symmetry

MDPI

Article

The Process Capability Index of Pareto Model under Progressive Type-II Censoring: Various Bayesian and Bootstrap Algorithms for Asymmetric Data

Rashad M. EL-Sagheer [1,2], Mahmoud El-Morshedy [3,4,*], Laila A. Al-Essa [5], Khaled M. Alqahtani [3] and Mohamed S. Eliwa [6,7,8]

1 Mathematics Department, Faculty of Science, Al-Azhar University, Naser City 11884, Egypt
2 High Institute of Computer and Management Information System, First Statement, New Cairo 11865, Egypt
3 Department of Mathematics, College of Science and Humanities in Al-Kharj, Prince Sattam bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia
4 Department of Mathematics, Faculty of Science, Mansoura University, Mansoura 35516, Egypt
5 Department of Mathematical Sciences, College of Science, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia
6 Department of Statistics and Operation Research, College of Science, Qassim University, Buraydah 51482, Saudi Arabia
7 Department of Statistics and Computer Science, Faculty of Science, Mansoura University, Mansoura 35516, Egypt
8 Section of Mathematics, International Telematic University Uninettuno, I-00186 Rome, Italy
* Correspondence: m.elmorshedy@psau.edu.sa

Abstract: It is agreed by industry experts that manufacturing processes are evaluated using quantitative indicators of units produced from this process. For example, the C_{py} process capability index is usually unknown and therefore estimated based on a sample drawn from the requested process. In this paper, C_{py} process capability index estimates were generated using two iterative methods and a Bayesian method of estimation based on stepwise controlled type II data from the Pareto model. In iterative methods, besides the traditional probability-based estimation, there are other competitive methods, known as bootstrap, which are alternative methods to the common probability method, especially in small samples. In the Bayesian method, we have applied the Gibbs sampling procedure with the help of the significant sampling technique. Moreover, the approximate and highest confidence intervals for the posterior intensity of C_{py} were also obtained. Massive simulation studies have been performed to evaluate the behavior of C_{py}. Ultimately, application to real-life data is seen to demonstrate the proposed methodology and its applicability.

Keywords: statistical model; process capability index; parametric bootstrap; simulation; statistics and numerical data; importance sampling technique

Citation: EL-Sagheer, R.M.; El-Morshedy, M.; Al-Essa, L.A.; Alqahtani, K.M.; Eliwa, M.S. The Process Capability Index of Pareto Model under Progressive Type-II Censoring: Various Bayesian and Bootstrap Algorithms for Asymmetric Data. *Symmetry* **2023**, *15*, 879. https://doi.org/10.3390/sym15040879

Academic Editors: Emilio Gómez Déniz, Héctor W. Gómez and Enrique Calderín-Ojeda

Received: 3 March 2023
Revised: 25 March 2023
Accepted: 31 March 2023
Published: 7 April 2023

1. Introduction

Process capability indicators are statistical measures of the inherent variability of a process and its ability to meet specifications. They are used to evaluate the quality and performance of parts and processes. Some common process capability indicators are: (i) C_p and C_{pk}: They show how capable a process is of meeting its specification limits, used with continuous data. C_p measures the potential capability of a process, while C_{pk} measures the actual capability during production. They are calculated as ratios of the specification width to the process width; (ii) Sigma: It is a capability estimate typically used with attribute data (i.e., with defect rates). It reflects the non-conformance rate of a process by expressing this performance in the form of a single number; and (iii) C_{pm}: It is a capability estimate that takes into account the deviation of the process mean from the target value. It is

used when the process output has a nominal or optimal value that is different from the midpoint of the specification limits. Process capability indicators can be applied to different fields and industries. For example, in agriculture, capability assessment is a method to assess land suitability for existing and potential agricultural and non-agricultural uses. It identifies possible physical, chemical and degradation constraints to land use on particular soils and landscapes. In medical sciences, the capability assessment can help measure the performance, precision and trueness of a process within the specifications and incorporate the loss function in capability analysis. Some examples of applications in the health field are: Assessment of quality control processes at a clinical laboratory chain; evaluation of well-being and social care interventions using the capability approach; measurement of process capability for electronic industries using a new index based on symmetric loss function. In engineering, the process capability indicators are appropriate and practical tools for asses ensuring that manufactured products conform to specifications. Moreover, product design information can be provided to reduce cost due to product failure. Therefore, quality control engineers use various statistical process methods to measure the capacity of a manufacturing process and to determine required specifications. In the literature, there are many compounds that can be applied to provide numerical measures of the ability of a process, see Montgomery [1] and Kane [2]. These vehicles/compounds include the process capability index/indicator (PCI), which provides a numerical indication of whether or not the production process is capable of producing products within specification limits. This specification is determined by the minimum specification limit L, the upper limit U, and the target value T. Many articles seem to introduce new indicators or examine the properties of existing indicators based on various assumptions. For example, Kotz and Johnson [3] presented an extensive bibliography on process capacity indicators. Kaminsky et al. [4] provided an overview of the use of product conforming process capacity indicators. Schneider et al. [5] discussed the uses PCI in the supplier certification process. The widest indices in particular of normally distributed characteristics are C_p, C_{pk}, C_{pm} and C_{pmk}. For more precise details and clarifications about these indices, we refer to Juran [6], Chan et al. [7], Kotz and Lovelace [8], Pearn et al. [9] and Gunter [10]. On the other hand, there are several PCIs that are valid for both nonnormal and typical output process characteristics, see Pearn and Chen [11,12], Clements [13] and Polansky [14].

In recent years, Maiti et al. [15] listed a new generic index C_{py} related directly or indirectly to most PCIs described in the past. Moreover, they include the characteristics of normal, continuous, abnormal as well as discrete quality and are defined as follows:

$$C_{py} = \frac{F(U) - F(L)}{F(UDL) - F(LDL)} = \frac{P}{P_0}, \tag{1}$$

where F is the distribution function of the quality characteristic X. L and U are the lower and upper limits of the specification, respectively. LDL and UDL are the desirable lower and upper limits, respectively. It can also be formulated in terms of tail probabilities as

$$C_{py} = \frac{1}{P_0}\left(1 - \left[\overline{F}(U) + F(L)\right]\right), \tag{2}$$

where P is the output of the process, and P_0 is the desired yield. For normal operation with

$$LDL = \mu - 3\sigma\,,\ UDL = \mu + 3\sigma\ \Rightarrow C_{py} = \frac{P}{0.9973}. \tag{3}$$

From (3) we observe that, if $P > P_0$, then $C_{py} > 1$, if $P = P_0$ (for a normal process, $P = 0.9973$), then $C_{py} = 1$, if $P < P_0$ then $C_{py} < 1$ and the value of $C_{py} \to 0$ as $P \to 0$. From the perspective of reliability research, survival analysis, and life test trials/experiments, it is usually not possible to observe all life spans of all products under test due to time constraints or other constraints such as money, material resources, mechanical or experimental difficulties, etc. Data collection. For this reason, reducing the total time and high cost of

testing is vital. In this type of experiment, units may fail or be removed prior to failure, which are used for future experiments. Thus, censored/controlled sampling may occur in practice. Nowadays, control methodologies in reliability analysis have several types that have been implemented in lifetime experiments. In practice, there are usually two random variables, the time and the number of failed units. This control charts strategy shows how the examiner imagines the experience based on a predetermined time. The first type of control regimen involves calculating a random number of units, which means that the exact time to stop the trial may be assumed. On the other hand, the type II control system involves a predetermined number of failure units and a random time. In both control schemes, units cannot be removed from the experiment until the final stage or failure of the number of units. However, the proposed methodology, incremental type II control/censorship (PT2C), has more flexibility than type II control by allowing units to be withdrawn from testing at different observed failure times. This approach has been explored in various studies such as Balakrishnan and Sandhu [16], Balakrishnan et al. [17], Fernindez et al. [18], Aslam et al. [19], Panahi [20], Wang et al. [21], Wang et al. [22], Luo et al. [23], Saberzadeh et al. [24], and Zhuang et al. [25]. [17], Fernindez et al. [18], Aslam et al. [19] and Panahi [20], Wang et al. [21], Wang et al. [22], Luo et al. [23], Saberzadeh et al. [24] and Zhuang et al. [25].

Let $X_{1:m:n}^{(R_1,\ldots,R_m)}, X_{2:m:n}^{(R_1,\ldots,R_m)}, \ldots, X_{m:m:n}^{(R_1,\ldots,R_m)}$ be the PT2C scheme described by Balakrishnan and Sandhu [16], see Figure 1. Probability function associated with PT2C with the scheme $R_i \geq 0, i = 1, 2, \ldots, m$ can be formulated as

$$L(\theta; \underline{x}) = A_{\circ} \prod_{i=1}^{m} f_X(x;\theta)[1 - F_X(x;\theta)]^{R_i},\ 0 < x_1 < \cdots < x_m < \infty, \tag{4}$$

where $A_{\circ} = n(n - R_1 - 1)\cdots(n - R_1 - R_2 - \cdots - R_{m-1} - m + 1)$.

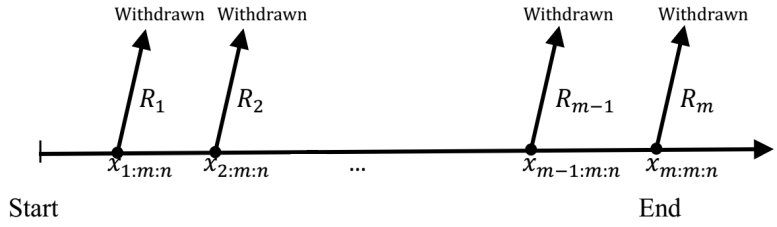

Figure 1. Description of PT2C scheme.

Several authors have discussed the statistical inference about PCIs based on censored samples from different lifetime distributions, see for example, Wu and Chiu [26], Hong et al. [27], Lee et al. [28], Saha et al. [29], Ahmadi and Doostparast [30,31] and EL-Sagheer et al. [32].

In this paper, our main objective is to estimate C_{py} using four different approaches, namely maximum likelihood, percentile bootstrap, bootstrap-t, and Bayes under PT2C samples from a Pareto distribution. Estimation techniques for model parameters are methods used to estimate the values of parameters within a model. These techniques generally use collected data and a model of the system to produce an estimate of the parameters of the system. There are a variety of techniques that can be used to estimate the parameters of a model. The most common techniques are maximum likelihood estimation (MLE) and Bayesian estimation. MLE is a method of estimating the parameters of a statistical model using the observations of a sample. It is based on the probability of the observations given the model parameters. Bayesian estimation is similar to MLE but takes into account the prior beliefs of the model parameters. It uses Bayesian probability theory to calculate the posterior probability of the model parameters given the observations. Other estimation techniques include the least squares method, which is used to estimate parameters of linear models, and the expectation-maximization algorithm, which is used to estimate parameters. The approximate confidence interval (ACI) of C_{py} is constructed based on the asymptotic normality of MLE. In the Bayesian framework, Markov chain Monte

Carlo (MCMC) techniques are applied due to the complexity of the system of equations to solve.

The rest of the paper is organized as follows. In Section 2, we have developed C_{py} for the Pareto model. Section 3 deals with MLE for the index C_{py} and asymptotic confidence interval. Bootstrap methods are investigated in Section 4. Bayes estimators using importance sampling technique under gamma priors with SELF are discussed in Section 5. In Section 6, a simulation study is conducted to compare the performance of the proposed techniques. Two real data sets are reported and discussed in Section 7. Finally, a brief conclusion is listed in Section 8.

2. The Index C_{py} for Pareto Model

Pareto [33] provided a lifetime model that contains two parameters as a model for the distribution of incomes called Pareto distribution (PD). The probability density function (PDF) and cumulative distribution function (CDF) of the PD can be proposed as

$$f_X(x;\lambda,\rho) = \frac{\rho\lambda^\rho}{(x+\lambda)^{\rho+1}}, \;\; x>0, \; \lambda,\rho>0, \tag{5}$$

and

$$F_X(x;\lambda,\rho) = 1-\left(\frac{\lambda}{x+\lambda}\right)^\rho, \;\; x>0, \; \lambda,\rho>0, \tag{6}$$

where λ and ρ are the scale and shape parameters, respectively. The importance of studying this distribution lies in the fact that it is a model applied for income distribution, and also as a loss model in property and accident insurance. This model has a heavy right-tail behavior, which makes it suitable for embedding large events in applications such as leverage loss pricing, for more details, we refer to Verma and Betti [34]. The PD is a very flexible model used for reliability and life testing studies, see Lawless [35] and Kim et al. [36]. Now, using (2), (5) and (6) the index C_{py}, where the quality characteristic X follows the PD, can be obtained as

$$C_{py} = \frac{1}{P_0}\left[\left(\frac{\lambda}{L+\lambda}\right)^\rho - \left(\frac{\lambda}{U+\lambda}\right)^\rho\right]. \tag{7}$$

3. The Maximum Likelihood Estimation (MLE) Approach

Distributions. It is The MLE is a widely used statistical technique used to obtain the unknown parameters of a given statistical model using data from a sample. MLE is based on the principle that the set of parameter values that maximize the probability of obtaining the observed data is the most likely value of the unknown parameter. MLE is a well-established and widely used method in many scientific fields because it is characterized by its ease in estimating the parameters of a particular population given a sample. It also has the advantage of being easy to implement computationally. The MLE technique is commonly used in applications such as regression analysis, hypothesis testing, and model fitting. In MLE, the parameters of interest are estimated from the sample data by finding the maximum likelihood estimates of the parameters. This is achieved by setting up the likelihood function, which is the probability of the observed data given the parameters, and then maximizing the likelihood function with respect to the model parameters. Let $X_1 < X_2 < \ldots < X_m$ be a PT2C sample from $PD(\lambda,\rho)$, with PDF and CDF as given in (5) and (6), respectively. Then, according to (4), the likelihood function (LF) of PT2C given as

$$L(\lambda,\rho|\underline{x}) = A_\circ \rho^m \prod_{i=1}^m \lambda^{\rho(R_i+1)}(x_i+\lambda)^{-[\rho(R_i+1)+1]}, \tag{8}$$

where C is defined in (4). The corresponding log-LF $\ell(\lambda,\rho|\underline{x}) = \ln L(\lambda,\rho|\underline{x})$ is

$$\ell(\lambda,\rho|\underline{x}) \propto m\ln\rho + \sum_{i=1}^m \rho(R_i+1)\ln\lambda - \sum_{i=1}^m [\rho(R_i+1)+1]\ln(x_i+\lambda). \tag{9}$$

Thus, the likelihood equations are

$$\sum_{i=1}^{m} \frac{\rho(R_i+1)}{\lambda} - \sum_{i=1}^{m} \frac{\rho(R_i+1)+1}{x_i+\lambda} = 0, \tag{10}$$

and

$$\frac{m}{\rho} + \sum_{i=1}^{m}(R_i+1)\ln\lambda - \sum_{i=1}^{m}(R_i+1)\ln(x_i+\lambda) = 0. \tag{11}$$

The Newton Raphson (NR) technique is applied to solve the previous system numerically. Once MLEs of λ and ρ denoted by $\hat{\lambda}$ and $\hat{\rho}$ are derived, the MLEs of the index C_{py} can be obtained as

$$\hat{C}_{py}^{ML} = \frac{1}{P_0}\left[\left(\frac{\hat{\lambda}}{L+\hat{\lambda}}\right)^{\hat{\rho}} - \left(\frac{\hat{\lambda}}{U+\hat{\lambda}}\right)^{\hat{\rho}}\right]. \tag{12}$$

The Fisher information matrix (FIM) is needed to create the ACIs for λ and ρ, see Cohen [37]

$$I_{\circ}(\lambda,\rho) = -E\begin{bmatrix} \frac{\partial^2\ell(\lambda,\rho|\underline{x})}{\partial\lambda^2} & \frac{\partial^2\ell(\lambda,\rho|\underline{x})}{\partial\lambda\partial\rho} \\ \frac{\partial^2\ell(\lambda,\rho|\underline{x})}{\partial\rho\partial\lambda} & \frac{\partial^2\ell(\lambda,\rho|\underline{x})}{\partial\rho^2} \end{bmatrix}. \tag{13}$$

Thus, the elements of the FIM can be reported as

$$\frac{\partial^2\ell(\lambda,\rho|\underline{x})}{\partial\lambda^2} = \sum_{i=1}^{m}\frac{\rho(R_i+1)+1}{(x_i+\lambda)^2} - \sum_{i=1}^{m}\frac{\rho(R_i+1)}{\lambda^2}, \tag{14}$$

$$\frac{\partial^2\ell(\lambda,\rho|\underline{x})}{\partial\lambda\partial\rho} = \frac{\partial^2\ell(\lambda,\rho|\underline{x})}{\partial\rho\partial\lambda} = \sum_{i=1}^{m}\frac{(R_i+1)}{\lambda} - \sum_{i=1}^{m}\frac{(R_i+1)}{x_i+\lambda}, \tag{15}$$

and

$$\frac{\partial^2\ell(\lambda,\rho|\underline{x})}{\partial\rho^2} = \frac{-m}{\rho^2}. \tag{16}$$

Further, the inverse of $I_{\circ}(\lambda,\rho)$ is given by

$$I_{\circ}^{-1}(\lambda,\rho) = \begin{bmatrix} var(\hat{\lambda}) & cov(\hat{\lambda},\hat{\rho}) \\ cov(\hat{\rho},\hat{\lambda}) & var(\hat{\rho}) \end{bmatrix}. \tag{17}$$

To calculate the ACI for $\hat{C}_{py}$, Greene [38] utilized the delta approach for this issue. Assume $\omega = (\frac{\partial C_{py}}{\partial\lambda}, \frac{\partial C_{py}}{\partial\rho})$. Then, the variance $\hat{C}_{py}$ can be listed as

$$var(\hat{C}_{py}) \simeq (\omega^T I_{\circ}^{-1}(\hat{\lambda},\hat{\rho})\omega)_{\downarrow(\theta=\hat{\theta})}, \tag{18}$$

where ω^T is the transpose matrix of ω. Thus, the $(1-\gamma)100\%$ ACI for $\hat{C}_{py}$ can be derived as

$$\left[\hat{C}_{py} \mp z_{\frac{\gamma}{2}}\sqrt{var(\hat{C}_{py})}\right]. \tag{19}$$

4. Bootstrap Methods

Bootstrap methods are a set of powerful statistical tools that are used in the field of data science and machine learning. These methods are used to identify potential sources of bias and variance in the results. Bootstrapping is also used to identify potential areas for improvement in the model. Bootstrapping has become increasingly popular in recent years due to its ability to provide a powerful and reliable way to estimate the parameters of a particular population given a sample. Bootstrapping involves the resampling of data from a given population in order to obtain a more accurate estimation of the true mean

and variance of the population. This process enables researchers to obtain a more reliable assessment of the accuracy of a given model. Furthermore, bootstrapping is used to identify potential sources of bias and variance in the results. By resampling the data multiple times, researchers can identify potential areas of improvement in the model. This is especially useful when the model is being used in prediction. Smoothing/bootstrap techniques are applied to provide more accurate confidence intervals, especially in small samples. The bootstrap methodology was originally developed by Efron [39], for more details see Davison and Hinkley [40]. In this segment, we consider the two most commonly utilized bootstrap approaches, namely, percentile bootstrap (boot-p) and bootstrap-t (boot-t) methods to obtain a more widely utilized CIs using the same steps mentioned in DiCiccio and Efron [41]. Figure 2 shows a simple description of the bootstrap procedure. The following algorithms describe the two types of bootstrap.

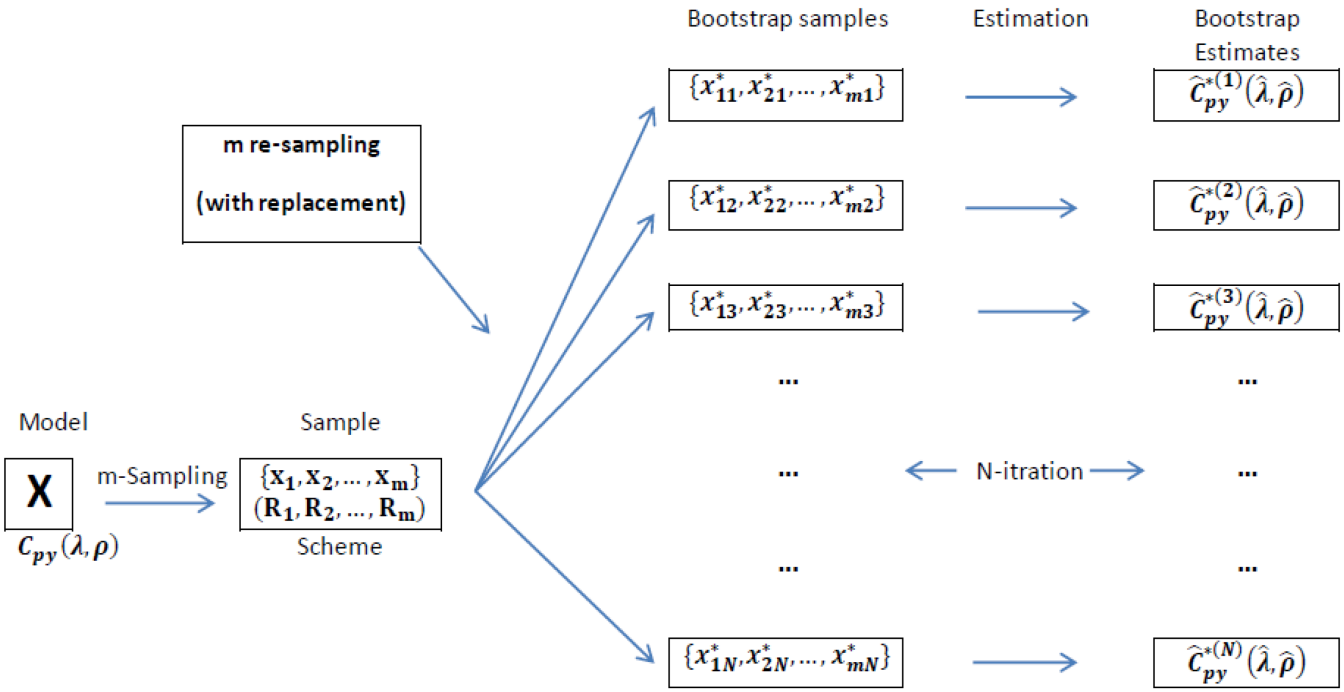

Figure 2. Bootstrap procedure.

4.1. Boot-P Algorithm

1. From the original data $x_1, x_2, \dots, x_m$ compute the MLEs of the unknown parameters λ and ρ, and denote the estimates as $\hat{\lambda}$ and $\hat{\rho}$.
2. Based on $\hat{\lambda}$ and $\hat{\rho}$, we generate independent bootstrap samples $x^*_1, x^*_2, \dots, x^*_m$ with the same values of R_i, $i = 1, 2, \dots, m$, from PD and then again we compute the parameters $(\hat{\lambda}, \hat{\rho})$, using MLE as well as $\hat{C}^*_{py}$ of C_{py}.
3. Repeat Step 2, N times to get a set of bootstrap samples of C_{py}, say $\hat{C}^{*(i)}_{py}$, $i = 1, 2, \dots, N$ and arrange $\hat{C}^{*(i)}_{py}$ in an ascending order $\left\{\hat{C}^{*[1]}_{py}, \hat{C}^{*[2]}_{py}, \dots, \hat{C}^{*[N]}_{py}\right\}$.
4. Let $\hat{C}^{*(\tau)}_{py}$ be the τ percentile of $\hat{C}^{*(k)}_{py}$ i.e$\hat{C}^{*(\tau)}_{py}$ is itself $\frac{1}{N}\sum_{k=1}^{N} I\left(\hat{C}^{*(k)}_{py} < \hat{C}^{*(\tau)}_{py}\right) = \tau$, $0 < \tau < 1$, where, $I(.)$ is the indicator function. Thus, the $(1-\gamma)100\%$ boot-p CIs of C_{py} is given by

$$\left[\hat{C}^{*\left(N\left(\frac{\gamma}{2}\right)\right)}_{py}, \hat{C}^{*\left(N\left(1-\frac{\gamma}{2}\right)\right)}_{py}\right] \tag{20}$$

4.2. Boot-T Algorithm

1. The same as boot-p algorithm.

2. Compute the t-statistics $T^*_{C_{py}} = \frac{\hat{C}^*_{py} - \hat{C}_{py}}{\sqrt{var(\hat{C}^*_{py})}}$, where $var(\hat{C}^*_{py})$ obtained using delta method in (18)
3. Repeat Step 2 and 3, N times, we have $T^{*(1)}_{C_{py}}, T^{*(2)}_{C_{py}}, \ldots, T^{*(N)}_{C_{py}}$.
4. Let $\Omega(x) = P\left(T^*_{C_{py}} \leq x\right)$ be the CDF of $T^*_{C_{py}}$. Define $\hat{C}^*_{py}(x) = \hat{C}_{py} + \Omega^{-1}(x)\sqrt{var(\hat{C}^*_{py})}$ for given x. Then, two-side $100(1-\gamma)\%$ boot-t CIs of C_{py} is given by

$$\left[\hat{C}^*_{py}\left(N\left(\frac{\gamma}{2}\right)\right), \hat{C}^*_{py}\left(N\left(1-\frac{\gamma}{2}\right)\right)\right] \tag{21}$$

5. Bayesian Estimation

Bayesian estimation is a powerful tool used to infer unknown parameters from measured data. It relies on Bayes' theorem, which is a theorem from probability theory that provides a means of updating the probability for a hypothesis as more evidence is collected. This approach has many advantages over traditional maximum likelihood estimation techniques due to its ability to incorporate prior knowledge into the estimation process. In addition, it has the capability to provide an estimate of the uncertainty associated with each parameter. Due to its many advantages, Bayesian estimation has become increasingly popular in a wide range of applications, including signal processing, machine learning, and artificial intelligence. For example, Bayesian estimation can be used to determine the parameters of a linear system given a set of observations. In machine learning, it can be used to determine the most likely hypothesis given a set of data. Similarly, it can be used in artificial intelligence for predicting the outcome of an event given a set of conditions. The family of gamma distributions is known to be flexible enough to cover a large variety of the experimenter's prior beliefs, see Kundu and Howlader [42]. Let that the parameters λ and ρ be stochastically independently distributed (SID) with conjugate gamma prior model. Suppose that $\lambda \sim$Gamma(a_1, b_1) and $\rho \sim$Gamma(a_2, b_2) and that these two priors are independent. Consequently, the joint prior density of λ and ρ can be written as

$$\pi(\lambda, \rho) \propto \lambda^{a_1-1}\rho^{a_2-1}e^{-(b_1\lambda + b_2\rho)},\ \lambda > 0,\ \rho > 0, \tag{22}$$

where a_i and b_i , $i = 1, 2$ are the hyper-parameters. Subsequently, via Bayes' theorem, the joint posterior density function of λ and ρ for given data is

$$\begin{aligned} \pi^*(\lambda, \rho|\underline{x}) &= \frac{L(\lambda,\rho|\underline{x})\pi(\lambda,\rho)}{\int_0^\infty \int_0^\infty L(\lambda,\rho|\underline{x})\pi(\lambda,\rho)d\lambda d\rho} \\ &\propto \lambda^{a_1-1}\rho^{m+a_2-1}e^{-b_1\lambda}\left[\prod_{i=1}^{m}\frac{\lambda^\rho}{x_i+\lambda}\right] \\ &\quad \times \exp\left\{-\rho\left[b_2 - \sum_{i=1}^{m} R_i \ln\lambda + \sum_{i=1}^{m}(R_i+1)\ln(x_i+\lambda)\right]\right\} \\ &\propto \pi^*_1(\lambda|\rho, \underline{x})\pi^*_2(\rho|\lambda, \underline{x})h(\lambda, \rho|\underline{x}). \end{aligned} \tag{23}$$

where

$$\pi^*_1(\lambda|\rho, \underline{x}) \propto \frac{\lambda^{a_1-1}e^{-b_1\lambda}}{\left(b_2 - \sum_{i=1}^{m} R_i \ln\lambda + \sum_{i=1}^{m}(R_i+1)\ln(x_i+\lambda)\right)^{m+a_2}}, \tag{24}$$

$$\pi^*_2(\rho|\lambda, \underline{x}) \sim Gamma\left(m + a_2, b_2 - \sum_{i=1}^{m} R_i \ln\lambda + \sum_{i=1}^{m}(R_i+1)\ln(x_i+\lambda)\right), \tag{25}$$

and

$$h(\lambda, \rho|\underline{x}) = \prod_{i=1}^{m}\frac{\lambda^\rho}{x_i+\lambda}. \tag{26}$$

The SELF can be written as

$$L_{SE}(\eta, \hat{\eta}) = (\hat{\eta} - \eta)^2, \tag{27}$$

where, $\hat{\eta}$ is the estimation of η. Thus, the Bayesian estimation of η under SELF is $\hat{\eta} = E_{\eta}(\eta|\underline{x})$. Then, Bayes estimate of $\varphi(\lambda, \rho)$ using SELF is computed by

$$\hat{\varphi}_{BS}(\lambda, \rho) = \frac{\int_0^\infty \int_0^\infty \varphi(\lambda, \rho) L(\lambda, \rho|\underline{x}) \pi(\lambda, \rho) d\lambda d\rho}{\int_0^\infty \int_0^\infty L(\lambda, \rho|\underline{x}) \pi(\lambda, \rho) d\lambda d\rho}. \tag{28}$$

Obviously, estimators of λ and ρ under SELF can be obtained by the importance sampling method.

Importance Sampling Procedure (ISP)

The ISP is one of the famous MCMC techniques. Also, it is consider an effective approach to attain Bayes estimates for $\varphi(\lambda, \rho)$. Moreover, the associated higher posterior density (HPD) intervals can be constructed through this method under PT2C data. MCMC is an important technique in the field of statistical computing. It is a powerful tool that can be used to sample from a given probability distribution, allowing for the estimation of a variety of parameters. The Markov chain, a collection of random variables that adheres to the Markov property, which claims that a system's future state is solely reliant on its present state, is the foundation of MCMC. MCMC algorithms use the Markov chain to sample from a target probability distribution and estimate various parameters. In particular, they are used to compute expectations, like the mean and variance, of a given model. The MCMC algorithm works by constructing a Markov chain whose state space is equal to the space of possible values of the target probability distribution. Starting from an initial state, the algorithm iteratively generates a sequence of states which are accepted or rejected based on the value of the target distribution at each state. As mentioned previously that $\pi_2^*(\rho|\lambda, \underline{x})$ is the PDF of a gamma distribution and, therefore, samples of ρ can be easily generated using any gamma generating routine. The density function $\pi_1^*(\lambda|\rho, \underline{x})$ cannot be reduced analytically to a well-known distribution, and thus cannot be directly sampled by standard methods. Through the ISP, we extract the sample from $\pi_1^*(\lambda|\rho, \underline{x})$ and $\pi_2^*(\rho|\lambda, \underline{x})$ and attain the Bayes estimates of (λ, ρ) and the corresponding estimator of the index C_{py}. The ISP approach is described as follows.

(1) Begin with initial guess value $\{\lambda^0, \rho^0\}$.
(2) Put $j = 1$,
(3) Generate $\lambda^{(j)}$ from $\pi_1^*(\lambda|\rho, \underline{x})$ utilizing the way/methodology reported via Metropolis et al. [43] with the $N((\lambda^{(j-1)}, \sigma^2)$.
(4) Generate $\rho^{(j)}$ from $\pi_2^*(\rho|\lambda, \underline{x})$.
(5) Put $j = j + 1$.
(6) Repeat Steps 2–5, for N times and simulate the sequence of samples of (λ_1, ρ_1), (λ_2, ρ_2), $\ldots$, (λ_N, ρ_N).
(7) The Bayesian estimate of $\varphi(\lambda, \rho)$ can be calculated by

$$\hat{\varphi}(\lambda, \rho) = \frac{\frac{1}{N-N_0} \sum_{i=N_0+1}^{N} \varphi(\lambda_i, \rho_i) h(\lambda_i, \rho_i|\underline{x})}{\frac{1}{N-N_0} \sum_{i=N_0+1}^{N} h(\lambda_i, \rho_i|\underline{x})}, \tag{29}$$

where N_0 is the burn-in-period of MCMC.

(8) Using the above sequence in Step 6 and 7, we can obtain the sequence $C_{py1}, C_{py2}, \ldots, C_{pyN}$, and then

$$\hat{C}_{py}^{MCMC} = E(C_{py}|\underline{x}) \approx \frac{1}{N-N_0} \sum_{k=N_0+1}^{N} C_{pyk}. \tag{30}$$

To compute the CRIs of C_{py}, sort $C_{py}^{(j)}, j = N_0+1, N_0+2, \ldots, N$ in ascending order as $\left\{C_{py}^{(1)} < C_{py}^{(2)} < \ldots < C_{py}^{(N)}\right\}$. Then, the $100(1-\gamma)\%$ symmetric CRIs of C_{py} can be obtained by

$$\left[C_{py(N\left(\frac{\gamma}{2}\right))}, C_{py(N\left(1-\frac{\gamma}{2}\right))}\right]. \tag{31}$$

6. Simulation Study and Discussion

In this segment, a lot of simulation experiments are performed to evaluate and test the performance of the four estimation techniques (ML, boot-p, boot-t, MCMC) by Monte Carlo simulations of the index C_{py} for PD. Point estimation is evaluated by mean squared errors (MSE), while interval estimation is evaluated based on mean lengths (ALs) and coverage probability (CPs) calculated as the number of CIs that covered true values divided by 10,000. Here, for the simulation study, we considered a different setting of parameter values such as $(\lambda, \rho) = (1, 0.085)$, $(1.5, 0.22)$, $(2.2, 0.5)$, $(15, 3)$ and $(35, 4.19)$ with the (L, U) as $(1.05, 33)$ and $P_0 = 0.95$. Then, the true values (TV) of C_{py} are evaluated to be 0.210313, 0.40858, 0.60299, 0.82715 and 0.864893, respectively.

For Bayesian computation: Bayes estimates and the HPD credible intervals (CRIs) are computed based on $N = 12{,}000$ MCMC samples and discard the first values $N_0 = 2000$ as "burn-in" with the hyper-parameters $(a_1, b_1) = (3, 2)$ and $(a_2, b_2) = (1, 1)$, respectively. For comparison purposes, we consider $(n, m) = (30, 15)$, $(30, 20)$, $(50, 30)$, $(50, 40)$, $(80, 55)$, $(80, 70)$, $(100, 80)$ and $(100, 90)$. For all the combinations of sample size, three various censoring schemes (CS) are determined. For simplicity, we abbreviate the censoring schemes. For example, (1, 1, 1, 1, 0, 0, 0) is represented as $(1^*4, 0^*3)$. Toward this end, we considered the various PT2C schemes which are listed in Table 1. The simulation results are listed in Tables 2–11, according to which we note the following:

1. It should be observed that the MSEs and average interval lengths drop from Tables 2–11 as sample sizes (n, m) rise.
2. The first scheme is the best strategy in terms of having lesser MSEs and ALs for fixed sample sizes and observed failures.
3. As C_{py} actual value rises, ALs also decrease in parallel.
4. As we predicted, Bayes estimates for C_{py} have the smallest MSEs and shortest ALs. Therefore, Bayes estimates outperform MLEs and bootstrap techniques.
5. In terms of MSEs and ALs, bootstrap techniques outperform the ML approach. Additionally, boot-t outperforms boot-p.
6. The estimates from the ML, bootstrap, and Bayesian approaches are all extremely close, and the CP values for the ACI are very high (around 0.95). The highest CPs are also found in the Bayesian CRIs.
7. In general, if prior knowledge about the problem under study is available, the Bayesian strategy used in conjunction with the significant sampling procedure is the optimum estimation approach.
8. Finally, we may say that the given inference methods produce reliable outcomes.

Table 1. Various PT2C schemes.

(n,m)	CS	$(R_1,R_2,\ldots,R_m)$	(n,m)	CS	$(R_1,R_2,\ldots,R_m)$
$(30,15)$	1	$(15,0^*14)$	$(30,20)$	1	$(10,0^*19)$
	2	$(0^*5,3^*5,0^*5)$		2	$(0^*5,1^*10,0^*5)$
	3	$(0^*14,15)$		3	$(0^*19,10)$
$(50,30)$	1	$(20,0^*29)$	$(50,40)$	1	$(10,0^*39)$
	2	$(0^*10,2^*10,0^*10)$		2	$(0^*15,1^*10,0^*15)$
	3	$(0^*29,20)$		3	$(0^*39,10)$
$(80,55)$	1	$(25,0^*54)$	$(80,70)$	1	$(10,0^*69)$
	2	$(0^*25,5^*5,0^*25)$		2	$(0^*30,1^*10,0^*30)$
	3	$(0^*54,25)$		3	$(0^*69,10)$
$(100,80)$	1	$(20,0^*79)$	$(100,90)$	1	$(10,0^*89)$
	2	$(0^*30,1^*20,0^*30)$		2	$(0^*40,1^*10,0^*40)$
	3	$(0^*79,20)$		3	$(0^*89,10)$
$(150,100)$	1	$(50,0^*99)$			
	2	$(0^*45,5^*10,0^*45)$			
	3	$(0^*99,50)$			

Table 2. MSE of C_{py} using the different estimation methods when $(\lambda,\rho)=(1,0.085)$.

(n,m)	TV of C_{py}	SC.	MLE	boot-p	boot-t	Bayesian
$(30,15)$	0.2103132	1	0.0172172	0.0175135	0.0155224	0.0132312
		2	0.0183546	0.0179924	0.0166452	0.0139874
		3	0.0193215	0.0186954	0.0176584	0.0153625
$(30,20)$		1	0.0054766	0.0049875	0.0042587	0.0039451
		2	0.0061254	0.0058632	0.0055634	0.0044635
		3	0.0066847	0.0062417	0.0059635	0.0049963
$(50,30)$		1	0.0045632	0.0043581	0.0039457	0.0035145
		2	0.0048635	0.0046258	0.0041996	0.0037657
		3	0.0051329	0.0049935	0.0046573	0.0041562
$(50,40)$		1	0.0041635	0.0039847	0.0037542	0.0034421
		2	0.0044324	0.0042968	0.0040584	0.0038652
		3	0.0049963	0.0046852	0.0044653	0.0042996
$(80,55)$		1	0.0033254	0.0032417	0.0029963	0.0027547
		2	0.0035416	0.0034658	0.0031547	0.0029984
		3	0.0036991	0.0035987	0.0034758	0.0032845
$(80,70)$		1	0.0029254	0.0028365	0.0026954	0.0025132
		2	0.0032546	0.0031659	0.0029638	0.0027653
		3	0.0033947	0.0032457	0.0032014	0.0030014
$(100,80)$		1	0.0023014	0.0022110	0.0020365	0.0019653
		2	0.0025615	0.0024635	0.0022874	0.0021047
		3	0.0026874	0.0026001	0.0024638	0.0022746
$(100,90)$		1	0.0015683	0.0013987	0.0012745	0.0011118
		2	0.0017654	0.0016542	0.0015326	0.0013652
		3	0.0018975	0.0017654	0.0016854	0.0014867
$(150,100)$		1	0.0011354	0.0010257	0.0009979	0.0009745
		2	0.0012978	0.0011564	0.0010365	0.0099997
		3	0.0013568	0.0012456	0.0011356	0.0107764

Table 3. MSE of C_{py} using the different estimation methods when $(\lambda, \rho) = (1.5, 0.22)$.

(n, m)	TV of C_{py}	SC.	MLE	boot-p	boot-t	Bayesian
$(30, 15)$	0.4085856	1	0.0212123	0.0195125	0.0175264	0.0162325
		2	0.0264525	0.0246536	0.0221633	0.0198365
		3	0.0337614	0.0326624	0.0314570	0.0271542
$(30, 20)$		1	0.0075258	0.0075145	0.0073406	0.0071522
		2	0.0088932	0.0087556	0.0085236	0.0083435
		3	0.0094812	0.0092432	0.0090358	0.0088642
$(50, 30)$		1	0.0049747	0.0048540	0.0045347	0.0043346
		2	0.0053435	0.0053642	0.0051166	0.0048257
		3	0.0058595	0.0057715	0.0055055	0.0052369
$(50, 40)$		1	0.0044812	0.0044836	0.0042554	0.0041478
		2	0.0047955	0.0047932	0.0045536	0.0043724
		3	0.0527245	0.0050462	0.0048712	0.0046211
$(80, 55)$		1	0.0037569	0.0036545	0.0035874	0.0033108
		2	0.0039432	0.0038555	0.0036456	0.0035348
		3	0.0045347	0.0045663	0.0042532	0.0041570
$(80, 70)$		1	0.0032258	0.0032224	0.0030446	0.0028666
		2	0.0035132	0.0035315	0.0032245	0.0031247
		3	0.0039636	0.0038133	0.0035477	0.0034109
$(100, 80)$		1	0.0025725	0.0024242	0.0022355	0.0021296
		2	0.0028574	0.0027302	0.0025266	0.0024309
		3	0.0032232	0.0032510	0.0029178	0.0027510
$(100, 90)$		1	0.0018445	0.0018323	0.0017245	0.0015655
		2	0.0021366	0.0021103	0.0020725	0.0019514
		3	0.0024441	0.0023970	0.0022112	0.0021723
$(150, 100)$		1	0.0014568	0.0013657	0.0012368	0.0010963
		2	0.0016478	0.0015639	0.0014698	0.0012362
		3	0.0019112	0.0018365	0.0017546	0.0013996

Table 4. MSE of C_{py} using the different estimation methods when $(\lambda, \rho) = (2.2, 0.5)$.

(n, m)	TV of C_{py}	SC.	MLE	boot-p	boot-t	Bayesian
$(30, 15)$	0.6029925	1	0.0324525	0.0315845	0.0297822	0.0255457
		2	0.0411945	0.0415863	0.0373678	0.0295654
		3	0.0522363	0.0525492	0.0445774	0.0365944
$(30, 20)$		1	0.0125425	0.0112381	0.0090655	0.0083535
		2	0.0237436	0.0216570	0.0135863	0.0092557
		3	0.0354547	0.0335693	0.0294325	0.0151985
$(50, 30)$		1	0.0094269	0.0092583	0.0085511	0.0079944
		2	0.0113509	0.0104575	0.0091347	0.0085269
		3	0.0265633	0.0235745	0.0156455	0.0118878
$(50, 40)$		1	0.0076510	0.0075377	0.0071363	0.0068259
		2	0.0084325	0.0081525	0.0079545	0.0074674
		3	0.0092922	0.0089740	0.0087944	0.0082845
$(80, 55)$		1	0.0055254	0.0053633	0.0049556	0.0047475
		2	0.0061436	0.0059642	0.0055663	0.0052263
		3	0.0068758	0.0067122	0.0064545	0.0059344
$(80, 70)$		1	0.0034566	0.0033745	0.0029877	0.0021966
		2	0.0042570	0.0042488	0.0031569	0.0028578
		3	0.0054312	0.0055120	0.0042488	0.0036344

Table 4. *Cont.*

(n, m)	TV of C_{py}	SC.	MLE	boot-p	boot-t	Bayesian
$(100, 80)$		1	0.0013644	0.0012923	0.0010189	0.0009976
		2	0.0015765	0.0014956	0.0012856	0.0011566
		3	0.0018978	0.0017689	0.0015223	0.0013445
$(100, 90)$		1	0.0011109	0.0010578	0.0009578	0.0009133
		2	0.0013577	0.0012745	0.0011645	0.0010445
		3	0.0016745	0.0015812	0.0013938	0.0012298
$(150, 100)$		1	0.0009535	0.0009324	0.0009135	0.0008645
		2	0.0011356	0.0010758	0.0009997	0.0093457
		3	0.0013587	0.0012785	0.0011024	0.0010534

Table 5. MSE of C_{py} using the different estimation methods when $(\lambda, \rho) = (15, 3)$.

(n, m)	TV of C_{py}	SC.	MLE	boot-p	boot-t	Bayesian
$(30, 15)$	0.8271514	1	0.0334512	0.0324755	0.0305856	0.0286475
		2	0.0456923	0.0449847	0.0415645	0.0378996
		3	0.0498745	0.0497856	0.0453678	0.0423845
$(30, 20)$		1	0.0112956	0.0094512	0.0091345	0.0088435
		2	0.0210478	0.0203635	0.0095912	0.0093747
		3	0.0253689	0.0245866	0.0099823	0.0096925
$(50, 30)$		1	0.0091525	0.0091277	0.0088656	0.0084678
		2	0.0097647	0.0096869	0.0092789	0.0089636
		3	0.0154236	0.0136878	0.0116988	0.0099774
$(50, 40)$		1	0.0069725	0.0067612	0.0064145	0.0058478
		2	0.0075447	0.0074556	0.0071275	0.0066555
		3	0.0082912	0.0081458	0.0077347	0.0072647
$(80, 55)$		1	0.0051623	0.0051163	0.0048856	0.0045756
		2	0.0056545	0.0055425	0.0053523	0.0051836
		3	0.0062423	0.0062741	0.0059269	0.0056925
$(80, 70)$		1	0.0033556	0.0033647	0.0031547	0.0027455
		2	0.0038347	0.0037858	0.0035555	0.0031534
		3	0.0045258	0.0044799	0.0041465	0.0036674
$(100, 80)$		1	0.0018654	0.0017825	0.0014923	0.0010814
		2	0.0021466	0.0021136	0.0019474	0.0012725
		3	0.0026847	0.0025614	0.0022555	0.0016836
$(100, 90)$		1	0.0015636	0.0014874	0.0012736	0.0009447
		2	0.0017947	0.0016585	0.0015233	0.0011058
		3	0.0019164	0.0018795	0.0017657	0.0013469
$(150, 100)$		1	0.0012658	0.0011784	0.0010345	0.0008936
		2	0.0013687	0.0012475	0.0011347	0.0009965
		3	0.0014658	0.0013785	0.0012785	0.0011474

Table 6. MSE of C_{py} using the different estimation methods when $(\lambda, \rho) = (35, 4.19)$.

(n, m)	TV of C_{py}	SC.	MLE	boot-p	boot-t	Bayesian
$(30, 15)$	0.864893	1	0.0486532	0.0476583	0.0456391	0.0413284
		2	0.0523214	0.0512474	0.0496355	0.0465213
		3	0.0556325	0.0543652	0.0523657	0.0496382
$(30, 20)$		1	0.0376521	0.0365427	0.0345266	0.0312476
		2	0.0396524	0.0387942	0.0370012	0.0332245
		3	0.0412356	0.0409683	0.0392177	0.0376542
$(50, 30)$		1	0.0293651	0.0275412	0.0258433	0.0199685
		2	0.0316525	0.0299653	0.0276543	0.0223144
		3	0.0336241	0.0321451	0.0300258	0.0247581
$(50, 40)$		1	0.0199875	0.0189573	0.0166784	0.0136542
		2	0.0213654	0.0199874	0.0183653	0.0156277
		3	0.0236217	0.0223657	0.0199685	0.0169994
$(80, 55)$		1	0.0112584	0.0099678	0.0085748	0.0077486
		2	0.0133657	0.0113699	0.0103654	0.0087459
		3	0.0149968	0.0136547	0.0125873	0.0096548
$(80, 70)$		1	0.0079685	0.0071547	0.0069845	0.0058974
		2	0.0082563	0.0080655	0.0077562	0.0069821
		3	0.0093847	0.0089546	0.0085994	0.0074630
$(100, 80)$		1	0.0036124	0.0035478	0.0029687	0.0027123
		2	0.0039258	0.0038654	0.0032999	0.0029487
		3	0.0041369	0.0040753	0.0038547	0.0035243
$(100, 90)$		1	0.0025841	0.0023986	0.0022454	0.0017369
		2	0.0026547	0.0024785	0.0024101	0.0018992
		3	0.0027653	0.0026894	0.0025846	0.0020553
$(150, 100)$		1	0.0016478	0.0014658	0.0013654	0.0009457
		2	0.0017856	0.0015945	0.0014783	0.0011986
		3	0.0018365	0.0016887	0.0015475	0.0012548

Table 7. ALs and CPs of 95% ACIs for C_{py} when $(\lambda, \rho) = (1, 0.085)$ and true value of $C_{py} = 0.210313$.

(n, m)	SC.	MLE		boot-p		boot-t		Bayesian	
		ALs	CPs	ALs	CPs	ALs	CPs	ALs	CPs
$(30, 15)$	1	0.6125844	0.9162	0.5963587	0.9227	0.5765478	0.9428	0.5536241	0.9456
	2	0.6354821	0.9175	0.6265833	0.9344	0.5963470	0.9429	0.5778422	0.9484
	3	0.6635423	0.9084	0.6569932	0.9295	0.6254715	0.9385	0.5996254	0.9415
$(30, 20)$	1	0.5696874	0.9237	0.5463542	0.9456	0.5294573	0.9436	0.5025875	0.9491
	2	0.5769335	0.9188	0.5632479	0.9344	0.5502141	0.9354	0.5300542	0.9517
	3	0.5963241	0.9199	0.5865477	0.9347	0.5746893	0.9467	0.5563274	0.9485
$(50, 30)$	1	0.5236211	0.9243	0.5147223	0.9273	0.4963582	0.9502	0.4701472	0.9623
	2	0.5462833	0.9277	0.5362474	0.9415	0.5214765	0.9590	0.4936921	0.9637
	3	0.5698244	0.9387	0.5578432	0.9388	0.5471288	0.9296	0.5203690	0.9551
$(50, 40)$	1	0.4869582	0.9344	0.4765417	0.9249	0.4569580	0.9379	0.4401245	0.9542
	2	0.4999587	0.9412	0.4895784	0.9197	0.4712354	0.9417	0.4625832	0.9574
	3	0.5123471	0.9392	0.4936574	0.9346	0.4896251	0.9297	0.4755625	0.9495
$(80, 55)$	1	0.4563582	0.9566	0.4468572	0.9357	0.4325174	0.9358	0.4199635	0.9647
	2	0.4763521	0.9384	0.4636981	0.9253	0.4521014	0.9395	0.4360149	0.9588
	3	0.4968932	0.9488	0.4825899	0.9287	0.4789652	0.9387	0.4586933	0.9624
$(80, 70)$	1	0.4263558	0.9436	0.4125130	0.9418	0.3999524	0.9518	0.3786542	0.9545
	2	0.4463587	0.9394	0.4352812	0.9399	0.4200142	0.9429	0.3965241	0.9556
	3	0.4599983	0.9287	0.4500124	0.9440	0.4469524	0.9385	0.4125837	0.9649

Table 7. *Cont.*

(n,m)	SC.	MLE		boot-p		boot-t		Bayesian	
(100, 80)	1	0.3999974	0.9413	0.3865475	0.9424	0.3769873	0.9534	0.3614728	0.9618
	2	0.4163589	0.9360	0.4025639	0.9377	0.3965472	0.9415	0.3856455	0.9597
	3	0.4365271	0.9387	0.4251786	0.9394	0.4153628	0.9424	0.4055766	0.9541
(100, 90)	1	0.3699657	0.9538	0.3578491	0.9495	0.3485647	0.9553	0.3278459	0.9642
	2	0.3935644	0.9479	0.3879654	0.9526	0.3748965	0.9522	0.3564721	0.9553
	3	0.4125432	0.9480	0.3996543	0.9547	0.3847517	0.9491	0.3665423	0.9495
(150, 100)	1	0.2845688	0.9499	0.2769542	0.9377	0.2598476	0.9424	0.2356478	0.9545
	2	0.3025479	0.9511	0.2965487	0.9394	0.2798654	0.9553	0.2586344	0.9556
	3	0.3265478	0.9499	0.3158974	0.9495	0.2996572	0.9522	0.2736955	0.9649

Table 8. ALs and CPs of 95% ACIs for C_{py} when $(\lambda, \rho) = (1.5, 0.22)$ and true value of $C_{py} = 0.40858$.

(n,m)	SC.	MLE		boot-p		boot-t		Bayesian	
		ALs	**CPs**	**ALs**	**CPs**	**ALs**	**CPs**	**ALs**	**CPs**
(30, 15)	1	0.5436123	0.9151	0.5234458	0.9223	0.5174547	0.9416	0.4987124	0.9457
	2	0.5634455	0.9172	0.5457256	0.9346	0.5299635	0.9425	0.5136685	0.9488
	3	0.5987258	0.9095	0.5763364	0.9295	0.5568452	0.9394	0.5398745	0.9415
(30, 20)	1	0.5163145	0.9234	0.4987475	0.9414	0.4786786	0.9417	0.4582365	0.9493
	2	0.5366635	0.9197	0.5214258	0.9367	0.4999367	0.9358	0.4791896	0.9514
	3	0.5523856	0.9187	0.5497364	0.9348	0.5211954	0.9459	0.5156784	0.9487
(50, 30)	1	0.4837658	0.9238	0.4756784	0.9299	0.4561654	0.9505	0.4199562	0.9625
	2	0.5036478	0.9279	0.4987691	0.9417	0.4738361	0.9493	0.4401784	0.9636
	3	0.5264369	0.9374	0.5135354	0.9380	0.4976364	0.9280	0.4655367	0.9558
(50, 40)	1	0.4599458	0.9345	0.4489745	0.9241	0.4397745	0.9374	0.4001457	0.9540
	2	0.4765789	0.9410	0.4684126	0.9192	0.4512684	0.9415	0.4327693	0.9579
	3	0.4899452	0.9395	0.4857475	0.9343	0.4711452	0.9286	0.4536573	0.9495
(80, 55)	1	0.4361397	0.9416	0.4236943	0.9364	0.4008784	0.9347	0.3899021	0.9647
	2	0.4469452	0.9377	0.4367764	0.9256	0.4233698	0.9398	0.4111035	0.9584
	3	0.4598783	0.9424	0.4497785	0.9272	0.4388784	0.9284	0.4257470	0.9625
(80, 70)	1	0.4165746	0.9435	0.3999364	0.9413	0.3794367	0.9515	0.3597145	0.9544
	2	0.4362453	0.9396	0.4251453	0.9391	0.3984658	0.9426	0.3765568	0.9556
	3	0.4487881	0.9280	0.4358784	0.9407	0.4135745	0.9380	0.3899698	0.9647
(100, 80)	1	0.3984364	0.9412	0.3788659	0.9427	0.3579635	0.9537	0.3410784	0.9618
	2	0.4156669	0.9363	0.3874743	0.9376	0.3695254	0.9418	0.3587369	0.9594
	3	0.4358152	0.9380	0.4287334	0.9394	0.3895754	0.9420	0.3718578	0.9542
(100, 90)	1	0.3545468	0.9526	0.3465755	0.9495	0.3267368	0.9544	0.3111452	0.9644
	2	0.3854475	0.9483	0.3647456	0.9516	0.3494741	0.9525	0.3378378	0.9556
	3	0.3999693	0.9474	0.3894778	0.9527	0.3712058	0.9483	0.3562967	0.9497
(150, 100)	1	0.2900147	0.9396	0.2786584	0.9591	0.2658479	0.9518	0.2487642	0.9642
	2	0.3125864	0.9280	0.3012124	0.9407	0.2836541	0.9421	0.2684571	0.9544
	3	0.3365472	0.9412	0.3265875	0.9427	0.3110124	0.9543	0.2874614	0.9596

Table 9. ALs and CPs of 95% ACIs for C_{py} when $(\lambda, \rho) = (2.2, 0.5)$ and true value of $C_{py} = 0.60299$.

(n, m)	SC.	MLE		boot-p		boot-t		Bayesian	
		ALs	CPs	ALs	CPs	ALs	CPs	ALs	CPs
(30, 15)	1	0.4236335	0.9191	0.4147354	0.9277	0.3948745	0.9343	0.3782145	0.9396
	2	0.4392478	0.9212	0.4288087	0.9318	0.4127965	0.9352	0.3899753	0.9415
	3	0.4435568	0.9223	0.4410657	0.9299	0.4256784	0.9261	0.4023968	0.9384
(30, 20)	1	0.3854457	0.9314	0.3799964	0.9344	0.3657589	0.9414	0.3389096	0.9413
	2	0.3984665	0.9295	0.3895357	0.9355	0.3766965	0.9435	0.3511445	0.9528
	3	0.4028098	0.9286	0.4001951	0.9416	0.3945357	0.9396	0.3764078	0.9429
(50, 30)	1	0.3469756	0.9318	0.3455789	0.9347	0.3312459	0.9457	0.3125530	0.9517
	2	0.3654096	0.9247	0.3599369	0.9459	0.3487786	0.9418	0.3327786	0.9525
	3	0.3845147	0.9251	0.3816457	0.9519	0.3655328	0.9387	0.3144478	0.9498
(50, 40)	1	0.3168145	0.9410	0.3111785	0.9525	0.2995336	0.9519	0.2756458	0.9558
	2	0.3365023	0.9373	0.3299096	0.9504	0.3157554	0.9498	0.2899692	0.9617
	3	0.3487024	0.9364	0.3466365	0.9487	0.3324478	0.9465	0.3055746	0.9575
(80, 55)	1	0.2865458	0.9425	0.2745784	0.9518	0.2563235	0.9527	0.2410364	0.9564
	2	0.3045965	0.9416	0.2998562	0.9529	0.2767967	0.9478	0.2654785	0.9613
	3	0.3177098	0.9450	0.3124078	0.9494	0.2954783	0.9517	0.2863257	0.9604
(80, 70)	1	0.2461784	0.9481	0.2357950	0.9545	0.2262473	0.9568	0.2098127	0.9555
	2	0.2576145	0.9432	0.2501475	0.9416	0.2355145	0.9419	0.2211365	0.9496
	3	0.2658635	0.9393	0.2612456	0.9447	0.2436698	0.9395	0.2275478	0.9517
(100, 80)	1	0.1836988	0.9497	0.1754124	0.9518	0.1568754	0.9513	0.1277589	0.9643
	2	0.1911884	0.9385	0.1889098	0.9494	0.1657789	0.9454	0.1356256	0.9562
	3	0.1986461	0.9371	0.1934365	0.9415	0.1785456	0.9445	0.1512447	0.9525
(100, 90)	1	0.1547778	0.9572	0.1456745	0.9480	0.1369365	0.9546	0.1124557	0.9534
	2	0.1699357	0.9535	0.1587096	0.9543	0.1511745	0.9498	0.1235357	0.9617
	3	0.1853456	0.9496	0.1783365	0.9509	0.1694692	0.9517	0.1452786	0.9553
(150, 100)	1	0.1236547	0.9485	0.1136585	0.9594	0.1036547	0.9457	0.0996847	0.9662
	2	0.1365428	0.9371	0.1235487	0.9416	0.1164782	0.9446	0.1100352	0.9625
	3	0.1436952	0.9472	0.1354266	0.9380	0.1284578	0.9544	0.1196548	0.9534

Table 10. ALs and CPs of 95% ACIs for C_{py} when $(\lambda, \rho) = (15, 3)$ and true value of $C_{py} = 0.82715$.

(n, m)	SC.	MLE		boot-p		boot-t		Bayesian	
		ALs	CPs	ALs	CPs	ALs	CPs	ALs	CPs
(30, 15)	1	0.3657325	0.9292	0.3577456	0.9419	0.3356365	0.9421	0.3142365	0.9515
	2	0.3765568	0.9313	0.3689784	0.9398	0.3512951	0.9316	0.3288475	0.9544
	3	0.3812698	0.9286	0.3799456	0.9317	0.3657753	0.9335	0.3433125	0.9497
(30, 20)	1	0.3254745	0.9125	0.3155085	0.9424	0.2966654	0.9414	0.2745058	0.9466
	2	0.3356695	0.9254	0.3341320	0.9385	0.3152852	0.9397	0.2857047	0.9525
	3	0.3399254	0.9177	0.3391470	0.9426	0.3257657	0.9372	0.2999450	0.9534
(50, 30)	1	0.2863247	0.9388	0.2789365	0.9396	0.2569958	0.9417	0.2365360	0.9616
	2	0.2958869	0.9295	0.2895960	0.9283	0.2761748	0.9393	0.2564475	0.9585
	3	0.3056547	0.9274	0.2987458	0.9214	0.2814369	0.942	0.2631650	0.9574
(50, 40)	1	0.2456124	0.9385	0.2387789	0.9395	0.2269756	0.9511	0.2054145	0.9641
	2	0.2563586	0.9346	0.2497562	0.9456	0.2354354	0.9490	0.2200365	0.9582
	3	0.2732698	0.9379	0.2699365	0.9397	0.2497698	0.9432	0.2358875	0.9543
(80, 55)	1	0.1956784	0.9418	0.1877745	0.9418	0.1763478	0.9523	0.1564556	0.9644
	2	0.2121658	0.9387	0.2045556	0.9429	0.1954544	0.9514	0.1764778	0.9585
	3	0.2236324	0.9401	0.2154778	0.9384	0.2077457	0.9477	0.1899097	0.9597

Table 10. *Cont.*

(n,m)	SC.	MLE		boot-p		boot-t		Bayesian	
		ALs	CPs	ALs	CPs	ALs	CPs	ALs	CPs
(80, 70)	1	0.1458478	0.9512	0.1411098	0.9425	0.1365639	0.9558	0.1254445	0.9648
	2	0.1568456	0.9493	0.1522457	0.9416	0.1467563	0.9515	0.1321456	0.9665
	3	0.1636876	0.9416	0.1612635	0.9399	0.1574485	0.9534	0.1399789	0.9594
(100, 80)	1	0.1236894	0.9525	0.1214478	0.9518	0.1158254	0.9562	0.1111524	0.9693
	2	0.1358853	0.9424	0.1341524	0.9477	0.1257852	0.9523	0.1204325	0.9644
	3	0.1456235	0.9434	0.1399456	0.9417	0.1298789	0.9517	0.1255145	0.9652
(100, 90)	1	0.1014458	0.9545	0.0999415	0.9618	0.0975963	0.9550	0.0913635	0.9643
	2	0.1198762	0.9556	0.1154635	0.9526	0.1101475	0.9491	0.1036784	0.9586
	3	0.1247557	0.9469	0.1212478	0.9512	0.1197658	0.9516	0.1099011	0.9544
(150, 100)	1	0.0884573	0.9426	0.0856354	0.9478	0.0814536	0.9623	0.7896544	0.9645
	2	0.9532456	0.9435	0.0935478	0.9416	0.0896545	0.9516	0.0843657	0.9651
	3	0.1035476	0.9545	0.0978546	0.9608	0.0943657	0.9551	0.0893219	0.9640

Table 11. ALs and CPs of 95% ACIs for C_{py} when $(\lambda, \rho) = (35, 4.19)$ and true value of $C_{py} = 0.864893$.

(n,m)	SC.	MLE		boot-p		boot-t		Bayesian	
		ALs	CPs	ALs	CPs	ALs	CPs	ALs	CPs
(30, 15)	1	0.3325145	0.9152	0.3245877	0.9227	0.3145839	0.9417	0.2936547	0.9454
	2	0.3465412	0.9173	0.3362481	0.9348	0.3247852	0.9428	0.3121457	0.9485
	3	0.3554786	0.9091	0.3445627	0.9299	0.3365248	0.9399	0.3278932	0.9416
(30, 20)	1	0.2832564	0.9234	0.2698573	0.9410	0.2563587	0.9415	0.2365479	0.9499
	2	0.2963582	0.9195	0.2798658	0.9361	0.2654713	0.9354	0.2436910	0.9517
	3	0.3165479	0.9186	0.2968475	0.9344	0.2802140	0.9453	0.2632107	0.9488
(50, 30)	1	0.2563910	0.9239	0.2465827	0.9295	0.2396847	0.9505	0.2114578	0.9622
	2	0.2651478	0.9277	0.2536921	0.9416	0.2436988	0.9494	0.2269941	0.9635
	3	0.2754687	0.937	0.2639472	0.9387	0.2563920	0.9284	0.2415873	0.9551
(50, 40)	1	0.2136954	0.9344	0.2014751	0.9248	0.1936574	0.9375	0.1769542	0.9542
	2	0.2236954	0.9415	0.2200361	0.9199	0.2100478	0.9416	0.1936478	0.9573
	3	0.2463951	0.9396	0.2365472	0.9344	0.2245837	0.9287	0.2103645	0.9476
(80, 55)	1	0.1765894	0.9419	0.1632982	0.9365	0.1593824	0.9347	0.1363524	0.9645
	2	0.1836547	0.9378	0.1745893	0.9253	0.1669475	0.9395	0.1456482	0.9584
	3	0.1936574	0.9431	0.1833221	0.9297	0.1745632	0.927	0.1573624	0.962
(80, 70)	1	0.1236542	0.9422	0.1165428	0.9418	0.1036478	0.9517	0.0996341	0.955
	2	0.1363954	0.9393	0.1236932	0.9395	0.1165872	0.9438	0.1100021	0.9540
	3	0.1436958	0.9296	0.1365241	0.9404	0.1234587	0.9380	0.1199687	0.9644
(100, 80)	1	0.1036547	0.9415	0.0968742	0.9436	0.0854793	0.9544	0.0736958	0.9625
	2	0.1136548	0.9307	0.1023457	0.9371	0.0936528	0.9415	0.0836542	0.9583
	3	0.1236544	0.9351	0.1125478	0.9394	0.1036954	0.9427	0.0965847	0.9514
(100, 90)	1	0.0836547	0.9524	0.0745869	0.9495	0.0632478	0.9535	0.0556842	0.9645
	2	0.0936547	0.9485	0.0832542	0.9516	0.0785693	0.9554	0.0693587	0.9546
	3	0.1023654	0.9476	0.0936475	0.9521	0.0854722	0.9482	0.0736458	0.9487
(150, 100)	1	0.0723654	0.9405	0.0665478	0.9384	0.0593654	0.9437	0.0499974	0.9645
	2	0.0813655	0.9308	0.0723694	0.9496	0.0652475	0.9389	0.0543692	0.9624
	3	0.0935416	0.9361	0.0846589	0.9517	0.0776549	0.9545	0.0625847	0.9582

7. Data Analysis

In this section, to illustrate the inferential procedures discussed in the previous sections, two sets of real data are analysed. The first set represents the initial failure times (in months) for 20 electric vans used for internal transportation and delivery at a large manufacturing

facility and details are presented in Zimmer et al. [44] and recently used by Saha et al. [22]. The second data group represents the failure intervals (in hours) of the air conditioning system of a 13 Boeing 720 aircraft taken from Proschan [45]. The two sets of data are as follows.

- Data Set I: 0.9, 1.5, 2.3, 3.2, 3.9, 5.0, 6.2, 7.5, 8.3, 10.4, 11.1, 12.6, 15.0, 16.3, 19.3, 22.6, 24.8, 31.1, 38.1, 53.0.
- Data Set II: 1, 4, 11, 16, 18, 18, 18, 24, 31, 39, 46, 51, 54, 63, 68, 77, 80, 82, 97, 106, 111, 141, 142, 163, 191, 206, 216.

By evaluating the quality of fit of the two datasets, we first determined whether the analysed datasets genuinely originate from PD or not. The Kolmogorov-Smirnov (K-S) statistic and associated p-values are the foundation of this method. The K-S statistic measures the separation between two samples' empirical distribution functions or between their empirical distribution functions and the CDF of reference distribution. As a result, the K-S statistic is solely used to evaluate fit quality and not as a selection criterion. It is 0.067503 for the first group with a p-value of 0.9999, and 0.11513 for the second group with a p-value of 0.8666. The fact that the p-values are so high means that we cannot rule out the possibility that the data came from a Pareto model, and as a result, this probability model fits the real datasets well. Figures 3 and 4 display empirical, Q-Q, and P-P charts, which demonstrate how well PD fits the data. Here, we have fixed hypothetical LSL and hypothetical USL are $(L, U) = (0.911, 31)$ for data Set I and $(2, 215)$ for data set II with desirable yield $P_0 = 0.95$.

According to the data set I, we can generate the PT2C sample of size $m = 9$ taken from a sample of size $n = 20$ with censoring scheme $R = (6, 2, 1, 2, 0^*5)$ using the algorithm described in Balakrishnan and Sandhu [16]. A PT2C sample generated from the data set I is given as follows

$$0.9 \quad 1.5 \quad 3.2 \quad 3.9 \quad 5.0 \quad 6.2 \quad 22.6 \quad 24.8 \quad 31.1$$

Similarly, based on the data Set II, we can generate the PT2C sample of size $m = 15$ taken from a sample of size $n = 27$ with censoring scheme $R = (4, 3, 1, 2, 0^*4, 1, 0^*2, 1, 0^*3)$. The PT2C sample is

$$1 \quad 4 \quad 11 \quad 16 \quad 18 \quad 18 \quad 18 \quad 31 \quad 39 \quad 51 \quad 54 \quad 68 \quad 82 \quad 141 \quad 216$$

For the previous data sets considered, based on a PT2C, we have computed the point estimates of the index C_{py} using ML, boot-p, boot-t and Bayes methods, the results are reported in Table 12. Further, we determined the 95% ACIs based on ML and bootstrap methods as well as 95% HPD credible interval using MCMC samples and the results are listed in Table 13. In the Bayes framework, we assume that the non-informative priors for λ and ρ, that is, when $a_i = 0.0001$ and $b_i = 0.0001$, $i = 1, 2$. In addition, 12000 MCMC samples were generated, and the first 2000 samples were generated as 'burn-in'. Figures 5 and 6 display the trace plots of C_{py} computed by MCMC approach for data Sets I and II.

Table 12. Different point estimates of C_{py}.

			bootstrap		**Bayes**
Data Set	(n, m)	**MLE**	**boot-p**	**boot-t**	**MCMC**
I	$(20, 9)$	0.894131	0.882481	0.875541	0.852453
II	$(27, 15)$	0.778609	0.769877	0.762213	0.711487

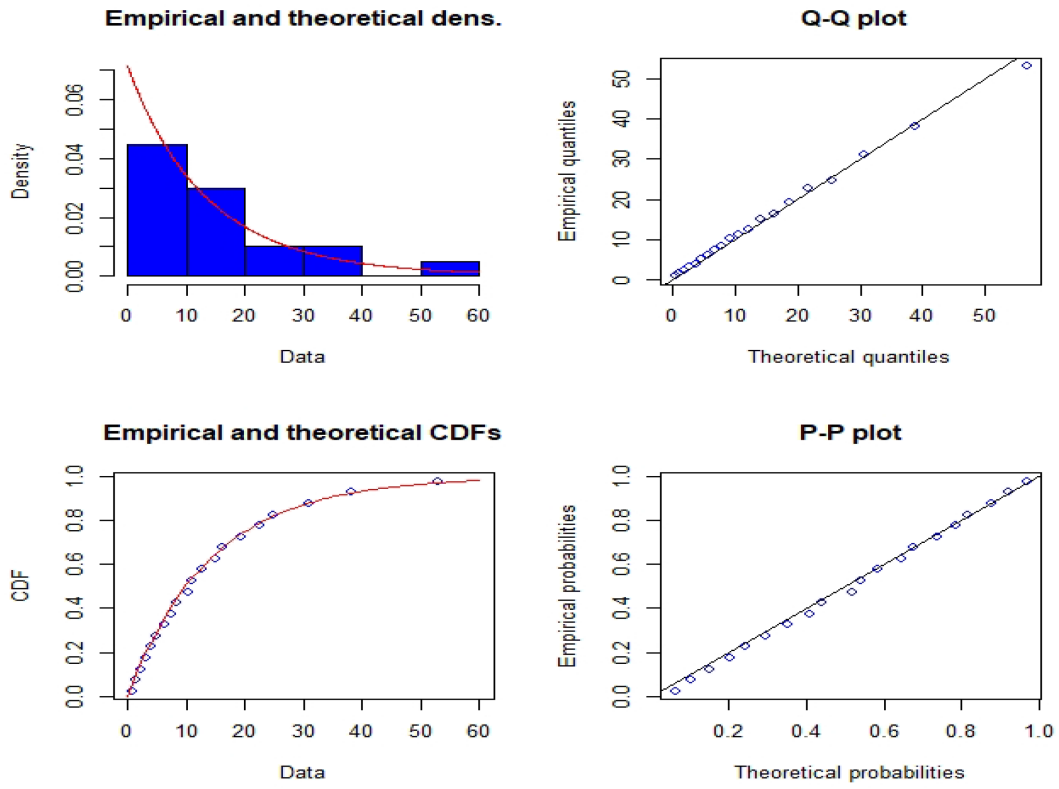

Figure 3. Empirical , Q-Q and P-P plots of PD for data set I.

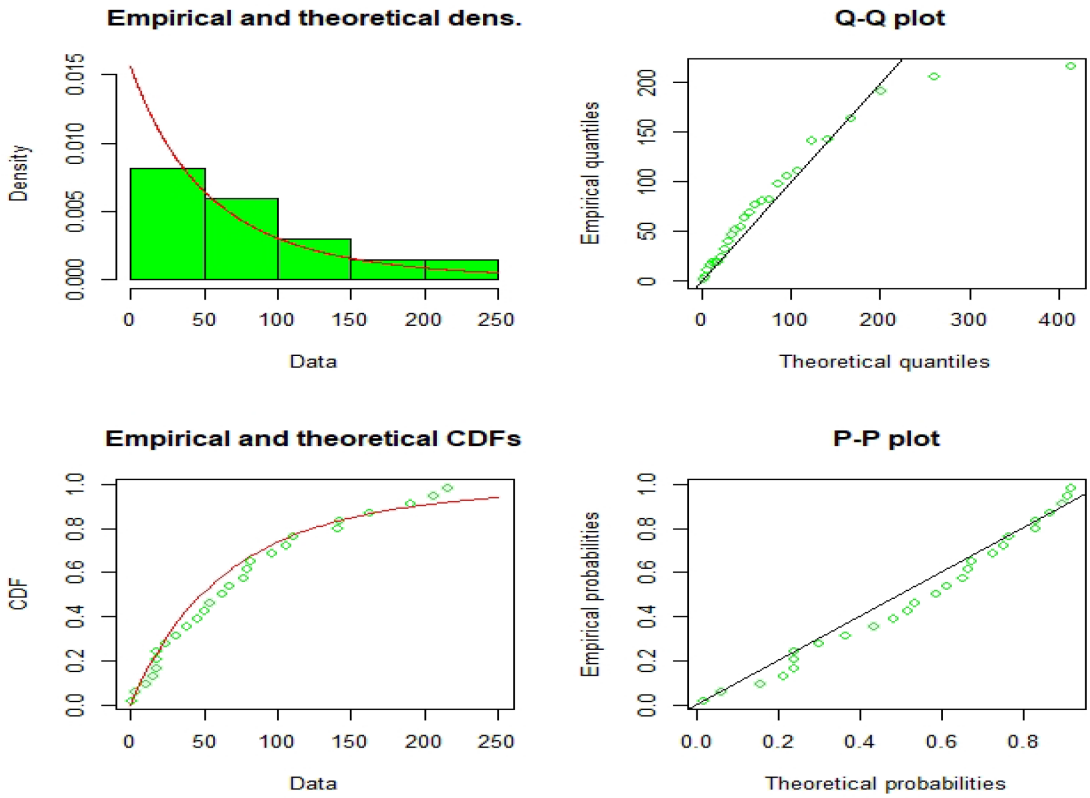

Figure 4. Empirical , Q-Q and P-P plots of PD for data set II.

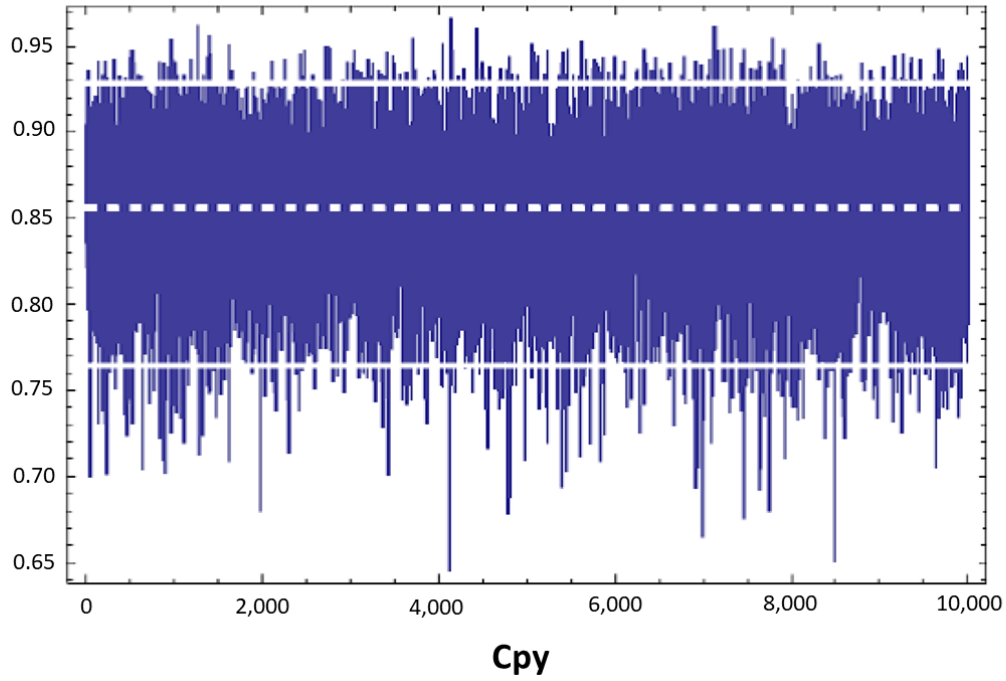

Figure 5. Trace plot of C_{py} obtained from MCMC for Set I.

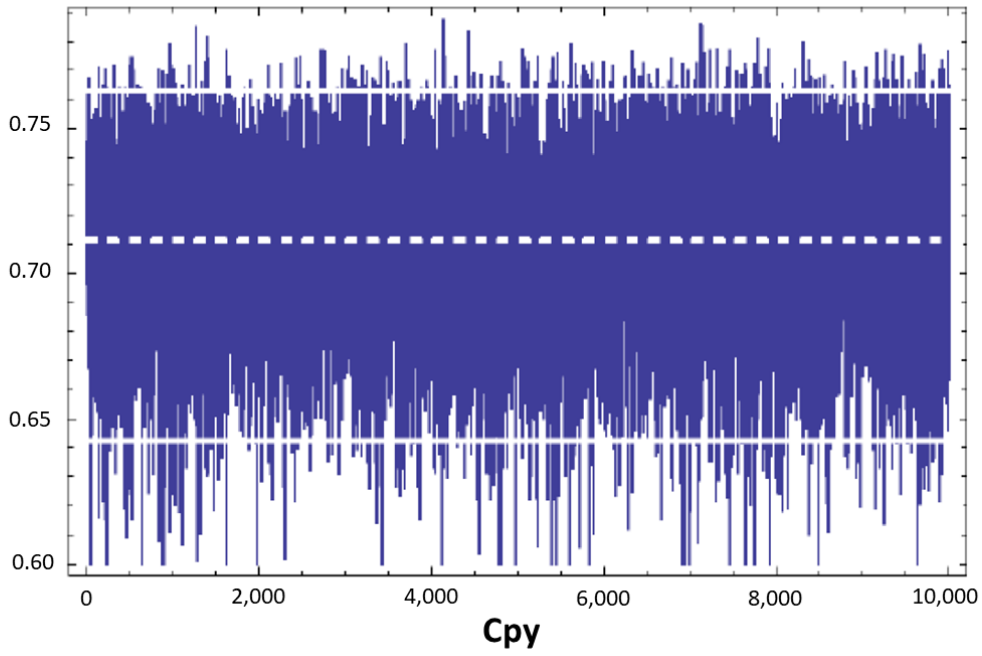

Figure 6. Trace plot of C_{py} obtained from MCMC for Set II.

Table 13. 95% ACIs and 95% HPD credible intervals of C_{py}.

	MLE			boot-p		
Data Set	**L**	**U**	**Length**	**L**	**U**	**Length**
I	0.856558	0.931704	0.0751454	0.849621	0.920542	0.070921
II	0.753653	0.803564	0.0499106	0.642583	0.791245	0.044663
	boot-t			**MCMC**		
Data set	**L**	**U**	**Length**	**L**	**U**	**Length**
I	0.837743	0.911722	0.073979	0.763623	0.923541	0.159918
II	0.738458	0.781534	0.043076	0.642475	0.763876	0.121401

8. Conclusions

In this manuscript, we have considered four different estimation techniques, namely maximum likelihood, bootstrap bounds, and Bayes to obtain the C_{py} index estimation and illustrate the proposed methods using two practical examples. MLEs are derived using NR's iterative numerical technique. Meanwhile, the asymptotic confidence intervals are generated based on the observed and predicted Fisher information matrices. In order to address the problem of small sample size, two bootstrap confidence intervals were generated. The Bayesian estimate within the squared error loss function is also taken into account and the estimates are derived through a significant sampling procedure using the Metropolis – Hasting algorithm. Moreover, credible periods with a higher corresponding background intensity are generated. Since it is not possible to compare these methods theoretically, we performed a large-scale simulation study to compare these methods with different sample sizes (n, m), different control schemes (1, 2, 3) and different values of (λ, ρ). In the simulation section, squared error values are averaged to evaluate point estimation performance while mean lengths and coverage rates are taken into account for interval estimation. Results for MSE estimates are reported in Tables 2–6, while results for ALs and CPs for estimates are presented in Tables 7–11. Finally, we feel that the contents of the manuscript may be useful to researchers and practitioners in various fields of industry where lifetime distributions are widely used.

In future research, we will discuss lifetime performance index assessment with numerical analysis based on the second type of adaptive stepwise control.

Author Contributions: Conceptualization, R.M.E.-S. and M.S.E.; Methodology, M.E.-M., K.M.A. and R.M.E.-S.; Software, M.E.-M., L.A.A.-E. and R.M.E.-S.; Validation, L.A.A.-E.; Formal analysis, M.S.E., R.M.E.-S. and K.M.A; Resources, M.E.-M. and K.M.A.; Data curation, M.S.E. and M.E.-M.; Writing—original draft, R.M.E.-S.; Writing—review & editing, M.S.E. and R.M.E.-S.; Supervision, R.M.E.-S. and M.S.E. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2023R443), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia and Prince Sattam bin Abdulaziz University, project number (PSAU/2023/R/1444).

Data Availability Statement: Data is listed within the paper.

Acknowledgments: Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2023R443), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. This study is supported via funding from Prince Sattam bin Abdulaziz University, project number (PSAU/2023/R/1444).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Montgomery, D.C. *Introduction to Statistical Quality Control*; John Wiley & Sons: New York, NY, USA, 2005.
2. Kane, V.E. Process capability indices. *J. Qual.* **1986**, *18*, 41–52. [CrossRef]
3. Kotz, S.; Johnson, N.L. Process capability indicesa review. *J. Qual. Technol.* **2002**, *34*, 2–19. [CrossRef]

4. Kaminsky, F.C.; Dovich, R.A.; Burke, R.J. Process capability indices: Now and in the future. *Qual. Eng.* **1998**, *10*, 445–453. [CrossRef]
5. Schneider, H.; Pruett, J.; Lagrange, C. Uses of process capability indices in the supplier certification process. *Qual. Eng.* **1995**, *8*, 225–235. [CrossRef]
6. Juran, J.M.; Gryna, F.M. *Quality Control Handbook*, 3rd ed.; McGraw-Hill: New York, NY, USA, 1974.
7. Chan, L.K.; Spiring, F.; Xiao, H. *An OC Curve Approach for Analyzing the Process Capability Index Cpk, Technical Report*; Department of Statistics, University of Manitoba: Winnipeg, MB, Canada, 1988.
8. Kotz, S.; Lovelace, C.R. *Process Capability Indices in Theory and Practice*; Arnold: London, UK, 1998.
9. Pearn, W.L.; Kotz, S.; Johnson, N.L. Distributional and inferential properties of process capability indices. *J. Qual. Technol.* **1992**, 24, 216–231. [CrossRef]
10. Gunter, B.H. The use and abuse of C_{pk}. *Qual. Prog.* **1989**, *22*, 108–109.
11. Pearn, W.L.; Chen, K.S. Estimating process capability indices for non-normal Pearsonian populations. *Qual. Reliab. Int.* **1995**, *11*, 386–388. [CrossRef]
12. Pearn, W.L.; Chen, K.S. A Bayesian-like estimator of C_{pk}. *Commun. Stat. Simul. Comput.* **1996**, *25*, 321–329. [CrossRef]
13. Clements, J.A. Process capability calculations for nonnormal distributions. *Qual. Prog.* **1989**, *22*, 95–100.
14. Polansky, A.M. A smooth nonparametric approach to process capability. *Qual. Reliab. Eng. Int.* **1998**, *14*, 43–48. [CrossRef]
15. Maiti, S.S.; Saha, M.; Nanda, A.K. On generalizing process capability indices. *Qual. Technol. Quant. Manag.* **2010**, *7*, 279–300. [CrossRef]
16. Balakrishnan, N.; Sandhu, R.A. Best linear unbiased and maximum likelihood estimation for exponential distributions under general progressive Type-II censored samples. *Sankhya B* **1996**, *58*, 1–9.
17. Balakrishnan, N.; Kannan, N.; Lin, C.T.; Wu, S.J.S. Inference for the extreme value distribution under progressive Type-II censoring. *Stat. Comput. Simul.* **2004**, *74*, 25–45. [CrossRef]
18. Fernindez, A.J.; Carlos, J.P.G.; Aslam, M.; Chi-Hyuck, J. Design of progressively censored group sampling plans for Weibull distributions: An optimization problem. *Eur. J. Oper. Res.* **2011**, *211*, 525–532. [CrossRef]
19. Aslam, M.; Huang, S.-R.; Jun, C.-H.; Ahmad, M.; Rasool, M. A reliability sampling plan based on progressive interval censoring under Pareto distribution of second kind. *Ind. Eng. Manag.* **2011**, *10*, 154–160.
20. Panahi, H. Interval estimation of Kumaraswamy parameters based on progressively type II censored sample and record values. *Miskolc Math. Notes* **2020**, *21*, 319–334. [CrossRef]
21. Wang, L.; Dey, S.; Tripathi, Y.M. Classical and Bayesian inference of the inverse Nakagami distribution based on progressive Type-II censored samples. *Mathematics* **2022**, *10*, 2137. [CrossRef]
22. Wang, R.; Ghen, H.; Guan, C. A Bayesian inference-based approach for performance prognostics towards uncertainty quantification and its applications on the marine diesel engine. *ISA Trans.* **2021**, *118*, 159–173. [CrossRef] [PubMed]
23. Luo, C.; Shen, L.; Xu, A. Modelling and estimation of system reliability under dynamic operating environments and lifetime ordering constraints. *Reliab. Eng. Syst. Saf.* **2022**, *218*, 108136. [CrossRef]
24. Saberzadeh, Z.; Razmkhah, M.; Amini, M. Bayesian reliability analysis of complex k-out-of-n: ℓ systems under degradation performance. *Reliab. Eng. Syst. Saf.* **2023**, *231*, 109020. [CrossRef]
25. Zhuang, L.; Xu, A.; Wang, X.-L. A prognostic driven predictive maintenance framework based on Bayesian deep learning. *Reliab. Syst. Saf.* **2023**, *234*, 109181. [CrossRef]
26. Wu, S.F.; Chiu, C.J. Computational testing algorithmic procedure of assessment for lifetime performance index of products with two parameter exponential distribution based on the multiply Type-II censored sample. *Stat. Comput. Simul.* **2014**, *84*, 2106–2122. [CrossRef]
27. Hong, C.W.; Wu, J.W.; Cheng, C.H. Computational procedure of performance assessment of lifetime index of businesses for the Pareto lifetime model with the right Type-II censored sample. *Appl. Math. Comput.* **2007**, *184*, 336–350. [CrossRef]
28. Lee, W.C.; Wu, J.W.; Hong, C.W. Assessing the lifetime performance index of products from progressively Type-II right censored data using Burr XII model. *Math. Comput. Simul.* **2009**, *79*, 2167–2179. [CrossRef]
29. Saha, M.; Dey, S.; Yadav, A.S.; Kumar, S. Classical and Bayesian inference of C_{py} for generalized Lindley distributed quality characteristic. *Qual. Reliab. Eng. Int.* **2019**, *35*, 2593–2611. [CrossRef]
30. Ahmadi, M.V.; Doostparast, M. Pareto analysis for the lifetime performance index of products on the basis of progressively first-failure censored batches under balanced symmetric and asymmetric loss functions. *Appl. Stat.* **2019**, *46*, 1196–1227. [CrossRef]
31. Ahmadi, M.V.; Doostparast, M. Evaluating the lifetime performance index of products based on progressively Type-II censored Pareto samples: A new Bayesian approach. *Qual. Reliab. Eng. Int.* **2022**, *38*, 1612–1634. [CrossRef]
32. EL-Sagheer, R.M.; Jawa, T.M.; Sayed-Ahmed, N. Assessing the lifetime performance index with digital inferences of power hazard function distribution using progressive Type-II censoring scheme. *Comput. Intell. Neurosci.* **2022**, *2022*, 6467724. [CrossRef]
33. Pareto, V. *Cours d'Economie Politique*; Rouge et Cie: Paris, France, 1897.
34. Verma, V.; Betti, G. EU statistics on income and living conditions (EUSILC): Choosing the survey structure and sample design. *Stat. Transit.* **2006**, *7*, 935–970.
35. Lawless, J.F. *Statistical Models and Methods for Lifetime Data*; John Wiley & Sons: Hoboken, NJ, USA, 2011.
36. Kim, J.H.T.; Ahn, S.; Ahn, S. Parameter estimation of the Pareto distribution using a pivotal quantity. *J. Korean Stat. Soc.* **2017**, *46*, 438–450. [CrossRef]

37. Cohen, A.C. Maximum likelihood estimation in the Weibull distribution based on complete and on censored samples. *Technometrics* **1965**, *7*, 579–588. [CrossRef]
38. Greene, W.H. *Econometric Analysis*, 4th ed.; Prentice-Hall: NewYork, NY, USA, 2000.
39. Efron, B. Bootstrap methods: Another look at the jackknife. *Ann. Stat.* **1979**, *27*, 1–26. [CrossRef]
40. Davison, A.C.; Hinkley, D.V. *Bootstrap Methods and Their Application*; Cambridge University Press: Cambridge, UK, 1997.
41. DiCiccio, T.J.; Efron, B. Bootstrap confidence intervals. *Stat. Sci.* **1996**, *11*, 189–212. [CrossRef]
42. Kundu, D.; Howlader, H. Bayesian inference and prediction of the inverse Weibull distribution for Type-II censored data. *Comput. Stat. Data Anal.* **2010**, *54*, 1547–1558. [CrossRef]
43. Metropolis, N.; Rosenbluth, A.W.; Rosenbluth, M.N.; Teller, A.H.; Teller, E. Equations of state calculations by fast computing machines. *J. Chem. Phys.* **1953**, *21*, 1087–1091. [CrossRef]
44. Zimmer, L.S.; Hubele, N.F. Quantiles of the sampling distribution of C_{pm}. *Qual. Eng.* **1998**, *10*, 309–329. [CrossRef]
45. Proschan, F. Theoretical explanation of observed decreasing failure rate. *Technometrics* **2000**, *42*, 7–11. [CrossRef]