liddle, Klaus-Dieter Schewe, and Xiaofang Zhou, editors, *Database and Expert Systems Applications*, pages 217–231, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[104] Serkan Özkan. Cve details. `https://www.cvedetails.com`.

Appendix A

Description of the research variables

The description of the research variables, provided by FIRST [a] and CVEdetails.[b] for CVSS v2. Along with the introduced variables

Variables	Categories	Description
Vulnerability ID	CVE	Each vulnerability is assigned a unique identification (ID) by the CVE Numbering Authorities (CNAs)
Vulnerability Level	low	0-3.9
	medium	4-6.9
	high	7-10
Confidentiality Impact	None	There is no impact to the confidentiality of the system.
	Partial	There is considerable informational disclosure.
	Complete	There is total information disclosure, resulting in all system files being revealed.
Integrity Impact	None	There is no impact to the integrity of the system.
	Partial	Modification of some system files or information is possible, but the attacker does not have control over what can be modified, or the scope of what the attacker can affect is limited.
	Complete	There is a total compromise of system integrity. There is a complete loss of system protection, resulting in the entire system being compromised.
Availability Impact	None	There is no impact to the availability of the system
	Partial	There is reduced performance or interruptions in resource availability
	Complete	There is a total shutdown of the affected resource. The attacker can render the resource completely unavailable
Access Complexity	Low	Specialized access conditions or extenuating circumstances do not exist. Very little knowledge or skill is required to exploit.
	Medium	The access conditions are somewhat specialized. Some preconditions must be satisfied to exploit.
	High	Specialized access conditions exist. It is hard to exploit and several special conditions must be satisfied to exploit
Authentication	Not required	Authentication is not required to exploit the vulnerability.
	Single system	The vulnerability requires an attacker to be logged into the system (such as at a command line or via a desktop session or web interface)
	Multiple systems	Exploiting the vulnerability requires that the attacker authenticate two or more times, even if the same credentials are used each time
Gained Access	None	
	User	
	Admin	
Summary	Text	The textual summary provided for each published vulnerability
Family	Apple	Apple operating systems (Table 5.1)
	Linux	Linux operating systems (Table 5.1)
	Microsoft	Microsoft operating systems (Table 5.1)
Days	Date	Number of days between the published and update dates of each vulnerability at the collection time
Year	Date	The publish year for each vulnerability
NummAttacks	Numeric	Number of attacks can be used to exploit each vulnerability
Frequency	Numeric	An indicator variable; whether the vulnerability affected single or multiple OSes.
Polarity	Numeric	The polarity of the textual summary of the vulnerability; a score ranges from $[-1, 1]$
Subjectivity	Numeric	The subjectivity of the textual summary of the vulnerability; a score ranges from $[0, 1]$.

[a] `https://www.first.org/cvss/v2/guide`
[b] `https://www.cvedetails.com`.

Appendix B

Copyright Permission

Alenezi, Freeh

From: eng <eng@scirp.org>
Sent: Wednesday, January 29, 2020 2:14 AM
To: Alenezi, Freeh
Subject: Publish your paper with open access journal Volume: 11, Issue 12 – 2019

Dear Dr. Freeh Alenezi,

Thanks for your email.

According to the publishing regulations, repeated publishing is not allowed.

However, it is ok to add references and then publish parts of your article.

Hope the above information is helpful.

If you have any interest in publishing your work here, please send it to my mailbox directly for quick submission.

Please feel free to contact me if you have any questions.

Sincerely,

Cindy Zhang
ENG Editorial Office
QQ: 1706777359
Email: eng@scirp.org

For New Paper submission, contact this email directly.

*Google-based Journal Impact Factor (GJIF) is **0.75**
*The **h5-index** released by Google Scholar is **15***

From: Alenezi, Freeh <fnalenezi@usf.edu>
Sent: Wednesday, January 29, 2020 0:15
To: eng
Subject: FW: Publish your paper with open access journal Volume: 11, Issue 12 – 2019

Hello,

I hope you are doing well. I have published an article with your journal and I am planning to publish again. I am going to submit my dissertation to USF in order to complete PhD requirements, but they have asked me to provide a permission from the publisher to use the content, again, in my dissertation. Please assist me on this matter. Thanks

1

Permission