Estimating a parametric lifetime distribution from superimposed renewal process data

by

Ye Tian

A thesis submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY

Major: Statistics

Iowa State University
Ames, Iowa
2013

DEDICATION

To my family.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I am extraordinarily grateful to my advisor, Dr. William Q. Meeker, for his insightful guidance, generous support and enthusiastic encouragement during my graduate study. I would like to express my sincere gratitude to my committee members, Drs. Huaiqing Wu, Ulrike Genschel, Ranjan Maitra and Vivekananda Roy for their valuable suggestions and precious help. I am also very thankful to Dr. Luis A. Escobar from Department of Experimental Statistics, Louisiana State University for his collaborations and help over these years.

A lot of thanks to the faculty members in the department for offering great graduate courses on various topics that broadened my knowledge, sharpened my technical skills and get me well prepared for my career. I am very grateful to all my friends who have been supporting me all these years. My life would have been much less colorful without their accompanies.

Last but not least, none of these would have been made possible without my family. My parents have always believed in me and supported me to chase my dreams. Special thanks go to my wife, Yunyun. No matter how difficult the situation was, she always trusted me and supported me. I was able to explore my study free of worry and fear, because I knew she would be with me for every moment.

ABSTRACT

Maintenance data provides information about the reliability of systems and components. For reparable systems, a failed component will be replaced with a new one, which, in some cases, can be assumed to have the same lifetime distribution as the old component. Estimation of the lifetime distribution is particularly complicated if there are multiple copies of a component installed in multiple locations (different systems and different slots within a system) and information on exactly which component was replaced is not available. In such applications, the replacement history for a collection of locations forms a superposition of renewal processes (SRP). In this dissertation, statistical models and methods motivated by real applications with SRP data were developed for estimating component failure-time distribution. In Chapter 2, the SRP data were modeled by a nonhomogeneous Poisson process (NHPP). This method is motivated by the data structure that can be formulated as a single SRP. The NHPP approximation to the SRP is more adequate as the number of slots in the SRP gets larger. When the SRP has a large number of slots, the ML estimator based on NHPP assumption performs well and the interval estimation procedure based on transformed parametric bootstrap-t method has coverage probabilities close to the nominal values. By comparing the NHPP estimator with an alternative estimator, we make recommendations about which estimator to use for analyzing SRP data. We hope the recommendation provides some guideline for statisticians and engineers for better reliability analysis. In Chapter 3, the exact likelihood for the SRP data is derived. This likelihood-based method is motivated by the data structure that can be formulated as a fleet of independent SRP's. All SRP's in the fleet are assumed to correspond to a same component lifetime distribution. By considering all possible data configurations that could lead to the observed event history, the likelihood can be computed as a weighted sum of conditional likelihoods corresponding to all unique data configurations, with weights being probabilities for each data configuration. We use an ML estimation procedure that starts with a crude estimate of the weights and updates them iteratively. In Chapter 4, we describe an R function that implements the maximum likelihood estimation procedure described in Chapter 3. The need to enumerate all possible data configurations makes the estimation procedure complicated and computationally intensive. We developed an R function that implements the estimation procedure. The function takes the recurrence event data as the input and returns the ML estimates and the estimated covariance matrix of the parameter estimates. Details about the function implementation and how to use the function are discussed.

CHAPTER 1. GENERAL INTRODUCTION

1.1 General Introduction

1.1.1 Background

Failure data from repairable systems arise and are of interest in many industrial applications. One feature for a repairable system is that once there is a component failure, the part replacement will restore the system operation. Often it is appropriate to assume that the replaced component has the same lifetime distribution as the old component. Then the failures at a particular location (which we call a "slot") can be modeled by a renewal process. Monitoring such systems provides recurrent event data . Examples of these systems include engine valve seats in a locomotive, gearboxes in a wind turbine, light bulbs in an aircraft, etc. Nelson (2003) and Cook and Lawless (2007) present many examples of the recurrent event data and describe several modeling and analysis methods for such data.

In practice, it is common that what is observed is a collection of similar repairable systems (e.g., multiple cars/locomotives/aircrafts in a fleet). Typically the failure data are available in an aggregate form, which means we know the event times for replacements, but the information about the system identity for the replacement or component within a system is unavailable. The reason this can happen is because many databases exist for financial reasons (e.g., the management would like to monitor and track the replacement cost for the fleet of vehicles), but the missingness of either system or slot level identity information causes difficulty in the reliability analysis to estimate the underlying failure-time distribution of components.

1.1.2 Data Structures

There are two major types of data structures corresponding to the failure data in the aggregated form, described in Section 1.1.1. These types of data structures can be described by two scenarios, which we call scenario 1 and scenario 2. The following two subsections briefly describe these two scenarios.

1.1.2.1 Scenario 1: A SRP of Multiple Slots

For a fleet of wind turbines, an event of interest is gearbox failure and replacement. Each wind turbine has one gearbox and for any replacement, we do not know which wind turbine the gearbox belongs to. Then the replacement history for the whole fleet can be represented by a superposition of renewal processes (SRP) that lacks of slot-level information. Usually an SRP like this has a large number of slots, corresponding to a large number of wind turbines that a company operates.

1.1.2.2 Scenario 2: A Fleet of SRP's

Meeker and Escobar (1998, page 401) describe a fleet of locomotives that has 120 diesel engines, each having 16 cylinders. Then each engine, which can be considered as a system, corresponds to an SRP. The whole fleet then is a union of 120 SRP's. Here the system-level information is available for each cylinder replacement, but it it unknown which cylinder within the engine (slot within the system) had the replacements. In many applications the SRP in this scenario will have a relatively small number of slots, and only a few events before the end-of-observation time. The entire data set will usually consist of a number of SRP's, one from each system in the fleet.

1.1.3 Motivation

As pointed out in Condra (1993), reliability can be defined as "quality over time". Therefore the component lifetimes are of particular interest to the manufactures. Manufacturers have incentive to develop long-lasting and highly reliable products to have a strong competitive advantage in the market. One important topic in reliability analysis is to estimate the component's failure-time distribution. The failure data described in Sections 1.1.2.1 and 1.1.2.2 contain information about the component's failure-time distribution. A simple distribution study, however, is not possible due to the missing information. Therefore there arises the need to develop alternative methods that can deal with the above-described failure data with missing information and estimate the lifetime distribution of the component. The following subsections describe the three major parts in this dissertation.

1.1.3.1 Nonhomogeneous Poisson Process Modeling of an SRP

This project is motivated by the scenario described in Section 1.1.2.1. In this project, we described a procedure for estimating the component lifetime distribution from the aggregated event data for a collection of repairable systems. We formulated the observed event process as a superposition of renewal processes (SRP) and modeled the SRP by a nonhomogeneous Poisson process (NHPP). Monte Carlo simulations show that when the number of slots in the SRP is relatively large, the proposed estimator performs well and, in some settings, can recover most of the unavailable information about the slot identity. In addition, we discussed an interval estimation procedure for the distribution parameters and quantiles, based on the parametric transformed bootstrap-t method. This procedure produces coverage probabilities close to the desired nominal values when the number of slots in the SRP is large.

We compared the performance of the proposed NHPP_{MLE} estimator with that of the alternative renewalEquation estimator and we showed that the proposed estimator outperforms the alternative in general. Based on the result, we made recommendations about which estimator to use for analyzing the SRP data. We hope the recommendation provides some guidance for statisticians and engineers, allowing better reliability analysis from SRP data.

1.1.3.2 Maximum Likelihood Estimation for a Collection of Independent SRP's

This project is motivated by the scenario described in Section 1.1.2.2, but is useful for dealing with both types of data structures.

We derived the likelihood function for the observed recurrent data for each SRP by considering all unique data configurations that could lead to the observed data, and then computing the likelihood for the observed events of the whole fleet as a weighted sum of the conditional likelihoods. By starting from a crude estimate of the weights and updating iteratively, we were able to obtain the ML estimates in an efficient way. The proposed estimation method is especially useful when the number of events for each SRP is relatively small, so that likelihood computation is feasible despite the complex combinatorial nature of the problem, but the number of systems is sufficiently large, so that the total number of events is large enough to give adequate statistical precision.

In addition, we evaluated the performance of two interval estimation procedures. One is based on the Wald approximation, and the other is based on the likelihood ratio test (LRT). Our simulations show that the LRT-based procedure tends to outperform the Wald-based procedure in producing better coverage properties.

1.1.3.3 Implementation of a Maximum Likelihood Procedure for Modeling the Superposition of Renewal Process Data

The estimation method described in Section 1.1.3.2 is complicated because we have to systematically decompose the problem into several levels and enumerate all the possible partition-data configuration combinations. The method is computationally intensive. We developed an R function to implement the estimation procedure. The function takes the recurrence event data as the input and returns the ML estimates and the estimated covariance matrix of the parameter estimates. The function allows the users to specify algorithm-control parameters in an flexible way. In this paper we describe the important building blocks of the function and give examples to show how to use the function.

1.1.4 Dissertation Organization

This dissertation consists of three main chapters, preceded by this general introduction and followed by a general conclusion. Each of these main chapters corresponds to a paper to be submitted for publication. Chapter 2 describes the modeling of the superposition of renewal processes by a nonhomogeneous Poisson process. Chapter 3 presents the estimation of a parametric component lifetime distribution from a collection of superimposed renewal processes using maximum likelihood estimation. Chapter 4 documents an R implementation of a maximum likelihood algorithm for estimating the component lifetime distribution from a collection of superimposed renewal processes.

References

Condra, L.W. (1993). *Reliability Improvement with Design of Experiments.* New York: Marcel Deeker.

Cook, R.J., Lawless, J.F., (2007), *The statistical analysis of recurrent events.* New York: Springer.

Nelson, W.B., (2003), *Recurrent Events Data Analysis for Product Repairs, Disease Recurrences, and Other Applications*, ASA-SIAM, Philadelphia, ASA, Alexandria.

CHAPTER 2. ESTIMATING A PARAMETRIC LIFETIME DISTRIBUTION FROM SUPERIMPOSED RENEWAL PROCESS DATA BASED ON NONHOMOGENEOUS POISSON PROCESS MODELING

A paper to be submitted

Ye Tian and William Q. Meeker

Department of Statistics

Iowa State University

Ames, IA 50014

Abstract

This paper proposes a procedure for estimating the component lifetime distribution from the aggregated event data from a collection of repairable systems. The observed event data are assumed to be a superposition of renewal processes (SRP). We model the SRP with a nonhomogeneous Poisson process (NHPP) model. In addition, we propose and evaluate an interval estimation procedure for the renewal process distribution parameters and quantiles. Extensive Monte Carlo simulations show that when the number of slots in the SRP is relatively large, the proposed estimator performs well. Also, the interval estimation procedure has coverage probabilities close to the desired nominal values.

Keywords

Bootstrap-t; Lifetime analysis; Maintenance data; Maximum likelihood estimation; Superposition of renewal processes; Warranty data

2.1 Introduction

2.1.1 Background and Motivation

Failure data from repairable systems are of interest in many industrial applications. One feature for a repairable system is that once there is a component failure, the part replacement will restore the system operation. Often it is appropriate to assume that the replaced component has the same lifetime distribution as the old one. Then the failures at a particular location (which we call a "slot") can be modeled by a renewal process. Monitoring such systems provides recurrent event data that can be used to estimate the underlying failure time distribution of the component. Examples of these systems include engine valve seats in a locomotive, gearboxes in a vehicle, light bulbs in an aircraft, etc. In practice, it is common that what is observed is a collection of similar repairable systems (e.g., multiple cars/locomotives/aircrafts in a fleet). Typically the failure data are available in an aggregate form, which means we know the event time for replacements, but the information about the system identity for the replacement or component within a system is unavailable. A common data structure corresponding to the above scenario is a fleet of many systems with each system comprised of one or more slots and neither the slot or system level information is available. In the following we use an example to describe the situation we described above. Here we have the replacement history of transmission gearboxes for a fleet of 200 vehicles. Each vehicle has one gearbox (i.e., one slot) and when the gearbox fails, it is replaced by a new one. The actual data are proprietary, so we have simulated similar data with scaled time to protect sensitive information. Figure 2.1 shows what really happens for each vehicle, with each line representing a vehicle and each cross symbol representing a replacement. To make the plot easy to read, we only show a subset of 40 vehicles from the whole fleet in Figure 2.1. Note that in practice, we do not have detailed information about individual vehicles because many databases exist for financial reasons and will not record the exact location (i.e., system or slot within a system) of replacements. Instead, what we can observe is a superposition of the renewal processes in the individual slot(s) across all systems in the fleet, as shown in Figure 2.2. For any replacement (denoted by cross symbol) in this event plot, we don't know which vehicle it comes from. This missing information about failed system imposes difficulty in reliability analysis when there is need to study the lifetime distribution of the component. The understanding of the lifetime feature is important for many purposes, including maintenance planning for individual units and improving product design.

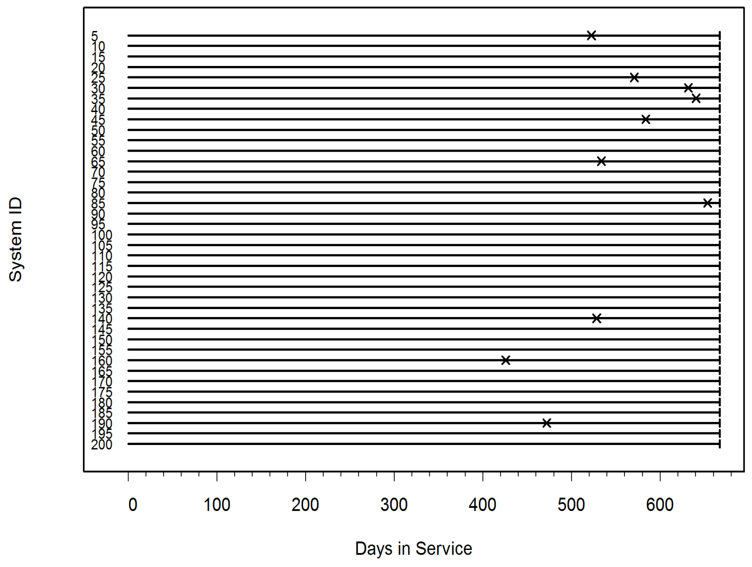

Figure 2.1 Event plot for the simulated gearbox data

In this paper we describe an efficient estimation method for estimating the lifetime distribution of a single component in a repairable system (e.g., a gearbox in a vehicle),

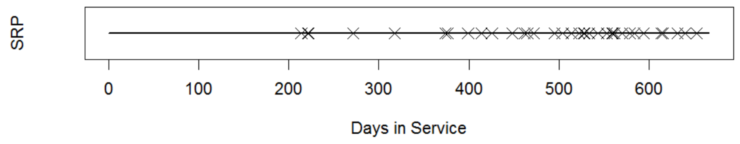

Figure 2.2 Event plot for the observed event history for the whole fleet

based on the aggregated failure data for a collection of similar systems without system identity information for the events.

2.1.2 Problem Formulation

From now on we focus on the event history for a fleet of multiple systems and each system has $m \geq 1$ slots with a component. If a component fails, it is replaced with a new one in the same slot, with the assumption that

1. The component replacement time is negligible compared to the lifetime
2. The component lifetime distribution remains for each replacement.

Then the event history of this slot can be represented by a renewal process (RP). Mathematically, for $i = 1, 2, 3, \cdots, T_i \sim$ iid $F(t; \boldsymbol{\theta})$, the sequence of $\{\mathcal{T}_j\} = \sum_{i=1}^{j} T_i$ forms a renewal process (RP), where $\mathcal{T}_j$'s are the observed renewal times (recurrence times). Here we assume the end-of-observation time is specified/fixed for each slot while the number of events before the end-of-observation time is random. The end-of-observation time may vary from slot to slot, usually due to staggered entry (or retirement) of slots into (from) the risk set. Usually the end-of-observation time for slots are the same within a system. The union of all the event histories forms a superposition of renewal processes (SRP). Denote the number of slots in the SRP by $nSlot$ and the number of events by r.

Also, the observed recurrence times for the SRP are recorded as $t_1 < t_2 < \cdots < t_r$ and for i = 1, 2, $\cdots$, $nSlot$, the i^{th} renewal process has an end-of-observation time t_c^i.

2.1.3 Related Work

Several approaches have been explored to analyze the recurrence data from an SRP and estimate the component failure time distribution. These methods are described briefly here.

2.1.3.1 Renewal Equation Method

Trindade and Haugh (1979, 1980) proposed a nonparametric estimator of the lifetime distribution from an SRP, based on the deconvolution of the renewal equation. The renewal equation states that, for any time point $t > 0$

$$F(t) = M(t) + \int_0^t M(x)\,\mathrm{d}F(t-x) \tag{2.1}$$

where $M(t)$ is the renewal expected cumulative function (ECF) of a renewal process. The ECF is defined as $M(t) = E[N(t)]$, where $N(t)$ is the number of events observed in the interval (0, t]. This method starts with the renewal equation, replaces the renewal ECF $M(t)$ by the nonparametric estimator, $\widehat{M}(t)$, then solves for $F(t)$ using numerical deconvolution, as shown below.

For simplicity, we assume a common end-of-observation time for all slots, denoted by t_c. Then we can divide (0, t_c] into n small intervals, i.e., (0, t_c/n], (t_c/n, $2t_c/n$], $\cdots$, ((n-1)t_c/n, t_c]. If n is large enough, $M(t)$ can be well approximated by a constant in each interval and the integral can be approximated by a summation of terms. Then we can substitute the empirical estimate of $M(t)$ evaluated at the midpoint (or endpoints) of each interval for $M(t)$ and calculate the CDF estimates recursively. Here $M(t)$ estimate of the k^{th} interval is $U(kt_c/n) = U((k-1)t_c/n) + \Delta U$, where ΔU is the number of events in the interval $((k-1)t_c/n,\, kt_c/n]$ for all the slots (ΔN), divided by the number of slots in the risk set (n_{risk}). The following equations demonstrate how to derive the CDF estimates, $\widehat{F}_n$,

- $t = t_c/\text{n}$

$$\begin{aligned} F_n(t_c/n) &= U(t_c/n) + \int_0^{t_c/n} U(x)\,\mathrm{d}F_n(t_c/n - x) \\ &\approx U(t_c/n) + U(0)[F_n(0) - F_n(t_c/n)] = U(t_c/n) \end{aligned} \tag{2.2}$$

- $t = 2t_c/\text{n}$

$$\begin{aligned} F_n(2t_c/n) &= U(2t_c/n) + \int_0^{2t_c/n} U(x)\,\mathrm{d}F_n(2t_c/n - x) \\ &= U(2t_c/n) + \int_0^{t_c/n} U(x)\,\mathrm{d}F_n(2t_c/n - x) + \\ &\quad \int_{t_c/n}^{2t_c/n} U(x)\,\mathrm{d}F_n(2t_c/n - x) \\ &= U(2t_c/n) + \int_{t_c/n}^{2t_c/n} U(x)\,\mathrm{d}F_n(2t_c/n - x) \\ &\approx U(2t_c/n) + U(t_c/n)[F_n(0) - F_n(t_c/n)] \\ &= U(2t_c/n) - U(t_c/n)F_n(t_c/n) \end{aligned} \tag{2.3}$$

- $t = i \times t_c/n$, for $i = 3, 4, \cdots, n$

$$F_n(it_c/n) = U(it_c/n) - \sum_{j=1}^{i-1} F_n(jt_c/n)[U((i-j)t_c/n) - U((i-j-1)t_c/n)]. \tag{2.4}$$

One problem associated with this approach is that the $\widehat{F}$ values may not be monotonically nondecreasing, which violates the definition of a CDF.

2.1.3.2 Quasi life table estimation

Baxter (1994) discussed a similar problem where periodically (e.g., every month), a group of N new components was put into operation. Direct observation of the component lifetimes is not available, only the number of failed components returned by the customers is reported at a sequence of equally spaced time intervals (e.g., each month). In the paper, the author derived a nonparametric estimator of the lifetime distribution function in a discretized manner. Let Y_n denote the number of failed components among all N slots in the n^{th} time interval, that is, $(n-1, n)$, and let u_n denote the corresponding failure probability. It is obvious that $Y_n \sim Bin(N, u_n)$ and a natural estimator of u_n is $\widehat{U}_n = Y_n/N$. However, $Y_1, Y_2, \cdots, Y_n$ are unobservable. What is observed is X_n, the number of failed components in the n^{th} time interval for all components in operation. It can be shown that X_n has the same distribution as $Y_1 + \cdots + Y_n$, so the estimator of $U_n = u_1 + u_2 + \cdots + u_n$ is $\widehat{U}_n = (Y_1 + Y_2 + \cdots + Y_n)/N = X_n/N$, where U_n is the probability of failing in the interval $(0, n)$. Based on the recursive result shown in Feller (1968, p.311) and the definition of the discretized pdf, $f_n = F_n - F_{n-1}$, Baxter (1994) calculated a nonparametric estimator of F_n as

$$\widehat{F}_n = \widehat{U}_n - \sum_{j=1}^{n-1} \widehat{F}_j(\widehat{U}_{n-j} - \widehat{U}_{n-j-1}) \tag{2.5}$$

This result is in good agreement with what Trindade and Haugh (1979, 1980) derived. Note that it is possible for $\widehat{F}_n$ to decrease over time. To remove the "violators", the author used monotone regression. Then a parametric distribution can be fit to the non-parametric CDF estimates. Baxter (1994) used a Weibull distribution, but as Tortorella (1996) has pointed out, one should use a probability plot to choose a distribution that best describes the estimates. This method provides a means to estimate the lifetime distribution parameters, but it only works well when the largest value of the $\widehat{F}_n$'s are considerably less than 1, as shown in the example in Baxter (1994). After a point in time when $\widehat{F}_n \approx 1$, the parametric estimation result will become inaccurate.

2.1.3.3 Other Methods

Cox (1962) showed that when the time is far away from the time origin, the interarrival times for the SRP begin to behave like *iid* and the asymptotic distribution in this limiting situation was derived. Kallen, Nicolai and Farahani (2010) modeled the time between repair in imperfect maintenance of coating with this asymptotic distribution. One limitation associated with this method is that it requires the system to be running for a long time before the asymptotic result becomes valid.

Tortorella (1996) proposed an estimation procedure by building a pooled discrete renewal process model and estimating the component reliability based on a maximum likelihood-like method. The author suggested a way to choose the weights required in the method, but did not provide any related results.

2.1.4 Overview

The remainder of this article is organized as follows. Section 2 describes our proposed estimation method based on NHPP modeling. Section 4 shows the performance of the proposed estimator and compares the efficiency with that of two other estimators. Section 4 gives an interval estimation method based on the parametric transformed bootstrap-t procedure. Section 5 applies the proposed quantity and interval estimation procedures to the simulated data set we discussed in Section 2.1.1. Some conclusions and the recommendations for future research are given in Section 6.

2.2 Proposed Method

Due to the limitations of existing methods described Section 2.1.3, it is desirable to develop an alternative estimation method.

2.2.1 Proposed Model

We start with the assumption that the lifetime distribution for a single component in a slot is Weibull with CDF

$$F\left(t;\beta,\eta\right)=1-\exp\left[-\left(\frac{t}{\eta}\right)^{\beta}\right]. \tag{2.6}$$

According to Khinchin's theorem as described in Khinchin (1956), an SRP behaves like a nonhomogeneous Poisson Process (NHPP) if it has a large number of superimposed renewal processes (corresponding to many slots in our problem). One appealing property of this result is that the SRP does not need to have been running for a long time to reach the steady state, as required by the asymptotic distribution method described in Cox (1962). Based on this theorem, we propose to model the SRP with an NHPP. This will provide a method to obtain a likelihood function of the observed SRP and allow the use of maximum likelihood estimation.

The NHPP is characterized by its ECF or its recurrence rate. The ECF is used to describe the expected cumulative history of a repairable system. Mathematically, the process ECF is $\mu(t) = \mathrm{E}\left[N(t)\right]$, where $N(t)$ is the observed number of events occurring before time t. In the NHPP framework, the ECF represents the expected number of events observed in the interval $(0, t]$, as

$$\mu(t) = \int_0^t \nu(t)\, dt \tag{2.7}$$

where $\nu(t)$ is the NHPP recurrence rate at time t. There are several commonly-used NHPP recurrence rate functions including the power law model and the loglinear model, as described in Meeker and Escobar (1998). In our model we will use a particular functional form that corresponds to an SRP with an underlying Weibull distribution for time between failures in each renewal process. Note that an SRP is a union of multiple single renewal processes (RP's) and so the ECF and recurrence rate function of the SRP can be computed as the sum of renewal ECFs and sum of renewal intensity functions from single RP's. For an SRP where all slots have a common end-of-observation time,

$$\mu(t;\boldsymbol{\theta}) = \mathrm{E}\left[N(t;\boldsymbol{\theta})\right] = \mathrm{E}\left[\sum_{i\in R(t)} N_i(t;\boldsymbol{\theta})\right] = \sum_{i\in R(t)} \mathrm{E}\left[N_i(t;\boldsymbol{\theta})\right] = \sum_{i\in R(t)} M_i(t;\boldsymbol{\theta}), \tag{2.8}$$

where $M_i(t;\boldsymbol{\theta})$ is the renewal ECF for the i^{th} renewal process (slot) and $R(t)$ is the index of slots in the risk set at time t. The recurrence rate function of the SRP, as the derivative of the ECF, according to Meeker and Escobar (1998, page 395) and Cook and Lawless (2007, page 12 and 40), is

$$\nu(t;\boldsymbol{\theta}) = \sum_{i\in R(t)} m_i(t;\boldsymbol{\theta}), \tag{2.9}$$

where $m_i(t;\boldsymbol{\theta})$ is the renewal intensity function for i^{th} renewal process. Note the risk set in (2.8) may change over time if different slots in the SRP have different end-of-observation times. In this situation, the expression (2.9) is unchanged. We need expressions for functions $M_i(t)$ and $m_i(t)$ for a single RP. In Section 2.2.2 we will introduce an approximation algorithm for computing m_i and M_i, and in Section 2.2.3 we will derive the likelihood for the SRP as a function of the two Weibull parameters, based on the NHPP approximation assumption.

2.2.2 Renewal ECF and Renewal Intensity Function for a Single Renewal Process

Jiang (2008) introduced a simple way to compute the renewal ECF $M(t)$ and the renewal intensity function $m(t)$ for a Weibull renewal process, based on a Gamma-normal series truncation approximation. Under the Weibull assumption in (3.1),

$$M(t) \approx F(t) + \sum_{n=2}^{N_\epsilon}\left[pG\left(t;a,b\right) + q\Phi\left(\frac{t-n\mu}{\sqrt{n}\sigma}\right)\right] \tag{2.10}$$

$$m(t) = \frac{dM(t)}{dt} \approx f(t) + \sum_{n=2}^{N_\epsilon}\left[pg\left(t;a,b\right) + q\frac{1}{\sqrt{n}\sigma}\phi\left(\frac{t-n\mu}{\sqrt{n}\sigma}\right)\right], \tag{2.11}$$

where

- $G(\cdot,a,b)$ and $g(\cdot,a,b)$ are the CDF and the pdf, respectively, of the Gamma distribution with shape parameter a and scale parameter b,

- $\Phi(\cdot)$ and $\phi(\cdot)$ are CDF and pdf of the standard normal distribution,

- $\mu = E(T_i)$, $\sigma = \sqrt{\text{var}(T_i)}$, $N_\epsilon = \inf\left\{n : \Phi\left[\frac{t-n\mu}{\sqrt{n}\sigma}\right] < \epsilon\right\}$, and
- $a = \mu/\sigma^2, b = \sigma^2/\mu, p = 1 + 0.41149 \times (1-\beta), p + q = 1.$

Here ϵ controls the accuracy of the approximation and Jiang (2008) suggested the value 10^{-6}.

The renewal ECF and renewal intensity function for renewal processes with β values 1, 3 and 5 are shown in Figure 2.3.

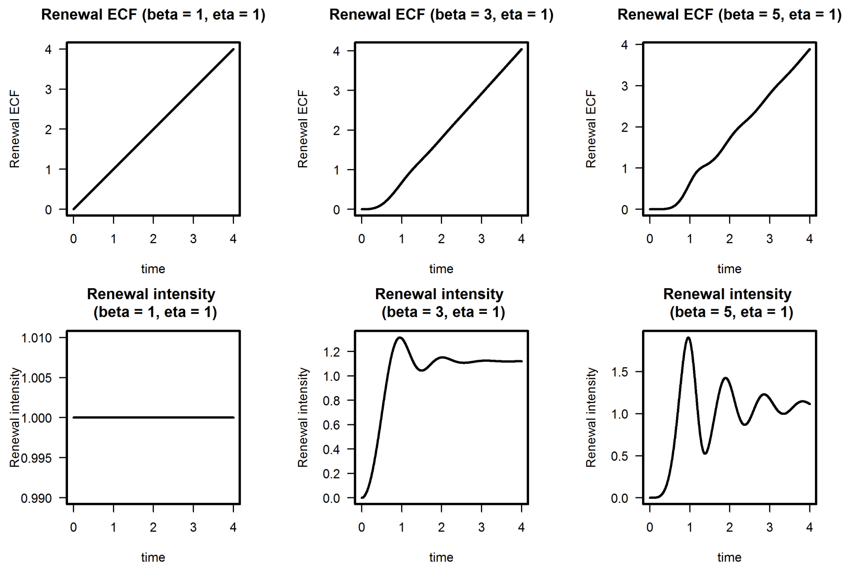

Figure 2.3 Renewal ECF and renewal intensity function for RP's with different β values. Left: $\beta = 1$; Middle: $\beta = 3$; Right: $\beta = 5$. Top row: renewal ECF plots; bottom row: renewal intensity plots

2.2.3 Likelihood

With the algorithm available for computing the (approximate) recurrence rate function and ECF of the NHPP, we are able to write out the likelihood for the observed SRP, which is approximately the likelihood of the NHPP. Following Meeker and Escobar (1998), the NHPP with recurrence times $t_1 \leq t_2 \leq \cdots \leq t_r$ and unique end-of-observation times (for different slots) $0 = t_c^0 < t_c^1 \cdots < t_c^{N_{t_c}}$ has the likelihood of

$$L_{\text{NHPP}} = \prod_{j=1}^{r} \nu(t_j; \beta, \eta) \times \exp\left\{ -\sum_{j=1}^{N_{t_c}} \left(\sum_{i \in R(t_c^j)} \left[M_i(t_c^j; \beta, \eta) - M_i(t_c^{j-1}; \beta, \eta) \right] \right) \right\} \quad (2.12)$$

where N_{t_c} is the number of unique end-of-observation times in the SRP.

We then use the maximum likelihood estimation procedure to obtain the estimates of the Weibull parameters, as well as quantiles of interest.

2.3 Comparison of the Proposed NHPP Estimator with Alternatives

In this section we describe a simulation experiment that we conducted to study and compare the performance of three methods for estimating the renewal process parameters. The three estimators in comparison are:

1. NHPP_{MLE}: The proposed estimator based on NHPP modeling, which depends on all recurrence data from the SRP
2. renewalEquation: An estimator based on the method described in section 2.1.3.2, which also needs the recurrence data from the SRP
3. cdMLE: The complete-data ML estimator assuming that we know the slot information for each event (i.e., we have complete data). This is used as a reference estimator.

2.3.1 Design of the Simulation Experiment

The simulation experiment is designed to study the effect of the following factors on the performance of the NHPP estimator:

- β: the Weibull shape parameter

- $nSlot$: number of slots in the SRP
- Er: expected total number of events in the SRP

To simplify the simulation, we assume that all slots have the same end-of-observation time. The levels of factors used were

- $\beta = 1, 1.5, 3, 4$
- $nSlot$ = 20, 200, 800
- Er = 20, 80, 160, 800

For each combination of the factor levels shown above we simulated and computed the cdMLE, NHPP_{MLE}, and renewalEquation estimates for 5000 data sets. The quantities of interest include the Weibull shape parameter β and a sequence of quantiles t_p, where p = 0.01, 0.1 $\sim$ 0.9.

2.3.2 Simulation Experiment Results

This section provides a summary of the simulation results. The primary comparison is between the NHPP_{MLE} and the renewalEquation estimators. Figure 2.4 shows relative efficiency (ratio of MSE, relative to the MSE of cdMLE) of these two estimators for different combinations of $nSlot$ and Er when β is 3. Because different β values produce similar results, we only show results based on one β. In all but exceptional cases, the NHPP_{MLE} and renewalEquation estimator will be less efficient than the cdMLE because of the information lost by not having slot identification information for the events. In all the four plots, the line at 1 represents the relative efficiency of the cdMLE, the triangle symbols represent the proposed NHPP_{MLE} estimator and the cross symbols correspond to the alternative estimator based on the renewal equation. We will compare these triangles and crosses with the line.

The top-left plot in Figure 2.4 describes the situation where $nSlot$ is large (e.g., 200) and Er is small (e.g., 20). In this situation, the NHPP_{MLE} estimator shows very strong performance for estimating all quantities of interest, in terms of high relative efficiency. Note in this situation, $nSlot$ is far larger than Er, which means there is a high chance that no slots in the SRP will have more than one event. This leads to the trivial situation where the cdMLE estimator is computed. Therefore the NHPP_{MLE} estimator has comparable performance to the cdMLE estimator. The alternative renewalEquation estimator does well in estimating β and some small quantiles (e.g., $t_{0.01}$, which is close to the fraction of failing), but the efficiency drops dramatically for higher quantiles, probably due to extrapolation.

The bottom-left plot in Figure 2.4 describes the situation where both $nSlot$ and Er are small. The performance of the NHPP_{MLE} estimator drops off, as shown in the decreasing relative efficiency when compared with the performance in the top-left plot. This finding is in good agreement with the Khinchin's theorem, which suggests that the behavior of the SRP resembles the NHPP better as $nSlot$ is larger, and therefore the likelihood shown in (2.12) becomes a better approximation to the real likelihood of the SRP. In this case, the renewalEquation estimator has comparable (or even slightly better) performance than the NHPP_{MLE} estimator.

The top-right plot in Figure 2.4 describes the situation where $nSlot$ is large (e.g., 200) and Er is relatively large (e.g., 160). Both the proposed NHPP_{MLE} estimator and the renewalEquation estimator perform relatively well, because the expected number of events in the SRP is large enough for adequate estimation. The performance of NHPP_{MLE} estimator is not as strong as shown in the top-left plot, compared to the cdMLE estimator, because with such a large number of events occurring, the cdMLE, which is based on the complete information, is theoretically optimal in the limiting situation. Here the NHPP_{MLE} has better performance than renewalEquation, by a small amount.

In the bottom-right plot in Figure 2.4, we can see that when $nSlot$ is small (e.g., 20) but Er is large (e.g., 160), neither NHPP_{MLE} estimator nor the renewalEquation estimator shows high efficiency. The relatively poor performance of the NHPP_{MLE} estimator is a result of the small number of slots in the SRP, which leads to inadequate NHPP modeling. The poor performance of the renewalEquation estimator is due to the reason that the expected number of events per slot is far greater than one, in which situation many of the nonparametric $\widehat{F}_n$'s exceeding one will be modified to be below one. This means all information after the first event in each slot will be essentially ignored. On contrary, the NHPP_{MLE} estimator can take all information into consideration, and therefore it shows slight advantage over the renewalEquation estimator.

Overall, the proposed NHPP_{MLE} estimator performs consistently better than the renewalEquation alternative over nearly all situations. The efficiency of NHPP_{MLE} estimator is usually high when $nSlot$ is large, because the NHPP modeling of the SRP is adequate. Even when $nSlot$ and Er are small, the NHPP_{MLE} estimator performs well relative to the alternative.

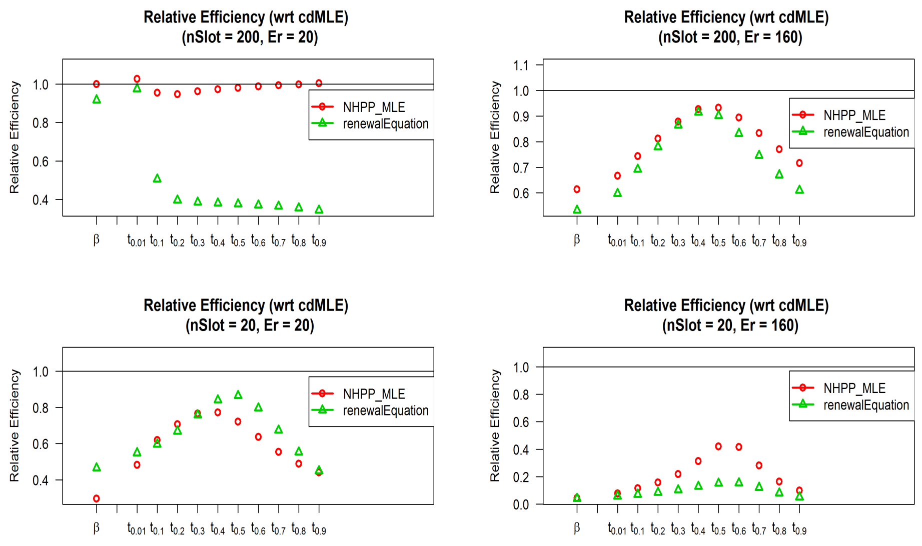

Figure 2.4 Comparison of the performance of the estimators

2.4 Interval Estimation

This section presents an interval estimation procedure based on the parametric transformed bootstrap-t method (PTBT). The quantities of interest include β and quantiles (t_p). Because all quantities of interest are positive, the bootstrap is conducted on the log scale of these quantities. The PTBT procedure is as follows:

1. Calculate the NHPP_{MLE} estimates, $\log(\widehat{\beta})$ and $\log(\widehat{\eta})$, and the corresponding estimates of standard errors, $\widehat{\text{se}}_{\log(\widehat{\beta})}$ and $\widehat{\text{se}}_{\log(\widehat{\eta})}$, by evaluating the square root of the diagonal elements of the inverse negative Hessian matrix at NHPP_{MLE}. For any $p \in (0,1)$, the NHPP_{MLE} of t_p on log scale can be computed as

$$\log(\widehat{t_p}) = \log(\widehat{\eta}) + \frac{1}{\widehat{\beta}} \log\left[-\log(1-p)\right] \tag{2.13}$$

   and the estimate of the standard error of $\log(\widehat{t_p})$ can be computed based on the estimate of the variance covariance matrix of $\left[\log(\widehat{\beta}), \log(\widehat{\eta})\right]'$ by delta method. This standard error is denoted by $\widehat{\text{se}}_{\log(\widehat{t_p})}$

2. Simulate nBoot independent SRPs according to Weibull distribution with parameters $\widehat{\beta}$, $\widehat{\eta}$ and the end-of-observation times. For each bootstrap SRP, compute $Z_{\log(\widehat{\theta}_j^*)} = \left[\log(\widehat{\theta}_j^*) - \log(\widehat{\theta})\right] / \widehat{\text{se}}_{\log(\widehat{\theta}_j^*)}$, where $\log(\widehat{\theta}_j^*)$ is the j^{th} bootstrap NHPP_{MLE} estimate of $\log(\widehat{\theta})$ and $\widehat{\text{se}}_{\log(\widehat{\theta}_j^*)}$ is the corresponding standard error estimate. Here θ can be β, η, or t_p.

3. The $100(1-\alpha)\%$ confidence interval for θ is

$$\left\{\exp\left[\log(\widehat{\theta}) + z_{\log(\widehat{\theta}^*)_{(\alpha/2)}} \times \widehat{\text{se}}_{\log(\widehat{\theta})}\right], \exp\left[\log(\widehat{\theta}) + z_{\log(\widehat{\theta}^*)_{(1-\alpha/2)}} \times \widehat{\text{se}}_{\log(\widehat{\theta})}\right]\right\} \tag{2.14}$$

   where $z_{\log(\widehat{\theta}^*)_{(\alpha/2)}}$ is the $\alpha/2$ quantile of the empirical distribution of $Z_{\log(\widehat{\theta}^*)}$

The coverage probabilities of the two-sided 90% confidence intervals and (both lower and upper) one-sided 95% confidence intervals are studied based on 4000 Monte Carlo simulations. The quantities of interest include β, $t_{0.01}$, $t_{0.1}$, $t_{0.632} \approx \eta$, $t_{0.9}$. Four sets of experiments were conducted, where $Er = 20$, and $nSlot = 20, 50, 200, 800$. The Weibull shape parameter β under study is 3.

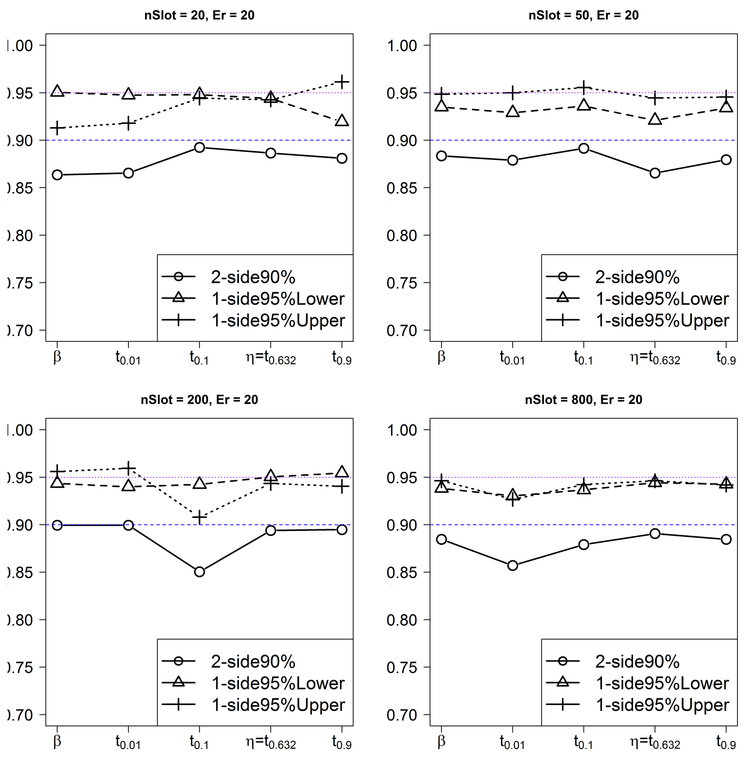

Figure 2.5 Coverage probabilities for PTBT confidence intervals with $Er = 20$ and $nSlot = 20, 50, 200, 800$

Figure 2.5 shows four plots of the coverage probabilities where the circles represent the coverage probabilities associated with two-sided 90% intervals, while triangles and crosses represent the coverage probabilities for lower and upper one-sided 95% intervals. The dashed and dotted horizontal lines show the desired coverage probabilities of 0.9 and 0.95 for the one-sided bounds and two-sided intervals, respectively. The plots from the top-left corner to the bottom-right corner describe situations where Er is fixed while $nSlot$ increases from 20 to 800. We can see that as the number of slots in the SRP ($nSlot$) increases, the coverage probabilities (both two-sided and one-sided) become closer to the nominal values. When $nSlot$ is large enough (e.g., 200 or 800), the observed coverage probabilites have good agreement with the desired values for all quantities except the quantile near the fraction of failing. The good agreement is due to the fact that the approximation of NHPP to the SRP becomes adequate as the number of slots gets larger, and this makes the likelihood more accurate. The exceptions are due to the discreteness in the data sample space, leading to an approximate discreteness in the sample space for the ML estimator of some quantities. For example, when $nSlot = 200$ and $Er = 20$, the fraction of failing will be around 0.1, and correspondingly the coverage probabilities observed for confidence intervals for $t_{0.1}$ deviate from the desired values. A detailed description of this discreteness problem, in a different setting, is given in Section 7 of Jeng and Meeker (2000).

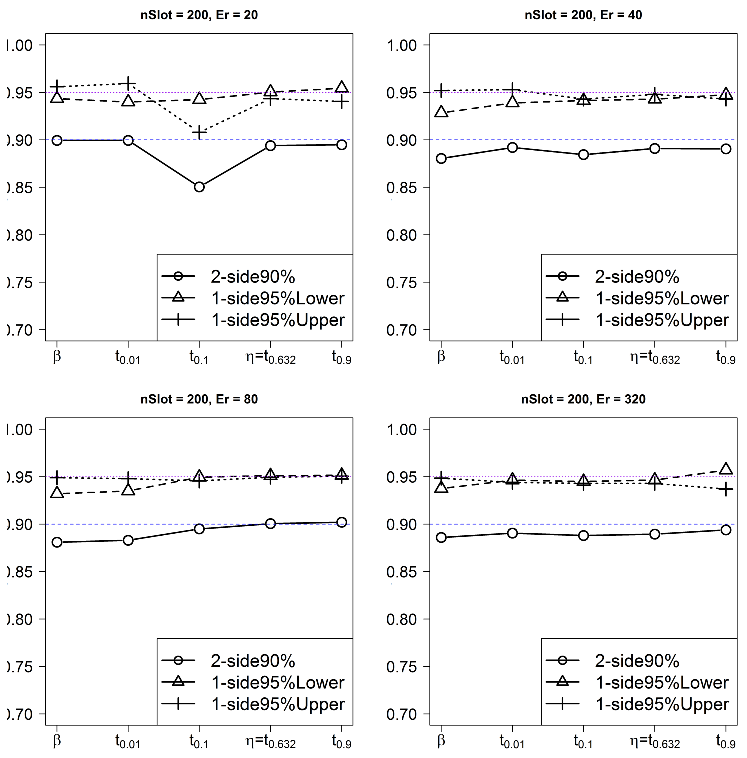

Figure 2.6 Coverage probabilities for PTBT confidence intervals with $nSlot = 200$ and $Er = 20, 40, 80, 320$

We also studied how the coverage properties behave when the SRP has more data. Figure 2.6 shows four plots of coverage probabilities for scenarios with increasing Er but constant $nSlot$. The plots from top-left corner to the bottom-right corner describe situations where $nSlot = 200$ and Er is increased from 20 to 320. As the expected number of events in the SRP (i.e., Er) becomes larger, the corresponding coverage probabilities get closer to the nominal values. This is because as Er gets larger, the distribution of Z_{θ^*} becomes more continuous. Therefore the discrete behavior begins to disappear and the asymptotic theory provides a better description of the behavior of the procedure.

2.5 Application to the Simulated Gearbox Data

In this section, we apply the proposed NHPP$_{\text{MLE}}$ estimator and the PTBT-based interval estimation procedure to the simulated gearbox SRP data shown in Figure 2.2. Here is a description of the data

- The fleet has 200 slots
- There are 37 events in the SRP. Thus the mean number of events per slot is 37/200 = 0.185

The point estimates and the confidence intervals for the two Weibull parameters, β and η, as well several quantile of interest, $t_{0.001}$, $t_{0.01}$, $t_{0.1}$ and $t_{0.2}$, are shown in Table 2.1. In this example the number of slots in the SRP is 200, large enough for the NHPP modeling to be adequate, thus the points estimates should be of high quality.

Figure 2.7 is the Weibull probability plot of the NHPP$_{\text{MLE}}$ estimates for a series of quantiles, together with the 90% confidence bands.

The empirical ECF in solid curve and the fitted ECF based on the NHPP modeling, in dotted curve, are shown in Figure 2.8. The confidence bands are also shown in the plot in dashed curves. The fitted ECF fits the empirical ECF well.

Table 2.1 Point estimates and PTBT confidence intervals for parameters and quantiles

Quantities	Estimate	5%PTBT CI	95%PTBT CI
β	3.009	2.399	4.026
η	1136.3	905.9	1311.8
$t_{0.001}$	114.4	81.3	209.3
$t_{0.01}$	246.3	198.5	348.0
$t_{0.1}$	537.9	492.4	602.6
$t_{0.2}$	690.2	628.1	755.4

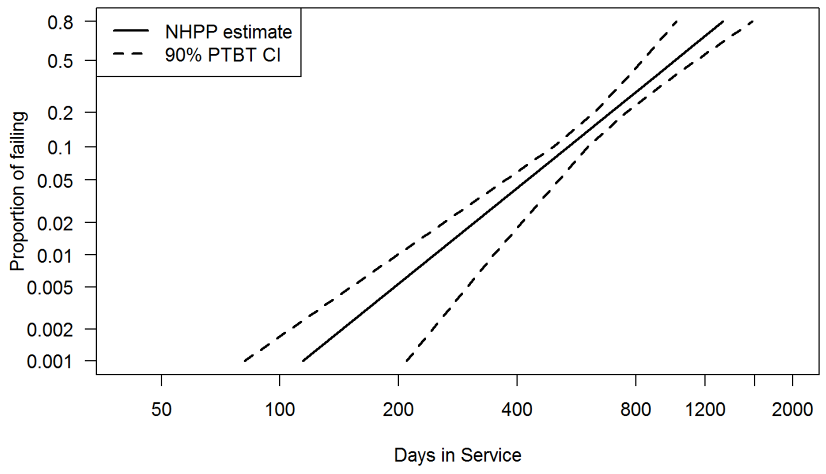

Figure 2.7 Weibull probability plot of the NHPP$_{\mathrm{MLE}}$ estimates and the 90% confidence bands

2.6 Concluding Remarks and Areas for Future Research

In this paper, we described a procedure for estimating the component lifetime distribution from the aggregated event data for a collection of repairable systems. One feature for this type of data is that we do not have the slot identity for the events. We formulated the observed event process as a superposition of renewal processes (SRP) and modeled the SRP by a nonhomogeneous Poisson process (NHPP). Monte Carlo simulations show that when the number of slots in the SRP is relatively large, the proposed

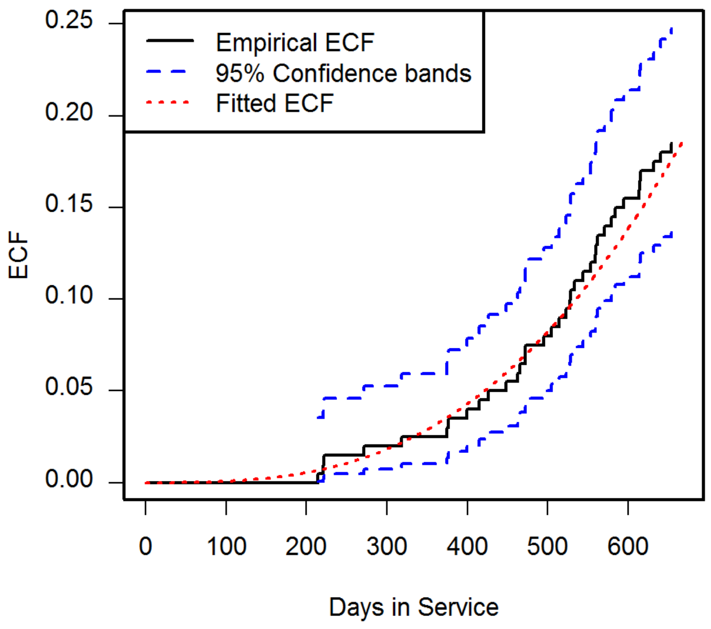

Figure 2.8 Empirical and fitted ECF plot of the simulated SRP with nonparametric confidence bands

estimator performs well for some exceptional situations (e.g., when the number of slots in the SRP is large while the number of events is relatively small). The performance can be nearly as good as when complete data are available. In addition, we discussed an interval estimation procedure for the distribution parameters and quantiles, based on the parametric transformed bootstrap-t method. This procedure produces coverage probabilities close to the desired nominal values when the number of slots in the SRP is large.

We compared the performance of the proposed NHPP_{MLE} estimator with that of the alternative renewalEquation estimator. As discussed in Section 2.3.2, we make the following recommendations about which estimator to use for analyzing the aggregated data we have discussed. The Recommendations are formed in Table 2.2. Here $nSlot$

still denotes the number of slots in the SRP and *nSample* denotes the total number of events in the observation. We hope the recommendation provides some guideline for statisticians and engineers for more precise reliability analysis.

Table 2.2 Recommendations for choosing estimators of aggregated data

	Small $nSlot$	Large $nSlot$
Small $nSample$	renewalEquation (if $nSample/nSlot \leq 1$)	NHPP_{MLE}
Large $nSample$	NHPP_{MLE}	NHPP_{MLE}

Some possible areas for future research include:

1. Development of alternative methods for situations with small *nSlot*, or modification of the NHPP procedure to allow for adequate estimation when *nSlot* is small.

2. Application of the NHPP procedure to distributions other than Weibull. The NHPP procedure can be easily adapted to other distributions if the renewal ECF expression for the renewal process is available under the specified distributional assumption.

References

Baxter, L., (1994), Estimation from quasi life tables, *Biometrika*, 81, 3, 567-577

Cook, R.J., Lawless, J.F., (2007), *The Statistical Analysis of Recurrent Events.* New York: Springer.

Cox, D. R., (1962), *Renewal Theory*,London: Methuen & Co.

Feller, W., (1968), *An Introduction to Probability Theory and Its Applications*, vol.1, New York: John Wiley

Jeng, S.L., Meeker, W.Q., (2000), Comparisons of approximate confidence interval procedures for type I censored data. *Technometrics*, 42, 135-148

Jiang, R., (2008), A Gamma-normal series truncation approximation for computing the Weibull renewal function. *Reliability Engineering and System Safety*, 93, 616-626

Khinchin, A.Ia.,(1956). On Poisson streams of random events. *Theory of Probability and Its Applications*, 1, 248-255

Meeker, W.Q., Escobar, L.A., (1998), *Statistical Methods for Reliability Data* New York: John Wiley & Sons.

Trindade, D.C. and Haugh, L.D., (1979), Nonparametric estimation of a lifetime distribution via the renewal function, *IBM Burlington Technical Report* TR 19.0463

Trindade, D.C. and Haugh, L.D., (1980), Estimation of the reliability of components from field renewal data, *Microelectronics Reliability*, 20, 205-218

Tortorella, M., (1996), Life estimation from pooled discrete renewal counts, in Jewell, N.P. *et al*. (Eds), *Lifetime Data: Models in Reliability and Survival Analysis*, Kluwer Academic, Dordrecht, pp. 331-8.

CHAPTER 3. ESTIMATING A PARAMETRIC COMPONENT LIFETIME DISTRIBUTION FROM A COLLECTION OF SUPERIMPOSED RENEWAL PROCESSES

A paper to be submitted

Ye Tian and William Q. Meeker
Department of Statistics
Iowa State University
Ames, IA 50014

Luis A. Escobar
Department of Experimental Statistics
Louisiana State University
Baton Rouge, LA 70808

Abstract

This paper proposes a procedure for estimating the component lifetime distribution using the aggregated event data from a fleet of systems. Typically a fleet contains multiple systems, with each system a set of replaceable components. For each component replacement event, we have the system-level information that components were replaced, but do not know which particular components were replaced. Thus the observed data is a collection of superpositions of renewal processes (SRP), one for each system in the fleet. We show how to compute the likelihood function for the SRP and provide suggestions for more efficient computations. We compare performance of this incomplete-data ML estimator with an alternative method that is based on a nonhomogeneous Possion process approximation and with the complete-data ML estimator.

Keywords

Likelihood ratio test; Maximum likelihood estimation; Recurrence data; Superposition of renewal processes; Wald approximation

3.1 Introduction

3.1.1 Background

Repairable systems arise and are of particular interest in many industrial reliability applications. Normally when there is a failure of a repairable system, a component replacement will restore the system operation. If we assume that a replaced component has the same lifetime distribution as the old one, then the observed recurrent event data can be represented by a renewal process. In practice, it is common that the system under observation contains a collection of similar replaceable components (e.g., valve seats or cylinders in a diesel locomotive engine). Typically the replacement data are available in an aggregate form (i.e., event time for each replacement is available, but we do not know which component underwent the replacement). In this case the aggregate data form a superposition of renewal processes (SRP).

3.1.2 Example

Nelson (2003) presents the recurrence data of a fleet of 120 diesel engines. Each engine has 16 cylinders and the cylinders can develop problems leading to leaks or low compression. Figure 4.1 shows the event plot for the cylinder replacement for a subset of 30 engines. The event plot tells us which engine (system-level information) each replacement comes from, but not the information about which cylinder position (or "slot") inside the engine. The missing slot-level information makes it difficult to estimate the component failure-time distribution.

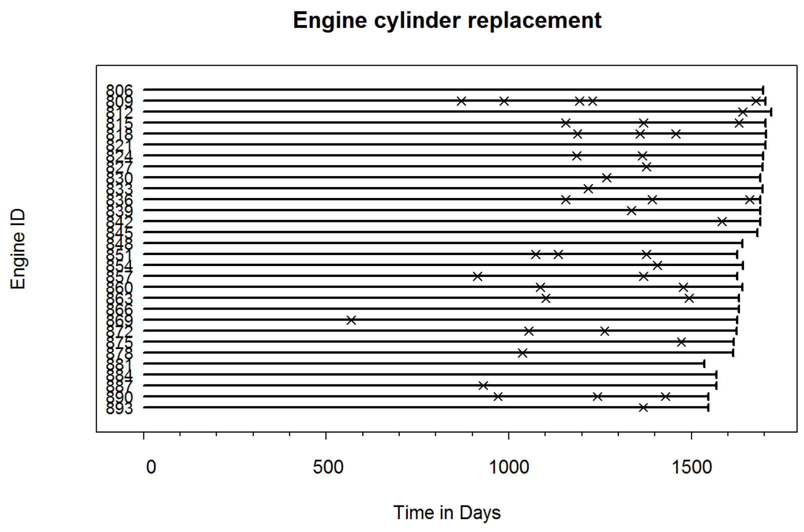

Figure 3.1 Event plot of diesel engine cylinder replacement

In this paper we propose a method for estimating the component lifetime distribution from the aggregated event data consisting of an SRP for each system. With an assumption of component lifetime, we derive the likelihood function for the observed recurrent data for the SRP, by considering all possible allocations of the recurrent events to slots in the SRP. Then we obtain the ML estimates by maximizing the derived likelihood function.

3.1.3 Related Work

Several other approaches have been explored to analyze the recurrence data of an SRP and estimate the component failure-time distribution. Cox (1962) showed that when the time is far away from the time origin, the interarrival times for the SRP begin to behave like they are *iid* with a particular given asymptotic distribution. Kallen, Nicolai and Farahani (2010) modeled the time between repair in imperfect maintenance of coating by using this asymptotic distribution. The asymptotic result, however, can only be used when the system has been running for a long time, which is not possible in most practical applications. Trindade and Haugh (1979, 1980) proposed a nonparametric estimator of the lifetime distribution, based on the deconvolution of the renewal equation. Baxter (1994) discussed a problem where periodically (e.g., every month), a group of new components is put into operation. In this problem, the direct observation of the component lifetimes is not available and what is recored is the number of failed components returned by the customers at a sequence of equally spaced time intervals (e.g., each month). The author derived a nonparametric estimator of the lifetime distribution function in a discretized manner and the result is in good agreement with that described in Trindade and Haugh (1979, 1980). Then he fit a Weibull distribution to the nonparametric estimates. The methods proposed in Trindade and Haugh (1979, 1980) and Baxter (1994) perform poorly if the nonparametric estimates of the distribution approach 1 (e.g., if the expected number of events per slot approaches or exceeds 1). Tortorella (1996) proposed an estimation procedure by building a pooled discrete renewal process model and estimating the component reliability based on a maximum likelihood-like method.

Khinchin (1956) showed that if the number of slots in an SRP is large, the SRP behaves like a nonhomogeneous (NHPP). By modeling the SRP with an NHPP and estimating the expected cumulative function (ECF) and recurrence rate function of the NHPP by those of the observed SRP, Tian and Meeker (2013) showed how to place the problem in a likelihood framework. Maximum likelihood estimation was used to produce estimates of the distribution parameters and quantiles. Their results showed that when the SRP has a large number of slots, the estimators perform well. When the number of slots is relatively small, the estimator can perform poorly, because the NHPP approximation is not adequate.

Due to the limitations discussed above, there was need to develop a new estimation method that works in applications with a small number of slots. In the following section, we describe an estimator that does not require a large number of slots in the SRP, nor does it need the system to be running for a long time. The only limitation may come from the number of events in the SRP, that is, the computationally intensive nature of the estimation method makes it difficult to deal with SRPs that have a large number of events. Considering this, we know that the proposed method will be especially useful for dealing with a fleet of SRPs where each SRP only has a relatively small number of events. When the number of systems in the fleet is relatively large, the expected total number of events becomes large enough to enable precise estimation.

3.1.4 Overview

The reminder of this article is organized as follows. Section 2 will describe the data structure and the proposed model. Section 3 derives the likelihood function for the SRP by considering all possible data configurations. Based on the SRP likelihood, we can compute the likelihood for a whole fleet of SRPs. Section 4 compares the performance of the proposed estimator with that of the other two estimators. Section 5 describes and compares the performance of two interval estimation procedures. Section 6 applies the proposed parameter and interval estimation procedures to analyze the engine cylinder replacement data. Some conclusions and the discussion of future work will be given in Section 7. An appendix at the end of this paper will give some technical details required in the derivation of the likelihood.

3.2 Data and Model

3.2.1 Data Structure

We consider a fleet of $nSys$ systems where each system contains m components operating in m slots. When a component fails, the failed component is replaced by a new one on the same slot. For each replacement, we only know the system index (identifying the system), but not the slot where the replacement was made within the system.

3.2.2 Model

We assume that the lifetime of a component, T, follows a distribution with CDF $F(t;\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is a vector of unknown parameters. For example, the Weibull distribution has CDF

$$F\left(t;\beta,\eta\right) = 1 - \exp\left[-\left(\frac{t}{\eta}\right)^{\beta}\right] \tag{3.1}$$

With an *iid* assumption, the event history for a single slot is a renewal process (RP). Let T_i denote the lifetime for the component before replacement i, $i = 1, 2, 3, \cdots$. The observed recurrence times, $\{\mathcal{T}_j\} = \sum_{i=1}^{j} T_i$, form a renewal process (RP). Here we consider a mechanism where the end-of-observation time is specified/fixed and the number of events before the end-of-observation time is random. A data set will consist of an SRP for each system in the fleet. Here we assume that all slots have the same end-of-observation times within one system.

3.3 Likelihood

Initially we define a likelihood for an SRP from a single system. Then likelihood for the whole fleet is the product of all SRP likelihoods. The maximization of the likelihood gives the idML estimates where "id" refers to "incomplete data," because the slot-level information is not available.

3.3.1 Data Configurations

An SRP arises of m statistically independent renewal processes in m slots within a system. First we identify different data configurations leading to the observed r events for the SRP.

Result 1 *For a system (SRP) with r events, the number of all possible data configurations leading to r observed events is equal to m^r*

Proof: Any of the r events can come from all the m slots in the SRP. Then according to the "counting principle" there are m^r possible data configurations that could generate those r events. For example, Figure 3.2 shows all 3^2 data configurations for the situation where there are $m = 3$ slots and $r = 2$ events.

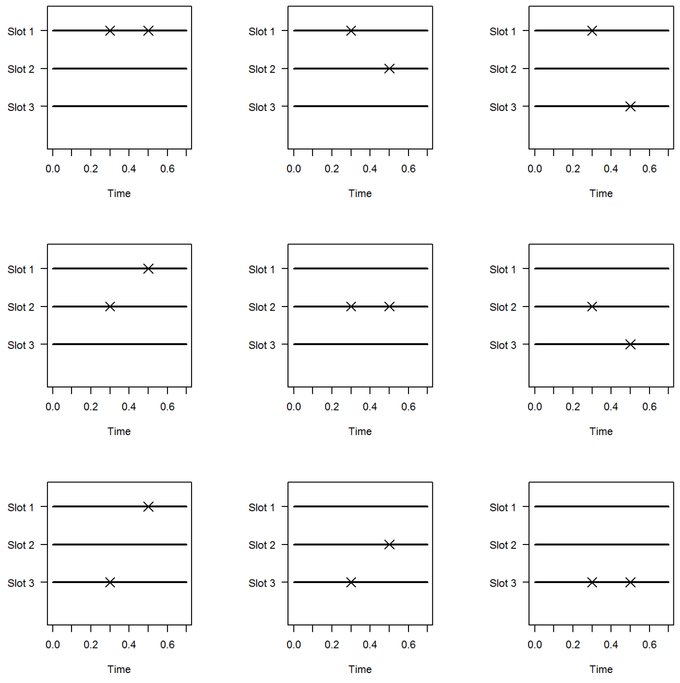

Figure 3.2 All 3^2 data configurations for the SRP with $m = 3$ and $r = 2$

In the next two subsections, we will show how to enumerate all data configurations that lead to the observed event history. This enumeration provides a basis to compute the likelihood for an SRP. Section 3.3.2 will discuss partitions of the observed r events in the SRP and all the slot combinations (i.e., the number of ways one can select slots from all m slots to which the r events can be assigned). Section 3.3.3 will describe all unique ways to allocate recurrence times to the selected slots, given one particular partition.

Table 3.1 Examples, SRP with $m = 16$ slots

r	i	Partitions of r $\mathcal{E}_i^r$	Slot Combination $k_i = \binom{m}{l_i} u_i$	Unique Data Configuration s_i
2	1	(2)	$\binom{16}{1} \times 1$	1
	2	$(1,1)$	$\binom{16}{2} \times 1$	1
3	1	(3)	$\binom{16}{1} \times 1$	1
	2	$(2,1)$	$\binom{16}{2} \times 2$	3
	3	$(1,1,1)$	$\binom{16}{3} \times 1$	1
4	1	(4)	$\binom{16}{1} \times 1$	1
	2	$(3,1)$	$\binom{16}{2} \times 2$	4
	3	$(2,2)$	$\binom{16}{2} \times 1$	3
	4	$(2,1,1)$	$\binom{16}{3} \times 3$	6
	5	$(1,1,1,1)$	$\binom{16}{4} \times 1$	1
5	1	(5)	$\binom{16}{1} \times 1$	1
	2	$(4,1)$	$\binom{16}{2} \times 2$	5
	3	$(3,2)$	$\binom{16}{2} \times 2$	10
	4	$(3,1,1)$	$\binom{16}{3} \times 3$	10
	5	$(2,2,1)$	$\binom{16}{3} \times 3$	15
	6	$(2,1,1,1)$	$\binom{16}{4} \times 4$	10
	7	$(1,1,1,1,1)$	$\binom{16}{5} \times 1$	1
6	1	(6)	$\binom{16}{1} \times 1$	1
	2	(5,1)	$\binom{16}{2} \times 2$	6
	3	(4,2)	$\binom{16}{2} \times 2$	15
	4	(4,1,1)	$\binom{16}{3} \times 3$	15
	5	(3,3)	$\binom{16}{2} \times 1$	10
	6	(3,2,1)	$\binom{16}{3} \times 6$	60
	7	(3,1,1,1)	$\binom{16}{4} \times 4$	20
	8	(2,2,2)	$\binom{16}{3} \times 1$	15
	9	(2,2,1,1)	$\binom{16}{4} \times 6$	45
	10	(2,1,1,1,1)	$\binom{16}{5} \times 5$	15
	11	(1,1,1,1,1,1)	$\binom{16}{6} \times 1$	1

3.3.2 Partitions

Definition 1 *A partition of a positive integer r is a way of writing r as a sum of positive integers.*

In our application, a partition indicates how the r events could have occurred in the m slots, without regard to time order or slot label. Table 3.1 lists examples of partitions of the integer r for $1 \leq r \leq 6$. For a given value of r, there are h partitions. The notation $\mathcal{E}_i^r$ indicates partition i out of the h partitions for the r events. Note the sum of the elements in each partition $\mathcal{E}_i^r$ is r, corresponding to the number of event(s) occurring in all m slots. For slots that have no events, the corresponding elements are 0, but they are not listed. We use a shorter representation that only lists slots with at least one event. Specifically, $\mathcal{E}_i^r$ represents a partition $(r_1, r_2, \ldots, r_{l_i})$, $l_i \leq m$, where $l_i \leq m$ is the length of the partition, corresponding to the number of slots with at least one event, as illustrated in Table 3.1.

Let u_i denote the number of unique permutations of $\mathcal{E}_i^r$. The unique elements of $\mathcal{E}_i^r$ are denoted by $(r_1^*, \cdots, r_q^*)$, and $(n_1^*, \cdots, n_q^*)$ denotes the number of times each of the q unique elements appear in $\mathcal{E}_i^r$. Then u_i can be computed as

$$u_i = \frac{l_i!}{n_1^*! \cdots n_q^*!}. \tag{3.2}$$

Given a partition with length l_i, the number of ways that one can choose l_i slots from the m slots in the SRP can be computed as

$$k_i = \binom{m}{l_i} \times u_i.$$

Thus partition $\mathcal{E}_i^r$ has k_i equivalent (except for slot labeling) slot combinations, defined as $\mathcal{A}_{i,j}^r$. Then

$$\mathcal{E}_i^r = \cup_{j=1}^{k_i} \mathcal{A}_{i,j}^r.$$

For example, for $r = 5$ and partition $\mathcal{E}_3^5 = (3, 2)$, from Table 3.1, there are $k_i = 240$ possible slot combinations. These are

$$\begin{aligned}
\mathcal{A}_{3,1}^5 &= \{(\text{Three events in slot } 1, \text{two events in slot } 2)\} \\
\mathcal{A}_{3,2}^5 &= \{(\text{Two events in slot } 1, \text{three events in slot } 2)\} \\
&\vdots \\
\mathcal{A}_{3,k_3=240}^5 &= \{(\text{Two events in slot } 15, \text{three events in slot } 16)\} \qquad (3.3)
\end{aligned}$$

All of these 240 $\mathcal{A}_{i,j}^r$'s are essentially equivalent, as they only differ in slot labels.

3.3.3 Data Configuration for a Given Partition

For each slot combination described in Section 3.3.2 (i.e., each $\mathcal{A}_{i,j}^r$), the number of ways that the observed r events could be allocated to the selected l_i slots is

$$s_i^* = \frac{\left(\sum_{j=1}^{l_i} r_j\right)!}{r_1!\, r_2! \ldots r_{l_i}!} = \frac{r!}{r_1!\, r_2! \ldots r_{l_i}!}.$$

Note that for fixed r

$$\sum_i k_i s_i^* = m^r.$$

To illustrate this, consider $r = 5$ and the partition is $\mathcal{E}_5^3 = (3, 2)$. In this partition, there are two slots with events, say slot A and slot B. Let slot A take three events and slot B take the other two events. All $s_i^* = 10$ data configurations corresponding to this partition are given in Table 3.2.

For a partition that has at least two elements of $\mathcal{E}_i^r$ that are the same, some of the s_i^* data configurations are equivalent in the sense that they produce exactly the same likelihood and differ only in the slot index. Table 3.3 illustrates this equivalence for the partition $\mathcal{E}_4^4 = (2, 1, 1)$ when there are $m = 16$ slots and $r = 4$ events. According to the partition, there are three slots with at least one event. One slot has two events and the other two slots have one event each. Suppose slot A has two events, and that slots B and

Table 3.2 All data configurations for the partition $\mathcal{E}_5^3 = (3, 2)$

j	slot A	slot B
1	t_1, t_2, t_3	t_4, t_5
2	t_1, t_2, t_4	t_3, t_5
3	t_1, t_2, t_5	t_3, t_4
4	t_1, t_3, t_4	t_2, t_5
5	t_1, t_3, t_5	t_2, t_4
6	t_1, t_4, t_5	t_2, t_3
7	t_2, t_3, t_4	t_1, t_5
8	t_2, t_3, t_5	t_1, t_4
9	t_2, t_4, t_5	t_1, t_3
10	t_3, t_4, t_5	t_1, t_2

C have one each. Then the number of all data configurations is $s_i^* = 12$, corresponding to the data configurations shown in Table 3.3.

Table 3.3 All data configurations for the partition $\mathcal{E}_i^4 = (2, 1, 1)$

j	slot A	slot B	slot C
1	t_1, t_2	t_3	t_4
2	t_1, t_2	t_4	t_3
3	t_1, t_3	t_2	t_4
4	t_1, t_3	t_4	t_2
5	t_1, t_4	t_2	t_3
6	t_1, t_4	t_3	t_2
7	t_2, t_3	t_1	t_4
8	t_2, t_3	t_4	t_1
9	t_2, t_4	t_1	t_3
10	t_2, t_4	t_3	t_1
11	t_3, t_4	t_1	t_2
12	t_3, t_4	t_1	t_1

Note, however, that data configurations 1 and 2 shown in Table 3.3 correspond to the same situation that t_1 and t_2 take place in one slot and t_3, t_4 occur in two other slots and thus will have the same likelihood. The other indicated pairs in Table 3.3 are similarly equivalent. For this partition, there are six unique data configurations.

The number of unique data configurations for partition $\mathcal{E}_i^r$, s_i, can be calculated as

$$s_i = \frac{s_i^*}{n_1^*! \cdots n_q^*!},$$

where $(n_1^*, \cdots, n_q^*)$ are the frequencies of the q unique elements in $\mathcal{E}_i^r$.

Table 3.1, for $1 \leq r \leq 6$ and $m = 16$, shows examples of partitioning integers and values of l_i, k_i, u_i and s_i. Results for SRPs with larger values of r can be computed by following the same algorithm.

3.3.4 The Likelihood for a Single SRP

In this section we drive the likelihood function for a single SRP (correponding to one system in the fleet). Then total log likelihood for the fleet is the sum of the log likelihood for the SRPs in the fleet.

Given an SRP with m slots, there are many ways to allocate observed recurrence times to all m slots, as shown in Section 3.3.1. Our goal is to consider all unique data configurations and compute the likelihoods, conditional on the particular data configurations, and the corresponding probabilities for these data configurations. Then the SRP likelihood can be computed as a weighted sum of these conditional likelihoods.

Let the observed event history be $\mathcal{H}_{t_c} = (t_1, t_2, \ldots, t_r, t_c)$, with recurrence times $t_1 < \cdots < t_r$, and end-of-observation time t_c. Denote by L the likelihood of the data. Consider an SRP with r events and let $R = \{r \text{ observed events}\}$. Then all slot combinations (i.e., $\mathcal{A}_{i,j}^r$'s) have the same probability because they correspond to the same partition and differ only in the slot label. Thus, we have

$$\pi_i = \Pr(\mathcal{A}_{i,1}^r \cap R) = \Pr(\mathcal{A}_{i,2}^r \cap R) = \cdots = \Pr(\mathcal{A}_{i,k_i}^r \cap R).$$

Result 2 *Conditional on R, the likelihood for the observed SRP is*

$$\begin{aligned}
L &= \Pr(\mathcal{H}_{t_c}|R) = \Pr\left[\mathcal{H}_{t_c} \cap \left(\cup_{i=1}^{h}\mathcal{E}_i^r\right)|R\right] \\
&= \Pr\left[\cup_{i=1}^{h}\left(\mathcal{H}_{t_c} \cap \mathcal{E}_i^r\right)|R\right] \\
&= \sum_{i=1}^{h}\Pr\left(\mathcal{H}_{t_c} \cap \mathcal{E}_i^r|R\right) \\
&= \sum_{i=1}^{h}\Pr\left[\mathcal{H}_{t_c} \cap \left(\cup_{j=1}^{k_i}\mathcal{A}_{i,j}^r\right)|R\right] \\
&= \sum_{i=1}^{h}k_i\Pr\left(\mathcal{H}_{t_c} \cap \mathcal{A}_{i,1}^r|R\right) \\
&= \sum_{i=1}^{h}k_i\Pr\left(\mathcal{H}_{t_c} \cap \mathcal{A}_{i,1}^r \cap R\right)/\Pr(R) \\
&= \sum_{i=1}^{h}k_i\Pr\left(\mathcal{H}_{t_c} \cap \left\{\cup_{j=1}^{s_i}\mathcal{B}_{i,j}^r\right\} \cap R\right)/\Pr(R) \\
&= \sum_{i=1}^{h}k_i\sum_{j=1}^{s_i}\Pr\left(\mathcal{H}_{t_c} \cap \mathcal{B}_{i,j}^r \cap R\right)/\Pr(R) \\
&= \sum_{i=1}^{h}k_i\sum_{j=1}^{s_i}\Pr\left(\mathcal{H}_{t_c}|\mathcal{B}_{i,j}^r \cap R\right)\Pr\left(\mathcal{B}_{i,j}^r \cap R\right)/\Pr(R) \\
&= \sum_{i=1}^{h}k_i\sum_{j=1}^{s_i}\Pr\left(\mathcal{B}_{i,j}^r \cap R\right)\Pr\left(\mathcal{H}_{t_c}|\mathcal{B}_{i,j}^r \cap R\right)/\Pr(R) \\
&= \sum_{i=1}^{h}k_i\sum_{j=1}^{s_i}\Pr\left(\mathcal{B}_{i,j}^r \cap \mathcal{A}_{i,1}^r \cap R\right)\Pr\left(\mathcal{H}_{t_c}|\mathcal{B}_{i,j}^r \cap R\right)/\Pr(R) \\
&= \sum_{i=1}^{h}k_i\sum_{j=1}^{s_i}\Pr\left(\mathcal{B}_{i,j}^r|\mathcal{A}_{i,1}^r \cap R\right)\Pr\left(\mathcal{A}_{i,1}^r \cap R\right)\Pr\left(\mathcal{H}_{t_c}|\mathcal{B}_{i,j}^r \cap R\right)/\Pr(R) \\
&\propto \sum_{i=1}^{h}k_i\sum_{j=1}^{s_i}p_{ij}\frac{\pi_i}{\Pr(R)}L_{ij} \\
&= \sum_{i=1}^{h}k_i\frac{\pi_i}{\Pr(R)}\left(\sum_{j=1}^{s_i}p_{ij}L_{ij}\right)
\end{aligned}$$

where $\mathcal{B}_{i,j}^r$ refers to the j^{th} unique data configuration within partition i with $\mathcal{A}^r = \cup_{j=1}^{s_i}\mathcal{B}_{i,j}^r$, $\pi_i = \Pr(\mathcal{A}_{i,1}^r \cap R)$ and thus $\pi_i/\Pr(R) = \Pr(\mathcal{A}_{i,1}^r|R)$. p_{ij} is the probability of $\mathcal{B}_{i,j}^r$ conditional on $\mathcal{A}_i^r$ with $\sum_{j=1}^{s_i}p_{ij} = 1$. L_{ij} is the likelihood for a data configuration.

In Section 3.3.5 and 3.3.6, we will discuss the computation of the π_i and p_{ij} values, respectively.

Note that $\Pr(R)$ is the probability of observing exactly r events in the m slots in the $(0, t_c]$ interval. It can be shown that $\Pr(R) = \sum_{i=1}^{h} k_i \pi_i$. In particular,

$$\begin{aligned}
\Pr(R) &= \Pr(R \cap \left(\cup_{i=1}^{h} \mathcal{E}_i^r\right)) \\
&= \sum_{i=1}^{h} \Pr(R \cap \mathcal{E}_i^r) \\
&= \sum_{i=1}^{h} \Pr(R \cap \left(\cup_{j=1}^{s_i} \mathcal{A}_{i,j}^r\right)) \\
&= \sum_{i=1}^{h} \sum_{j=1}^{s_i} \Pr(R \cap \mathcal{A}_{i,j}^r) \\
&= \sum_{i=1}^{h} k_i \Pr(R \cap \mathcal{A}_{i,1}^r) \\
&= \sum_{i=1}^{h} k_i \pi_i .
\end{aligned}$$

For a given data configuration, it is easy to write out the corresponding likelihood function, L_{ij}. For example, one data configuration corresponding to the partition $\mathcal{E}_3^5$ is shown in the first row of Table 3.2. The data configuration describes the situation where t_1, t_2, t_3 occur in slot 1, and t_4, t_5 occur in slot 2. There are no events in any of the other $(m-2)$ slots. Then the corresponding likelihood, L_{31}, is

$$L_{31} = [f(t_1)f(t_2 - t_1)f(t_3 - t_2)S(t_c - t_3)]\,[f(t_4)f(t_5 - t_4)S(t_c - t_5)]\,[S(t_c)]^{m-2} \qquad (3.4)$$

3.3.5 Computation of the Slot Combination Probabilities π_i

Let us illustrate the computation of the slot combination probabilities π_i with an example. Consider the case where $r = 5$ and the partition is $\mathcal{E}_2^5 = (3, 2)$ from Table 3.1.

Then

$$\begin{aligned}\pi_2 &= \Pr(\mathcal{A}^r_{2,1} \cap R) \\ &= \Pr(\text{Three events in slot 1, two events in slot 2, no events in other slots}) \\ &= \Pr(\text{Three events in slot 1, two events in slot 2})p_0^{(m-2)} \\ &= p_3 p_2 p_0^{(m-2)}\end{aligned} \tag{3.5}$$

where p_0 is the probability of zero events in a slot, p_3 is the probability of three events in a slot, and p_2 is the probability of two events in a slot.

In general, given $\mathcal{E}^r_i = (r_1, \ldots, r_{l_i})$, the probability of one slot combination, π_i, can be computed as

$$\pi_i = p_0^{m-l_i} \prod_{u=1}^{l_i} p_{r_u}, \tag{3.6}$$

where p_{r_u} is the probability of r_u events in a slot. The computation of (3.6) involves recursive-numerical evaluation of convolutions. See the Appendix for technical details.

3.3.6 Computation of Conditional Configuration Probabilities p_{ij}

The conditional probability of unique data configuration j given partition i is $p_{ij} = \Pr(\mathcal{B}^r_{ij} | \mathcal{A}^r_{i,1})$. The computation of the p_{ij} values appears to be extremely complicated. For each possible data configuration the probability can be expressed as a multiple integral where the dimension of the integral is $r+m$, corresponding the $r+m$ random variables involved in the needed probabilities (i.e., all of the event times and the unobserved next event times in each of the slots), conditional on the r observed events in an SRP. Furthermore the integrals need to be evaluated over a complicated region corresponding to the orderings of the random variables for the given data configuration. Given the computational complexity of this exact evaluation, we use, instead, simple simulation to do the evaluations. The simulation scheme is as follows:

Algorithm 1: Simulation of configuration probabilities

1. Simulate an SRP with m slots using the specified distribution parameters, and the given end-of-observation time, t_c.

2. If the observed number of events in the simulated SRP is not r, go to Step 1. Otherwise, check which data configuration, $\mathcal{B}_{ij}^{r}$, it belongs to, and update the count for that data configuration by 1.

3. Repeat Steps 1 and 2 until some specified number of SRPs corresponding to the second highest-probability partition have been simulated.

4. Compute p_{ij} as the observed relative frequency of each data configuration within the partition. Some partition(s) may have zero counts in the simulation because this partition has a very small probability to happen. Then we assign equal probabilities to all data configurations within this partition (i.e., $p_{ij} = 1/s_i, i = 1, \cdots, s_i$). This will not change the result too much because any such partition is rare.

3.3.7 Fleet Likelihood

For a fleet of $nSys$ systems, indexed by k, the number of slots in system k is m_k and the end-of-observation time is t_c^k. The number of recurrent events for system k is r_k and the recurrence times are $\boldsymbol{t_k} = (t_{k1}, t_{k2}, \cdots, t_{kr_k})$. Then the log likelihood for system k, $\mathcal{L}_k(\boldsymbol{\theta}; \boldsymbol{t_k}, t_c^k)$, can be computed according to Section 3.3.4. Assuming that all systems in the fleet are independent, the total log-likelihood for the fleet data is

$$\mathcal{L}_{\text{total}}(\boldsymbol{\theta}) = \sum_{k=1}^{nSys} \mathcal{L}_k(\boldsymbol{\theta}; \boldsymbol{t_k}, t_c^k). \tag{3.7}$$

3.3.8 Estimation and Computational Issues

The idML estimates are obtained by maximizing the log likelihood function shown in Equation 3.7.

The simulation procedure described in Section 3.3.6 can be computationally intensive. One approximate method is to start by assigning equal p_{ij} to all unique data configurations within a partition. This approximation works because SRPs with relatively small r but relatively large m have some dominant partition corresponding to the scenario where no slot has more than one event. In this case, there is only one unique data configuration within that dominant partition, so assigning equal p_{ij} to each data configuration gives an exact result to that partition. Also, assigning equal p_{ij}'s to other less dominant partitions will not change the likelihood too much because those partitions have a small probability and thus contribute little to the likelihood. This approximation simplifies the computation to begin the idML iterations. Based on this idea, we propose an iterative procedure to obtain the approximate idML estimates, as described below. For simplicity, we focus on one SRP. The procedure can be easily generalized to a fleet of multiple SRPs.

Algorithm 2: Approximate idML Estimation

1. Set $k = 0$

2. Give equal p_{ij} to every data configuration within a partition, and compute the first approximation to the ML estimates by maximizing (3.7). Denote the estimates by $\widehat{\boldsymbol{\theta}}^{\boldsymbol{k}} = \widehat{\boldsymbol{\theta}}^{\boldsymbol{0}}$.

3. Simulate SRPs with m slots and end-of-observation time t_c, according to the lifetime distribution with parameters $\widehat{\boldsymbol{\theta}}^{\boldsymbol{k}}$. Then use the simulated SRPs to estimate p_{ij}'s.

4. Substitute the simulated p_{ij} values obtained in last step to the likelihood function and by maximizing (3.7), compute the ML estimate, denoted by $\widehat{\boldsymbol{\theta}}^{\boldsymbol{k+1}}$.

5. Set $k = k + 1$. If $k < nIter$, go to Step 3.

6. The approximate idML estimate is $\widehat{\boldsymbol{\theta}}^{\boldsymbol{nIter}}$.

The estimate $\widehat{\boldsymbol{\theta}}^{\boldsymbol{nIter}}$ is expected to be close to real idML estimates. In order to evaluate the performance of this iterative procedure, we studied how the estimate evolves in each iteration for maxIter = 10 steps. In this experiment we assume that the component lifetime distribution is Weibull, with CDF shown in Equation 3.1. This experiment uses three factors:

- β: the Weibull shape parameter
- m: the number of slots in the SRP
- Er: the expected number of events for each SRP

The number of SRPs in the fleet, $nSys$ is constrained such that $nSys \times Er = 20$ (i.e., the expected number of events for the whole fleet is 20). The factor levels considered in this experiment are:

- β: 1, 3
- m: 4, 16
- Er: 4, 1

Note that because for a total expected number of events of 20, the fleet has $nSys = 5$ SRPs when $Er = 4$ and $nSys = 20$ SRPs when $Er = 1$.

Figure 3.3 describes the situation where $\beta = 3$ and Figure 3.4 corresponds to the situation where $\beta = 1$. Each figure has eight plots in four rows. The four rows correspond to the four combinations of the factor levels of m and Er and the two plots on the same row correspond to the estimate of β and one quantile of interest. The quantile is selected to be close to one half of the fraction of failing (or fraction of slots with events). For example, if $m = 16$ and $Er = 4$, then about 1/4 of the slots have events, and therefore we focus on $t_{0.1}$. Each plot shows iterative estimates from three realizations, denoted by different symbols. Iteration 0 represents the estimator based on equal p_{ij} specification (i.e., $\widehat{\boldsymbol{\theta}}^{\mathbf{0}}$). These figures suggest that when β is relatively large (e.g., 3), the estimate usually becomes stable after a few iterations, for both β and quantile. When

β is small (e.g., 1), we can see larger fluctuations in the estimates, probably due to the larger variability in p_{ij}'s within a data configuration, computed by simulations. Note the distributional variability is larger for $\beta = 1$ than $\beta = 3$. Therefore, it may take more steps for the estimates to stabilize. In overall, the observed trend of the iterative estimate is converging for large β values and it is expected that the convergence point is the desired idML estimate. Because the wearout failure mode (i.e., relatively large β) is of particular interest in our research, the above result indicates that this iterative procedure may lead us to some stable estimate, as an approximation to the desired idML estimate, in a few steps.

3.4 Performance of the idML Estimator

3.4.1 Design of Simulation Study

This section describes a simulation study to compare the performance of the proposed idML estimator with two other estimators. The three estimators in comparison are described below:

1. cdML: which requires the knowledge of the lifetimes of all units. This is equivalent to knowing not only the SRP information, but also the slot location for all replacements. This is used as a comparison reference.

2. idML: the proposed incomplete data estimator, $\widehat{\boldsymbol{\theta}}_{\text{idMLE}}$, which depends on having individual SRP data from each system.

3. NHPP_{ML}: which treats the whole fleet as a single SRP and applies the NHPP modeling to this SRP, as described in Section 3.1.3 and Tian and Meeker (2013).

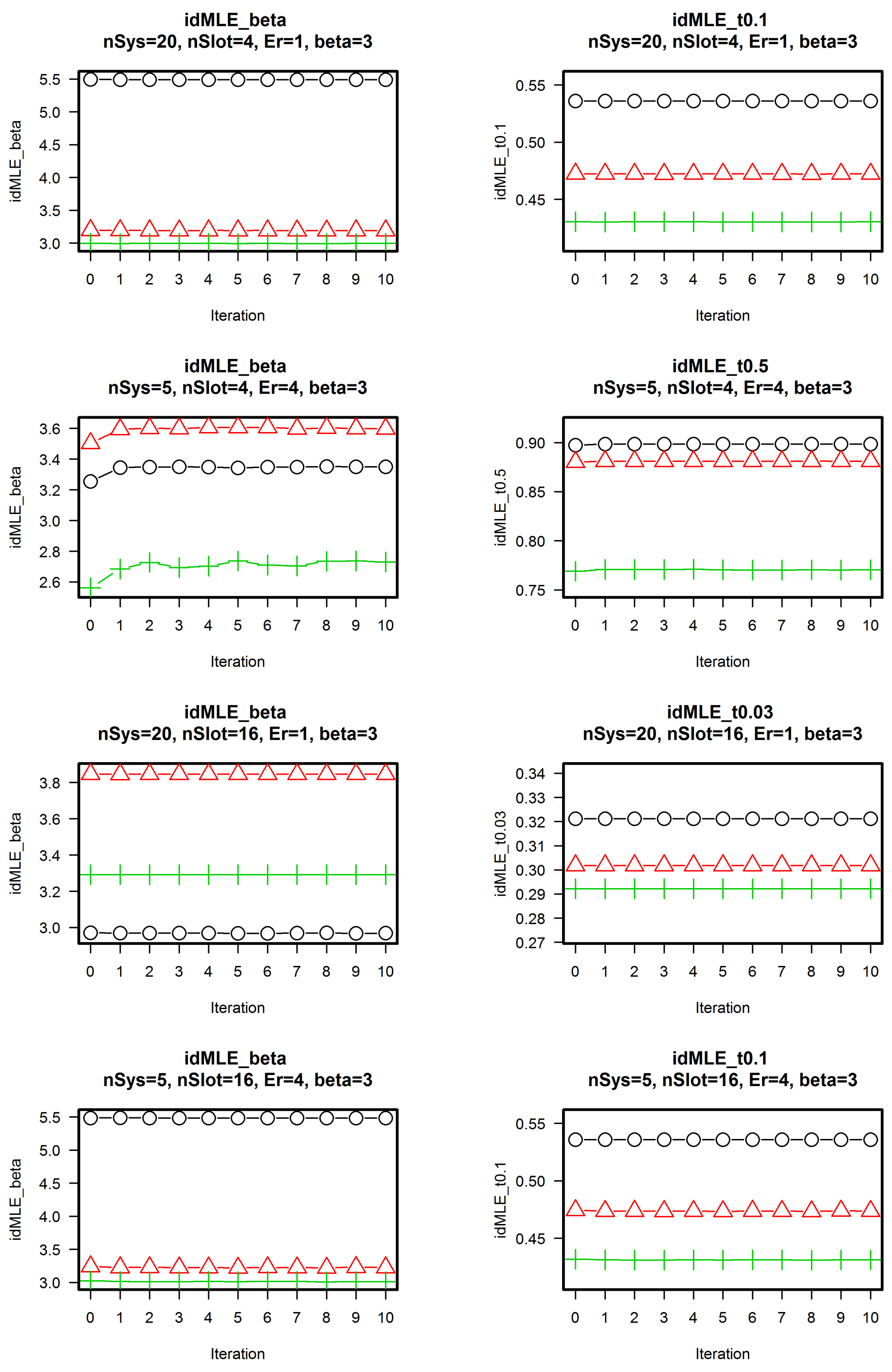

Figure 3.3 idML estimates for β (left) and t_p (right) at each iteration for different combinations of $nSys$, m and Er. Here $\beta = 3$ (3 realizations)

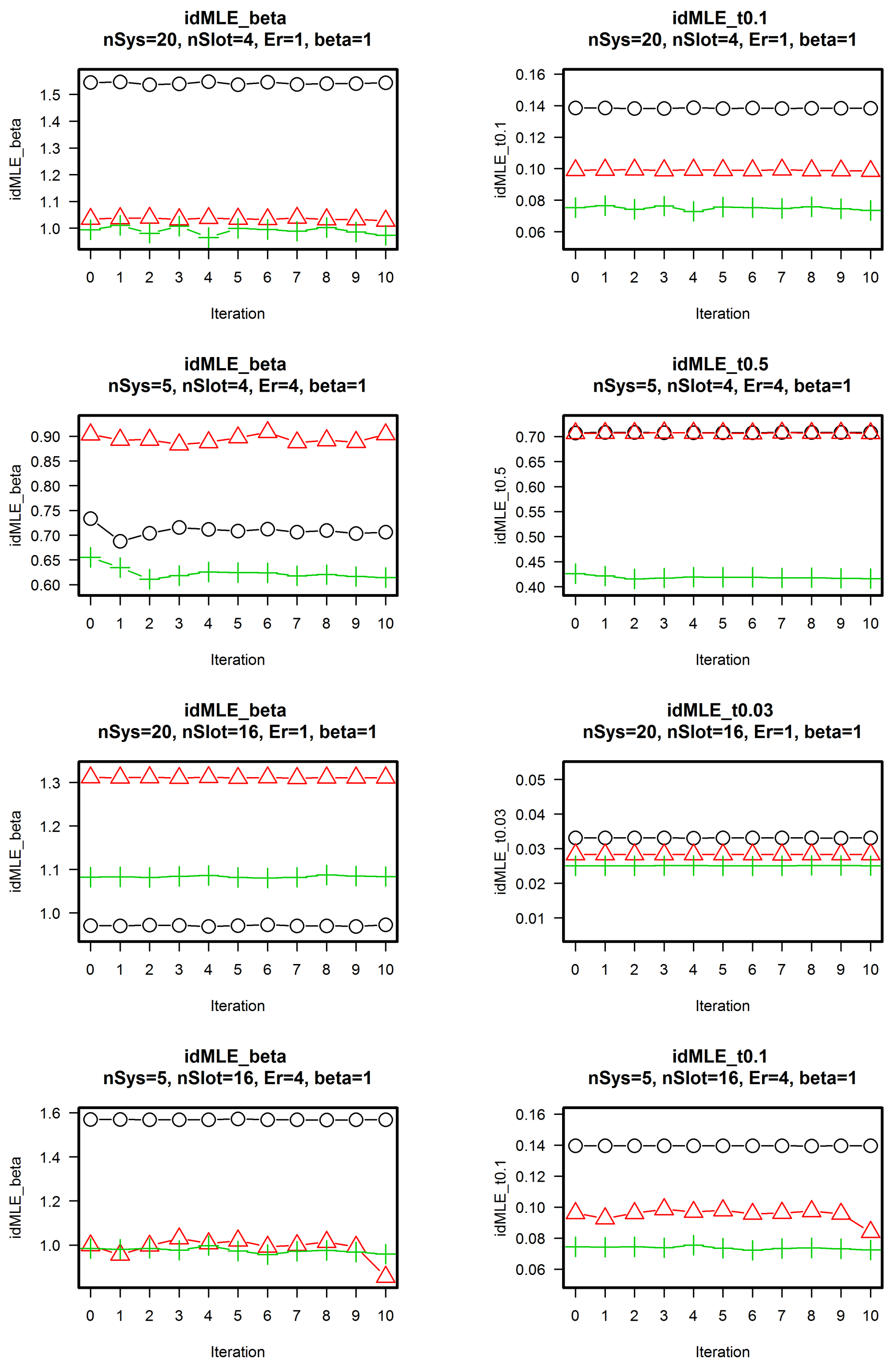

Figure 3.4 idML estimates for β (left) and t_p (right) at each iteration for different combinations of $nSys$, m and Er. Here $\beta = 1$ (3 realizations)

3.4.1.1 Other Experimental Factors

The simulation was designed to mimic the replacement history for a fleet of $nSys$ systems (SRPs). For simplicity, we assume that all systems have m slots, and the end-of-observation times for all systems are the same, as determined by the expected number of events per system. We assume a Weibull component lifetime distribution with shape parameter $\beta = 3$ and scale parameter $\eta = 1$.

The particular factors used were:

- $nSys$: the number of SRPs in the fleet
- m: the number of slots in the SRP
- Er: the expected number of events for each SRP

3.4.1.2 Factor Levels

We conducted simulations at all combinations of the following levels of the factors.

- $nSys$: 5, 15
- m: 4, 16
- Er: 4

Note that the total expected number of events for the fleet would be $nSys \times Er$ (i.e., either 20 or 60).

For each combination of the factor levels we simulated and computed idML, cdML and NHPP estimates for 1000 data sets. The quantities of interest include the Weibull shape parameter β and a sequence of quantiles t_p, where $p = 0.01$, $0.1 \sim 0.9$.

3.4.2 Simulation Results

For each quantity of interest, we calculate the Monte Carlo estimate of MSE based on the 1000 samples and quantify the relative efficiency of idML and NHPP_{ML} estimators, with regard to the cdML estimator, which is based on the complete data information. We use the cdML estimator as the reference because with all information available, this estimator is expected to perform well. The relative efficiency (ratio of MSE error, relative to the MSE of the cdML estimator) indicates how much information is lost if we do not know the slot identity for component replacements.

Figure 3.5 shows the relative efficiency plots for all the four factor-level combinations in Section 3.4.1.2. In all the four plots, cdML reference estimator is indicated by a horizontal line with an efficiency of 1.

The top-left plot in Figure 3.5 corresponds to the situation where the fleet has $nSys = 5$ systems with each system has $m = 4$ slots and the expected number failing per system is $Er = 4$. From the plot we can see that the idML estimator has high efficiency for estimating large quantiles (e.g. $t_{0.5}, t_{0.7}$). This is because each slot has one event on average, and therefore larger quantiles will be well estimated. The efficiencies for small quantiles drop gradually as they move away from $t_{0.5}$ toward small probabilities. The relative efficiency of β estimate is about 0.5. The NHPP estimator, denoted by cross symbols, does not compare well with the idML estimator, especially in estimating large quantiles and β. The performance of the NHPP estimator is comparable to that of the idML estimator in estimating small quantiles, for example, $t_{0.01}$ and $t_{0.1}$. The poor performance of the NHPP estimator can be explained by the small number of slots ($nSys \times m = 20$) in the whole fleet. With such a small number, the NHPP model does not provide an adequate description of the SRP, as mentioned in Section 3.1.3.

The bottom-left plot in Figure 3.5 describes the situation where $nSys = 15$, $m = 4$, and $Er = 4$. In this case the idML and NHPP estimators have comparable performances. The idML estimator has better performance than NHPP estimator in estimating large quantiles (e.g., $t_{0.5}$, $t_{0.7}$, $t_{0.9}$), but NHPP is slightly better than idML estimator in estimating small quantiles (e.g., $t_{0.01}$, $t_{0.1}$ and $t_{0.3}$) and β. The reason why the idML estimator does not do as well, relative to the NHPP estimator, is because the total number of slots in the fleet is $15 \times 4 = 60$, large enough for the the NHPP model to provide an adequate model for the collective SRP.

The top-right plot in Figure 3.5 corresponds to the situation where $nSys = 5$, $m = 16$, and $Er = 4$. The idML estimator of β has a slightly larger MSE than that of the cdML and NHPP estimators. The MSE for the idML estimator of the lower tail quantile, $t_{0.01}$, is larger than that of the corresponding cdML estimator, but smaller than the NHPP estimator. For quantiles from $t_{0.1}$ to $t_{0.9}$, the idML estimator has a smaller MSE than the other two estimators. After in-depth analysis of these three estimators, we learned that the idML estimator has a smaller magnitute of bias for larger quantiles (e.g., $t_{0.3}$ to $t_{0.9}$) and a somewhat smaller variance for all quantiles of interest, than the cdML and NHPP estimators. The comparison of bias, variance and MSE among cdML, idML and NHPP estimator for this situation is given in Figure 3.6, with three estimators denoted by cross, circle and triangle symbols, respectively. The idML estimator of $t_{0.1}$ has larger bias than the other two estimators, but because the variance is smaller, the MSE is still smaller than both cdML and NHPP estimators. The bias for the idML estimator of β is also larger than the other two estimators. But because the variance for idML estimator is only slighly smaller than that of cdML and NHPP estimators, the MSE is slightly larger for idML than the ther two estimators.

The bottom-right plot in Figure 3.5 refers to the situation where $nSys = 15$, $m = 16$, and $Er = 4$. The performance of the idML estimator, relative to the other two estimators, is similar to that is observed in the top-right plot. The idML estimator also has smaller MSE in quantiles from $t_{0.3}$ to $t_{0.9}$ than the other two estimators, but the difference in MSE between idML and cdML estimators is smaller than what is observed in the top-right plot. This is because the expected number of events for the whole fleet in this situation is three times the situation described in the top-right plot, allowing the cdML estimator to have decreased bias, thus smaller MSE.

Overall, the proposed idML estimator tends to outperform the NHPP estimator, especially when the number of slots is far greater than the number of event in the SRP (corresponding to the cylinder data). The advantage of idML over NHPP estimator diminishes as the number of slots in the whole fleet gets large.

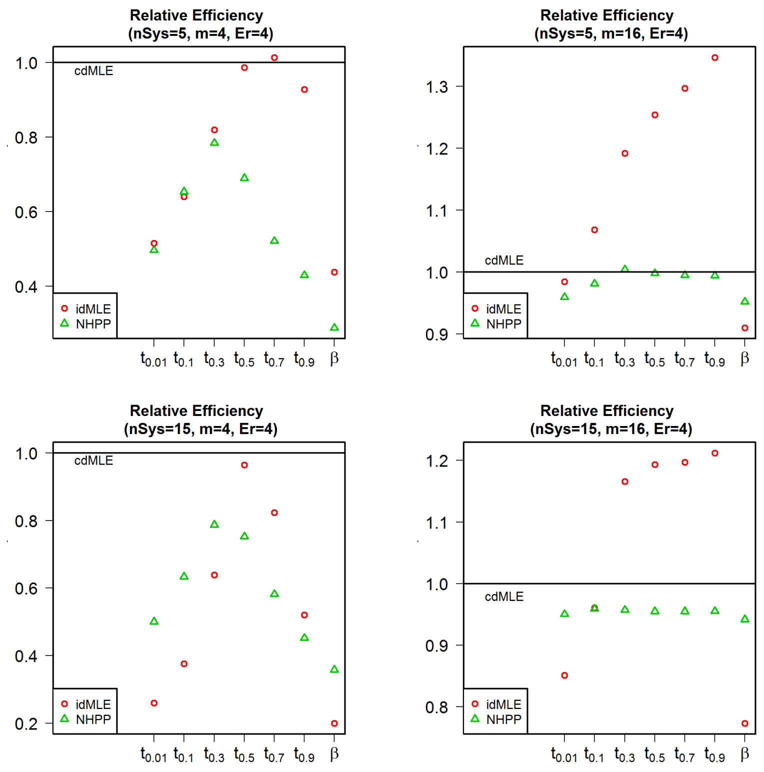

Figure 3.5 Relative efficiency of estimators w.r.t. cdMLE (top-left: $nSys = 5$, $m = 4$, $Er = 4$; top-right: $nSys = 5$, $m = 16$, $Er = 4$, bottom-left: $nSys = 15$, $m = 4$, $Er = 4$; bottom-right: $nSys = 15$, $m = 16$, $Er = 4$)

3.5 Bias of the idML Estimator

Section 3.4.2 showed that the idML estimator of β and some lower tail quantiles has larger (positive) bias than the cdML estimator. When the expected number of events is small (e.g., 4) relative to the number of slots (e.g., 16) in an SRP, the bias is small. But when the expected number of events (e.g., 4) is close to the number of slots (e.g., 4), the idML estimation procedure can have significant bias. In the following analysis we will

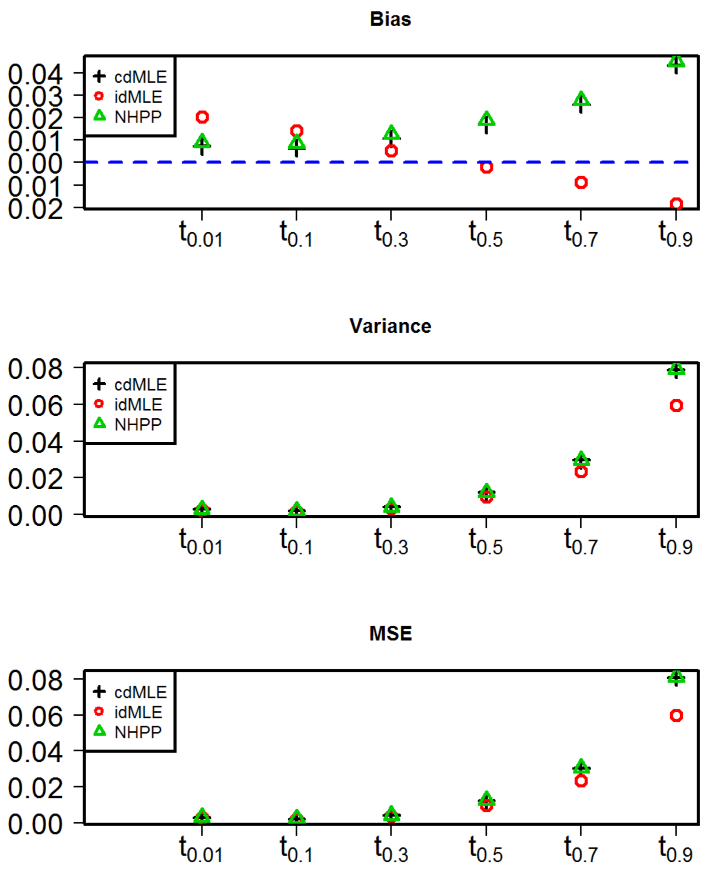

Figure 3.6 Comparison of bias, variance and MSE for 1000 samples among cdML, idML and NHPP estimates when $nSys = 5$, $m = 16$ and $Er = 4$

focus on the comparison of the idML and cdML estimators and explain why the idML estimator can have a larger bias than the cdML estimator.

The likelihood (3.4) can be expressed as a weighted sum of the likelihood for a data configuration L_{ij}. That is,

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{h} \sum_{j=1}^{s_i} w_{ij}(\boldsymbol{\theta}) L_{ij}(\boldsymbol{\theta}), \tag{3.8}$$

where $w_{ij} = k_i \pi_i p_{ij} / \sum_{i=1}^{h} k_i \pi_i$ is the probability for the j^{th} unique partition-data configuration combination within partition i and $\sum_{i=1}^{h} \sum_{j=1}^{s_i} w_{ij} = 1$. Each L_{ij} corresponds to one possible partition-data configuration combination that could lead to the observed r

events. The cdML estimate is obtained by maximizing the single L_{ij} corresponding to the known partition-data configuration combination that actually happened behind the simulation. In order to show why the idML estimator tends to have large bias, we use a simulated example to illustrate what the likelihoods (L_{ij}) look like for all partition-data configuration combinations. In the example we use a single simulated SRP with $r = 3$ events and $m = 4$ slots. In this situation, there are $\sum_{i=1}^{h=3} s_i = 5$ unique partition-data configuration combinations and the 0.9 level relative likelihood contour plots ($t_{0.3}$ vs β) for all the five partition-data configuration combinations and the idML are shown in Figure 3.7. The circle symbol represents the center of the idML 0.90 relative likelihood contour and the other five symbols indicate the center of the 0.90 relative likelihood contours for the other five partition-data configuration combinations that could occur. The partition-data configuration combination with label '1.1.1_1.2.3' (denoted by inverted triangle) represents the highest-probability (for any reasonable values of the Weibull parameters) data configuration, which means the observed three events take place in three different slots. In a long sequence of simulated SRPs, the cdML estimator would, with high probability, correspond to this data configuration. The first part of the label, '1.1.1', represents the partition, and the second part of the label, '1.2.3', corresponds to the data configuration within the partition. Interestingly, the idML estimate is not in the range of the five possible cdML estimates. The idML estimate would be in the range of the other five estimates if the weights (i.e., w_{ij}) in (3.8) were to be fixed (i.e., not a function of the Weibull parameters). Instead, the idML relative likelihood contour is shifted to the right of the other 0.90 relative likelihood contours. The idML relative likelihood contour shifts to the right because the weights w_{ij} depend on the parameters (e.g., β and $t_{0.3}$) and, in an effort to maximize the likelihood, the idML procedure pulls the β estimate to a higher value region in order to increase the weight for the highest-probability data configuration, even though the corresponding L_{ij} will decrease somewhat. The top-left and bottom-left plots in Figure 3.8 show w_{ij} values for all of the five partition-data configuration combinations as a function of β (while, for simplicity, fixing η as the cdML estimate), showing the effect that β has on the w_{ij} values for SRPs with $m = 4$ and $m = 16$ slots, respectively. Note that the w_{ij} value for the highest-probability partition-data configuration combination (i.e., '1.1.1_1.2.3') increases as β increases. The top-right and bottom-right plots of Figure 3.8 show the profile likelihoods as a function of β for the corresponding SRPs. The shift in the profile likelihood curves from the cdML to the idML procedure corresponds to the positive bias for β. The bias is smaller for an SRP with $m = 16$ slots (i.e., bottom-right plot) than that for an SRP with $m = 4$ slots (i.e., top-right plot).

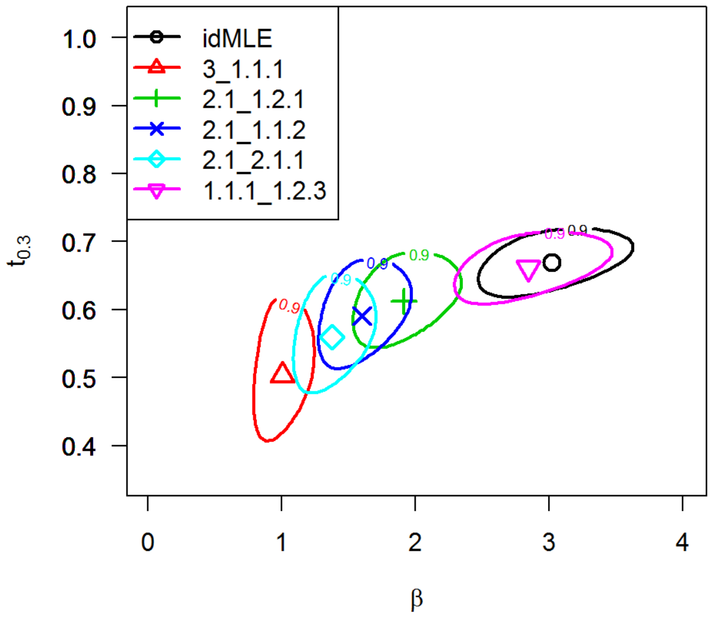

Figure 3.7 Contour plots (0.9 level) for all partition-data configuration combination and idML likelihood for a SRP with 3 events and 4 slots

The likelihood L_{ij} for the highest-probability partition-data configuration combination (i.e., '1.1.1_1.2.3') is much larger than that for the other four L_{ij} values because this

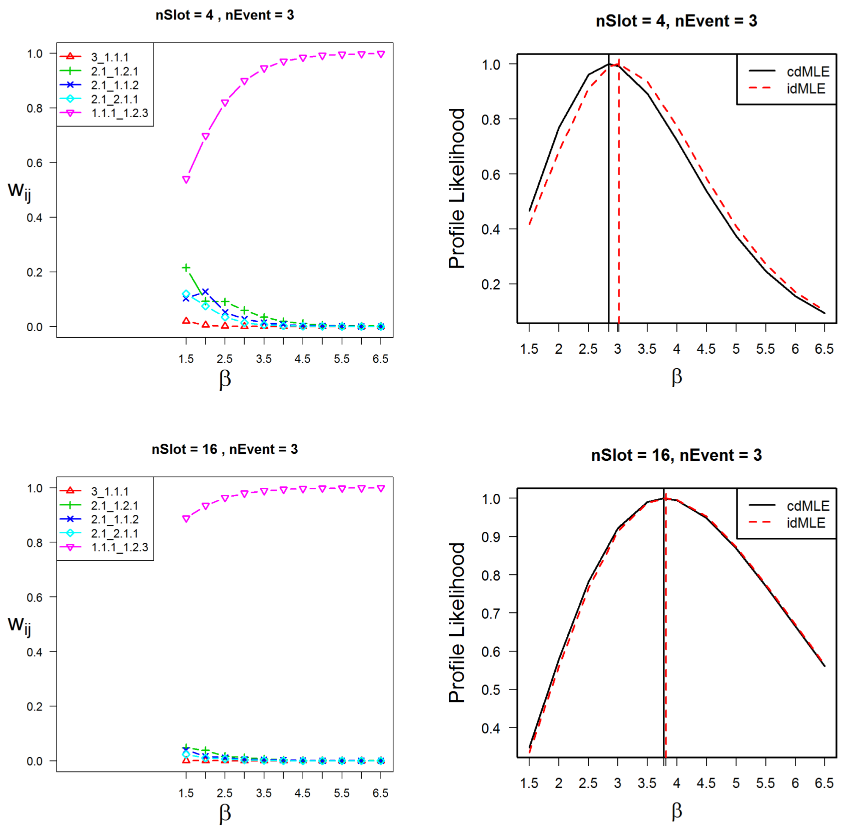

Figure 3.8 w_{ij} and profile likelihood values for all five partition-data configuration combinations for a simulated SRP with different β values (Top: $m = 4, r = 3$. Bottom: $m = 16, r = 3$. Left: w_{ij}. Right: Profile likelihood)

partition-data configuration combination is much more likely to take place. According to the top-left plot in Figure 3.8, when β is 2.5, w_{ij} for the highest-probability partition-data configuration combination is about 0.82. When β is 3, the w_{ij} value increases to 0.90, putting more weight to the largest L_{ij}. The increase in β will cause a decrease in the L_{ij} value, but because of the substantial increase in the w_{ij}, the total likelihood still increases. This is the reason why the idML β estimator tends to be larger (and thus more biased) than cdML estimator. Because the estimator of β is positively correlated with the estimators of some lower tail quantiles, the estimators of these quantiles also have positive bias.

As seen in the bottom-right plot of Figure 3.8, when the SRP has $r = 3$ events and $m = 16$ slots, the observed biases for β are much smaller than those for an SRP with $m = 4$ events. This is because in this case, a β value of 3.5 already gives a w_{ij} value of 0.989 to the highest-probability data configuration. If β is increased from 3.5 to 4, the w_{ij} value only increases from 0.989 to 0.994, not too much improvement relative to the corresponding decrease in L_{ij} value. Therefore there is no strong incentive for β to become larger and the idML estimate of β will be only slightly larger than the cdML estimate of β.

3.6 Interval Estimation

This section compares two interval estimation methods, based on the Wald approximation and based on inverting the likelihood ratio test (LRT).

3.6.1 Wald Interval

Let θ be a particular quantity of interest on the log scale. For example, θ can be $\log(\beta)$ or $\log(t_p)$, where β and t_p are the shape parameter and p quantile of the Weibull distribution. We use the log scale because all the quantities of interest are positive, so on log scale, the range of the quantity is unrestricted. Also, there is an expectation that the normal distribution approximation underlying the Wald method will be better on the unrestricted log scale. Denote the idML estimate of θ by $\widehat{\theta}$. Let $\widehat{\text{se}}_{\widehat{\theta}}$ denote an estimate of the standard error of $\widehat{\theta}$, computed as a function of the elements of the observed Hessian matrix. Then the 100(1 - α)% Wald confidence interval for θ is

$$\left[\widehat{\theta} \pm z_{1-\alpha} \times \widehat{\text{se}}_{\widehat{\theta}}\right] \tag{3.9}$$

where α is the confidence level and $z_{1-\alpha}$ is the $1 - \alpha$ quantile of the standard normal distribution.

3.6.2 LRT Interval

Let $\boldsymbol{\theta} = (\theta_1, \theta_2)$ be the unknown parameters, where θ_1 is the parameter of interest and θ_2 is a nuisance parameter. Then in our examples $\boldsymbol{\theta}$ would be $[\log(\beta), \log(t_p)]$ or $[\log(t_p), \log(\beta)]$. If the likelihood function is $L(\theta_1, \theta_2)$, then the profile likelihood for θ_1 is

$$\mathrm{R}(\theta_1) = \max_{\theta_2} \left[\frac{L(\theta_1, \theta_2)}{L(\widehat{\theta}_1, \widehat{\theta}_2)} \right] \tag{3.10}$$

where $\widehat{\theta}_1$ and $\widehat{\theta}_2$ are the idML estimates of θ_1 and θ_2, respectively. The likelihood-ratio statistics is $\mathrm{LRT}(\theta_1) = -2 \log \mathrm{R}(\theta_1)$ and the asymptotic distribution of $\mathrm{LRT}(\theta_1)$, when evaluated at the true θ_1, is χ^2_1. If the roots to the equation

$$\mathrm{LRT}(\theta_1) - \chi^2_{1,(1-\alpha)} = 0 \tag{3.11}$$

are $\theta_{1L} \leq \theta_{1U}$, then the LRT interval with a confidence level α for θ_1 is

$$[\theta_{1L}, \theta_{1U}] \,. \tag{3.12}$$

3.6.3 Comparison of Interval Estimation Methods

The coverage probabilities for confidence interval procedures were studied for both Wald and LRT based intervals. Figure 3.9 shows the coverage probabilities for the situation where $nSys = 5$, $m = 16$, and $Er = 4$. The left plot corresponds to the Wald interval result and the right plot is associated with the LRT interval result. In both the two plots, the circles represent the two-sided 90% interval, the triangles and crosses represent the lower and upper one-sided 95% intervals. In this case, the quantities of interest include $\beta, t_{0.1}, t_{0.3}, t_{0.632} \approx \eta,$ and $t_{0.9}$. By comparing the two plots, we conclude that LRT-based intervals produce coverage probabilities that are much closer to the desired nominal values than the Wald intervals. This is true for both two-sided intervals and one-sided bounds.

Figure 3.10 shows the coverage probabilities for the situation where $nSys = 5$, $m = 4$, and $Er = 4$. In this case, the quantities of interest are $\beta, t_{0.1}, t_{0.2},$ and $t_{0.632} \approx \eta$. The two plots show that neither of the two methods give coverage probabilities that are very close to the desired values. This is because the idML estimator has rather serious bias when the SRP has $m = 4$ slots and $Er = 4$ events. The performance for estimating the quantile $t_{0.632}$ seems to be the best among all the quantities of interest. This finding is consistent with what is shown in Figure 3.5, where the relative efficiency for estimating quantiles around $t_{0.7}$ is the highest.

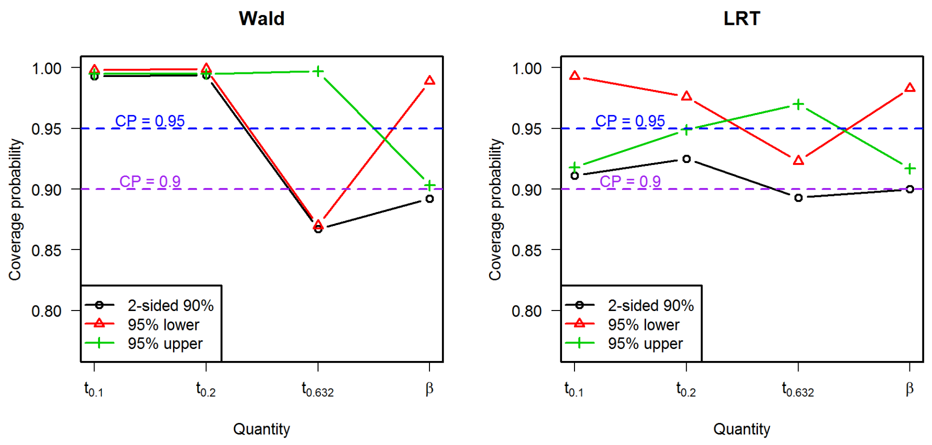

Figure 3.9 Coverage probability for Wald and LRT intervals ($nSys = 5, m = 16,\ Er = 4$)

3.7 Application to the Engine Cylinder Replacement Data

In this section, we applied the proposed idML estimation and LRT-based interval estimation procedures to analyze the diesel engine cylinder replacement data shown in Figure 4.1, corresponding to a fleet of 30 engines. Here is some description of the data

- The fleet has 30 engines (systems)

- Each engine has 16 cylinders (slots)

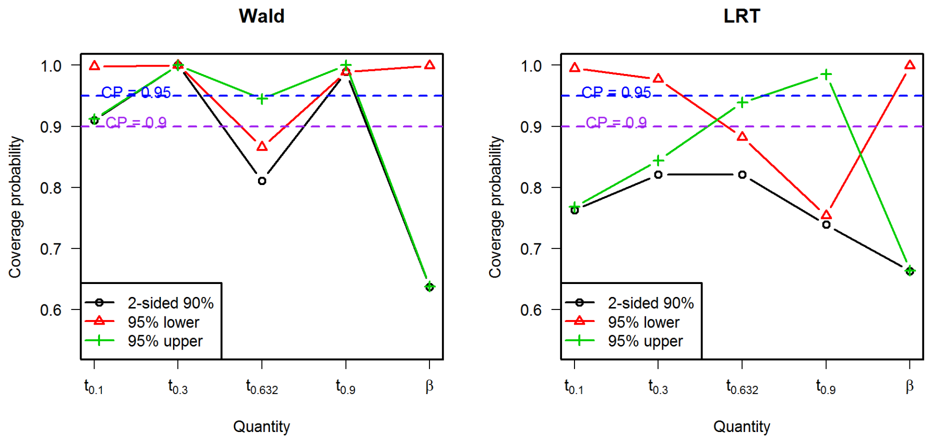

Figure 3.10 Coverage probability for Wald and LRT intervals ($nSys = 5, m = 4,\ Er = 4$)

- There is a total of 59 events. The fraction of slots with failure $\approx 59/[30 \times 16] \approx 0.123$. Therefore we expect to have good estimation up to and a little beyond $t_{0.1}$.

The point estimates and the 90% LRT confidence intervals for the shape Weibull β and some quantiles of interest, including $t_{0.001}$, $t_{0.01}, t_{0.05}, t_{0.1}$ and $t_{0.632} \approx \eta$ are shown in Table 3.4.

Table 3.4 Point estimates and LRT confidence intervals for parameters and quantiles of cylinder data

Quantities	Estimate	5%CI	95%CI
β	3.95	3.21	4.79
$t_{0.001}$	478.22	358.35	597.62
$t_{0.01}$	858.20	731.50	971.61
$t_{0.05}$	1297.20	1198.50	1385.28
$t_{0.1}$	1556.84	1471.51	1646.94
$t_{0.632} \approx \eta$	2753.93	2482.75	3154.23

Figure 3.11 is the Weibull probability plot of the idML estimates for a series of quantiles, together with the 90% confidence bands, obtained from the LRT-based procedure.

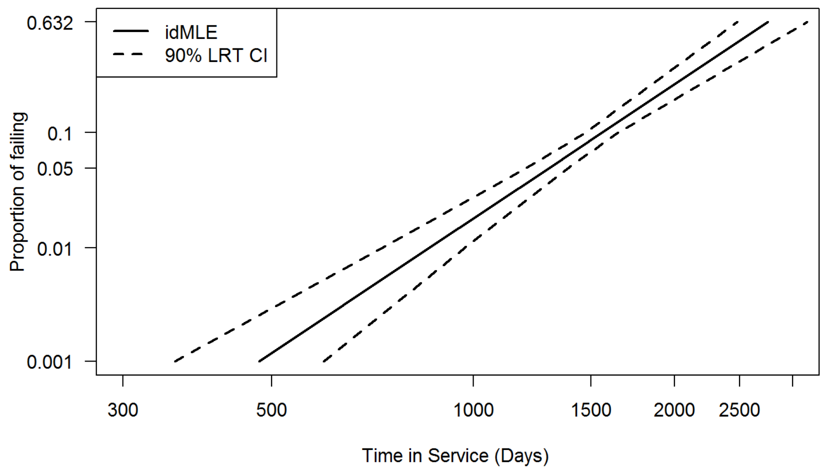

Figure 3.11 Weibull probability plot of the idML estimates and the 90% confidence bands for cylinder data

Figure 3.12 shows the empirical and fitted mean cumulative function (MCF) plot for the cylinder data. The solid curve represents the nonparametric estimate of the MCF, the dashed curve corresponds to the fitted MCF based on the idML estimates. The two dotted curves represent the 90% confidence bands. The fact that there is a high degree of agreement between the observed MCF and the fitted MCF, and that the whole fitted MCF curve falls inside the 90% confidence bands indicates that the idML estimate is a good candidate for describing the cylinder data.

3.8 Conclusions and Future Work

In this paper we proposed a method for estimating the lifetime distribution of a component from the aggregated event data for a fleet with multiple systems. Each system contains a set of identical replaceable components and the event history for each system is represented by an SRP. We derived the likelihood function for the observed

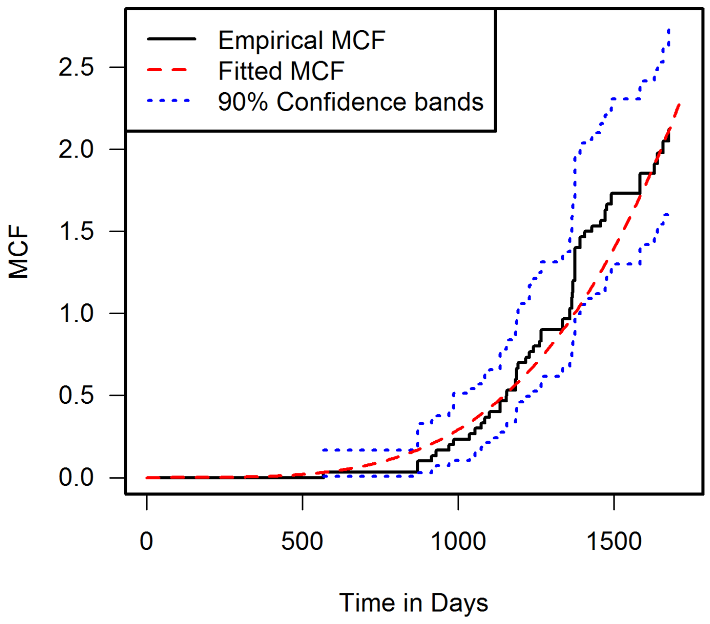

Figure 3.12 Empirical and fitted MCF plot for the subset of cylinder data

recurrent data for each SRP, by considering all unique data configurations, and then showed how to compute the likelihood for the observed events of the whole fleet as a weighted sum of the conditional likelihoods. By starting from a crude estimate of the weights and updating iteratively, we were able to obtain the idML estimates in an efficient way. The proposed estimation method is especially useful when the number of events for each SRP is relatively small, so that likelihood computation is feasible despite of the complex combinatorial nature, but the number of systems is sufficiently large, so that the total sample size is large enough to give adequate precision and for large sample theory to be adequate.

In addition, we evaluated the performance of interval estimation methods, based on Wald approximation and the LRT. We have shown that the LRT-based procedure can outperform the Wald-based procedure in producing better coverage properties.

Some possible areas for future are:

1. In the estimation procedure, we use simulations to estimate the data configuration probabilities conditional on the partition (i.e., p_{ij}), as part of the likelihood function. This is time-consuming and may lead to inaccuracy. It might be possible to derive the expression or an approximation for p_{ij} and compute these probabilities numerically, instead by simulations. This might produce more accurate result of the idML estimates.

2. It would be useful to conduct a larger simulation study to investigate the properties of the idML estimator over a wider range of parameters.

3. It is well known that the cdML estimator of the Weibull shape parameter β is biased. In some cases, the idML estimator has even more bias (e.g., when the number events is close to the number of slots in the SRP). It may be worthwhile to include a bias-correction step in the estimation.

4. An approximate likelihood computation could be developed that uses only high-probability partitions. A reasonable range of β values can be used to compute the partition probabilities. Then the partitions with high probabilities (e.g., larger than some specified threshold) can be identified and used to compute the likelihood. This could make the likelihood evaluation more efficient.

3.A Computation of the p_i

Here we describe how to compute the p_i's ($i = 0, 1, \cdots$) shown in Equation 3.6 in Section 3.3.5. Let $T_1, T_2, \ldots$ be the lifetimes for components in one renewal process (i.e., in one slot) where the lifetime distribution has CDF $F(t)$ and survival function $S(t)$. The recursive sequence to compute the probability of given number of events in a specific component is as follows:

$$\begin{aligned}
p_0 &= \Pr\left(\text{Zero events in the slot}\right)\\
&= \Pr\left(T_1 > t_c\right) = 1 - F\left(t_c\right) \equiv F^{(0)}(t_c) - F^{(1)}\left(t_c\right)\\
p_1 &= \Pr\left(\text{One event in the slot}\right)\\
&= \Pr\left(T_1 \le t_c, T_1 + T_2 > t_c\right)\\
&= \int_0^{t_c} S^{(0)}(t_c - u) f(u) du\\
&= F(t_c) - \int_0^{t_c} F(t_c - u) f(u) du\\
&= \int_0^{t_c} F^{(0)}(t_c - u) f(u) du - \int_0^{t_c} F^{(1)}(t_c - u) f(u) du.\\
&= F^{(1)}(t_c) - F^{(2)}\left(t_c\right)\\
p_2 &= \Pr\left(\text{Two events in the slot}\right)\\
&= \Pr\left(T_1 + T_2 \le t_c, T_1 + T_2 + T_3 > t_c\right)\\
&= \int_0^{t_c} S^{(1)}(t_c - u) f(u) du\\
&= F^{(2)}(t_c) - F^{(3)}\left(t_c\right)\\
&\vdots\\
p_{j+1} &= \int_0^{t_c} S^{(j)}(t_c - u) f(u) du\\
&= F^{(j+1)}(t_c) - F^{(j+2)}\left(t_c\right)
\end{aligned}$$

where $F^{(i)}(t)$ denotes the i-fold convolution of the CDF and $F^{(0)}(t) = 1, F^{(1)}(t) = F(t)$, etc. The proof of this results is given in Appendix B and the numerical algorithm for computing the convolution can be found in Halil (2005).

3.B Proof of the Probability of a Certain Number of Events in a Slot

In this appendix, we will show why p_j can be expressed as the difference of convolution of CDF's, as described in Appendix A.

$$\begin{aligned}
p_0 &= \Pr(T_1 > t_c) = 1 - F^{(1)}(t_c) \equiv F^{(0)}(t_c) - F^{(1)}(t_c) \\
p_1 &= \Pr(T_1 < t_c, T_1 + T_2 > t_c) \\
&= \Pr(T_1 < t_c, T_2 > t_c - T_1) \\
&= \int_0^{t_c} \int_{t_c - t_1}^{\infty} f(t_1) f(t_2) dt_2\, dt_1 \\
&= \int_0^{t_c} f(t_1) \int_{t_c - t_1}^{\infty} f(t_2) dt_2\, dt_1 \\
&= \int_0^{t_c} f(t_1) S(t_c - t_1) dt_1 \\
&= \int_0^{t_c} f(t_1) \left[1 - F(t_c - t_1)\right] dt_1 \\
&= \int_0^{t_c} f(t_1) dt_1 - \int_0^{t_c} f(t_1) F(t_c - t_1) dt_1 \\
&= F(t_c) - F^{(1)}(t_c) \equiv F^{(1)}(t_c) - F^{(2)}(t_c)
\end{aligned}$$

Now consider the case of p_2. To facilitate the development, define $V = T_1 + T_2$. Denote by $f_V(v)$ and $S_V(v)$ the density and the survival of V, respectively.

$$\begin{aligned}
p_2 &= \Pr(T_1 + T_2 < t_c, T_1 + T_2 + T_3 > t_c) \\
&= \Pr(V < t_c, V + T_3 > t_c) \\
&= \int_0^{t_c} \int_{t_c - v}^{\infty} f_V(v) f(t_3) dv \\
&= \int_0^{t_c} f_V(v) \int_{t_c - v}^{\infty} f(t_3) dt_3\, dv \\
&= \int_0^{t_c} f_V(v) \left[1 - F(t_c - v)\right] dv. \qquad (3.13)
\end{aligned}$$

It can be shown that

$$f_V(v) = \int_0^v f(w)f(v-w)dw. \tag{3.14}$$

Using (3.14) in (3.13) and exchanging the order of integration, one gets

$$\begin{aligned} p_2 &= \int_0^{t_c} \int_0^v f(w)f(v-w)S(t_c - v)dw\,dv \\ &= \int_0^{t_c} f(w) \int_w^{t_c} f(v-w)\left[1 - F(t_c - v)\right] dv\,dw. \end{aligned} \tag{3.15}$$

Changing variables $z = v - w$, in (3.15)

$$p_2 = \int_0^{t_c} f(w) \int_0^{t_c - w} f(z)\left[1 - F(t_c - w - z)\right] dz\,dw \tag{3.16}$$

$$= \int_0^{t_c} f(w)\left[F^{(1)}(t_c - w) - F^{(2)}(t_c - w)\right] dw \equiv F^{(2)}(t_c) - F^{(3)}(t_c). \tag{3.17}$$

The generalization of the proof to the case p_{j+1} is straightforward. It suffices to trace the proof for p_2.

References

Baxter, L., (1994), Estimation from quasi life tables, *Biometrika*, 81, 3, 567-577

Cox, D. R., (1962), *Renewal Theory*, London: Methuen & Co.

Feller, W., (1968), *An Introduction to Probability Theory and Its Applications*, vol.1, New York: John Wiley

Halil, A., (2005), A pointwise estimator for the k-fold convolution of a distribution function, *Communications in Statistics-Theory and Methods*, 34, 1939-1956

Khinchin, A.Ia.,(1956). On Poisson streams of random events. *Theory of Probability and Its Applications*, 1, 248-255

Meeker, W.Q., Escobar, L.A., (1998), *Statistical Methods for Reliability Data* New York: John Wiley & Sons.

Nelson, W.B., (2003), *Recurrent Events Data Analysis for Product Repairs, Disease Recurrences, and Other Applications*, ASA-SIAM, Philadelphia, ASA, Alexandria.

Tian, Y., Meeker, W.Q., (2013), Estimating a parametric lifetime distribution from superimposed renewal process data based on nonhomogeneous poisson process modeling, *to be submitted*

Tortorella, M., (1996), Life estimation from pooled discrete renewal counts, in Jewell, N.P. *et al*. (Eds), *Lifetime Data: Models in Reliability and Survival Analysis*, Kluwer Academic, Dordrecht, pp. 331-8.

Trindade, D.C. and Haugh, L.D., (1979), Nonparametric estimation of a lifetime distribution via the renewal function, *IBM Burlington Technical Report* TR 19.0463

Trindade, D.C. and Haugh, L.D., (1980), Estimation of the reliability of components from field renewal data, *Microelectronics Reliability*, 20, 205-218

CHAPTER 4. IMPLEMENTATION OF A MAXIMUM LIKELIHOOD PROCEDURE FOR MODELING THE SUPERPOSITION OF RENEWAL PROCESS DATA

A paper to be submitted

Ye Tian and William Q. Meeker
Department of Statistics
Iowa State University
Ames, IA 50014

Abstract

This paper documents an R implementation of a maximum likelihood algorithm for estimating the component lifetime distribution from a collection of superimposed renewal processes. The R function calls a C program for computationally intensive parts of the algorithm. The main R function receives a data frame with recurrence times as the input, together with some algorithm-control specifications. The function returns the Weibull parameter estimates and an estimate of the estimator's covariance matrix.

Keywords

C; Covariance matrix; Maximum likelihood estimation; R; Superposition of renewal processes (SRP)

4.1 Introduction

Tian, Escobar and Meeker (2013) described an estimation procedure for the data structures that can be described by a superposition of renewal process (SRP) or a fleet of systems, yielding multiple SRP's. In this article, we describe how the procedure is implemented using the computational tools in the implemented R and C programming languages. Major pieces of code are explained, including their input/output, as well how to call them appropriately.

4.1.1 Data Structure

The data structure for the multiple SRP's is illustrated using example data in Figure 4.1. This plot shows the cylinder replacement history for a fleet of 30 diesel engines, with each engine having 16 cylinders. In this example, each cylinder position can be considered as a slot and the event history for each slot can be described by a renewal process. The fleet has 30 independently running systems. The event plot tells us the engine where a cylinder replacement takes place (i.e. system-level information), but not the exact cylinder position (i.e. slot-level information). Therefore what is available for each system is a sequence of recurrence times with no slot identity.

4.1.2 Maximum Likelihood Formulation

Because the data are independent from system to system, the total log likelihood can be expressed as the sum of log likelihoods for each system (SRP). For each SRP, we identify all possible data configurations that could lead to the observed SRP. In other words, we enumerate all of the ways that these recurrences can be allocated to all the slots in the SRP. Then according to the law of total probability, the likelihood for a particular SRP is the weighted sum of all individual conditional likelihoods corresponding to all unique data configurations. The weights are probabilities (conditional on the observed

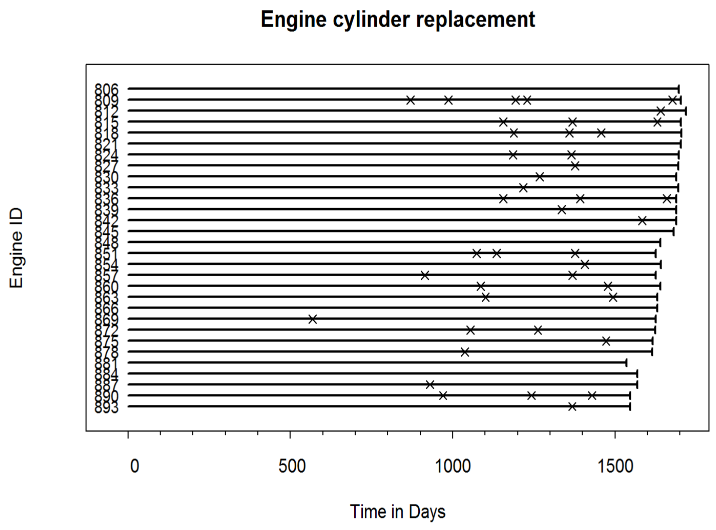

Figure 4.1 Event plot of diesel engine cylinder replacement

number of events) for these data configurations.

4.2 Likelihood for an SRP

4.2.1 Expression for the Likelihood

When the SRP has no events or one event, the estimation problem is trivial and the likelihood derivation is straightforward. In the following, we derive the likelihood where the SRP has at least two events.

Following Tian, Escobar, Meeker (2013), the likelihood for the observed SRP with $R = \{r \text{ observed events}\}$, event history $\mathcal{H}_{t_c} = (t_1, t_2, \ldots, t_r)$ and the end of observation time t_c is

$$L = \Pr(\mathcal{H}_{t_c}|R) = \sum_{i=1}^{h} k_i \pi_i \left[\sum_{j=1}^{s_i} p_{ij} L_{ij} \right] / \sum_{i=1}^{h} k_i \pi_i \tag{4.1}$$

Here π_i is the probability for a partition slot combination within partition i. A partition is a way to write r as a sum of positive integers. The partition indicates which slots in the SRP may have had events and how those events might have been allocated to the slots. A partition is denoted by $\mathcal{E}_i^r = (r_1, \cdots, r_{l_i})$. For example, for an SRP with $r = 3$ events and $nSlot = 16$ slots, there are three different partitions: (3), (2, 1), (1, 1, 1). For example, (2, 1) indicates that one slot in the SRP has two events and another slot has one event. All other slots have no events occurring. There are $k_i = 240$ equivalent slot combinations associated with this partition, corresponding to the ways those two slots with events are selected from all of the $nSlot = 16$ slots in the SRP. These 240 slot combinations are:

$$\begin{aligned}
\mathcal{A}_{2,1}^3 &= \{(\text{Two events in slot } 1, \text{one event in slot } 2)\} \\
\mathcal{A}_{2,2}^3 &= \{(\text{One event in slot } 1, \text{two events in slot } 2)\} \\
&\vdots \\
\mathcal{A}_{2,240}^3 &= \{(\text{One event in slot } 15, \text{Two events in slot } 16)\} \qquad (4.2)
\end{aligned}$$

Let k_i denote the number of ways that one can choose l_i slots from the m slots in the SRP, given a partition with length l_i. The value of k_i can be computed as

$$k_i = \binom{m}{l_i} \times \frac{l_i!}{n_1^*! \cdots n_q^*!},$$

where $(n_1^*, \cdots, n_q^*)$ are the frequencies of the q unique elements in $\mathcal{E}_i^r$.

The value p_{ij} is the probability of j^{th} (out of s_i) unique data configuration, conditional on partition i, with $\sum_{j=1}^{s_i} p_{ij} = 1$. Here a data configuration corresponds to the way the observed recurrence times can be allocated to the selected l_i slots. For example, for the partition $\mathcal{E}_2^3 = (2, 1)$ corresponding to an SRP with $r = 3$ events, there are $s_i = 3$ unique data configurations, shown in Table 4.1.

The data configuration $j = 1$ in Table 4.1 indicates the situation where the first two recurrence times take place in the same slot (which we call slot A) and the third recurrence time occurs in another slot (which we call slotB). Here the number of unique data configurations within a partition, s_i, can be computed as

Table 4.1 All data configurations for the partition $\mathcal{E}_2^3 = (2,1)$

j	slot A	slot B
1	t_1, t_2	t_3
2	t_1, t_3	t_2
3	t_2, t_3	t_1

$$s_i = \frac{r!}{[r_1!\, r_2! \ldots r_{l_i}!]\, [n_1^*! \cdots n_q^*!]}. \tag{4.3}$$

where the r_i values are the elements of the partition $\mathcal{E}_i^r$ and again the n_i^* values are the frequencies of the q unique elements of $\mathcal{E}_i^r$.

L_{ij} is the likelihood for the j^{th} data configuration within partition i. For example, data configuration 1 in Table 4.1 has a likelihood:

$$L_{ij} = [f(t_1)f(t_2 - t_1)S(t_c - t_2)]\, [f(t_3)S(t_c - t_3)]\, [S(t_c)]^{nSlot-2} \tag{4.4}$$

4.2.2 Reparameterization

Reparameterization can be beneficial in the optimization. For example, if we assume a Weibull distribution for the component failure time, then a common parameterization is (β, η), the usual shape and scale parameters. Here η is approximately 0.632 quantile of the Weibull distribution. For a problem with heavy censoring, let's say an expected fraction of failing of 0.1, we have good information about $t_{0.1}$, but not upper tail quantiles. Any attempt to estimate larger quantiles will involve extrapolation. In addition, as pointed out by Meeker and Escobar (1996), β and t_p with small p will be approximately independent in this case. Usually p is selected as one half of the observed fraction of failing. In the SRP problem, we can quantify the fraction of components that have failed, as

$$p_{\text{fail}} = \frac{nEvent}{nEvent + nSlot} \tag{4.5}$$

Then we reparameterize the problem into $[\log(\beta), \log(t_p)]$, where $p \approx p_{\text{fail}}/2$. The reason why the parameters are estimated on log scale is because all the quantities of interest are positive, so on log scale, the range of the quantity is unrestricted. Also, there is an expectation that the normal distribution approximation underlying the Wald method will be better on this unrestricted log scale.

4.2.3 Estimation Algorithm

Because closed form expressions for the p_{ij} values are not available, we use simulations to estimate these values. The simulation is computationally intensive. One approximate method is to start by assigning equal p_{ij} to all unique data configurations within a partition. This approximation works because SRP's with relatively small r but relatively large m have some dominant partition corresponding to the scenario where no slot has more than one event. In this case, there is only one unique data configuration within that dominant partition, so assigning equal p_{ij} to each data configuration gives an exact result to that partition. Also, assigning equal p_{ij}'s to other less dominant partitions will not change the likelihood too much because those partitions have a small probability and thus contribute little to the likelihood. This approximation simplifies the computation to begin the idML iterations. Based on this idea, we propose an iterative procedure to obtain the approximate idML estimates, as described below. For simplicity, we focus on one SRP. The procedure can be easily generalized to a fleet of multiple SRP's.

Algorithm 1: Approximate idML Estimation

1. Set $k = 0$

2. Give equal p_{ij} to every data configuration within each partition, and compute the first approximation to the ML estimates by maximizing (4.1). Denote the estimates $\hat{\boldsymbol{\theta}}^{\boldsymbol{k}} = \hat{\boldsymbol{\theta}}^{\boldsymbol{0}}$.

3. Simulate SRP's with *nSlot* slots and end-of-observation time t_c, according to the lifetime distribution with parameters $\hat{\boldsymbol{\theta}}^{\boldsymbol{k}}$. Then use the simulated SRP's to estimate p_{ij}'s.

4. Substitute the simulated p_{ij} values obtained in last step to the likelihood function and by maximizing (4.1), compute the ML estimate, denoted as $\hat{\boldsymbol{\theta}}^{\boldsymbol{k+1}}$.

5. Set $k = k + 1$. If $k < N_{\text{iter}}$, go to Step 3.

6. The approximate idML estimate is $\hat{\boldsymbol{\theta}}^{N_{\text{iter}}}$.

Here N_{Iter} is an input to the algorithm. In practice we found that a very small number (e.g., 1) is usually good enough, if the estimate of β is not too small.

4.3 Implementation and Code

4.3.1 Computation of π_i

As described in Tian, Escobar, Meeker (2013), the probability of any slot combination ($\mathcal{A}^r_{i,j}$) corresponding to partition i, π_i, can be calculated as

$$\pi_i = p_0^{m-l_i} \prod_{u=1}^{l_i} p_{r_u} \tag{4.6}$$

where p_{r_u} is the probability of r_u events in a slot,

$$p_j = F^{(j)}(t_c) - F^{(j+1)}(t_c) \tag{4.7}$$

and $F^{(j)}(t_c)$ is the j-fold convolution of the Weibull distribution function, evaluated at t_c. That is,

$$F^{(j)}(t_c) = \begin{cases} F(t_c) & j = 1 \qquad (4.8) \\ \int_0^\infty F^{(j-1)}(t_c - x)dF(x) & j > 1 \qquad (4.9) \end{cases}$$

This convolution can be obtained numerically, as shown in Halil (2006). A simple explanation of this algorithm is given below. Divide the interval $(0, t_c]$ into m equally-spaced intervals, with the division points t_i, where $i = 1, \cdots, m$. Then $F^{(n)}(t_i)$ can be recursively calculated as

$$F^{(n)}(t_i) = \sum_{j=1}^{i} \frac{F^{(n-1)}(t_i - t_{j-1}) + F^{(n-1)}(t_i - t_j)}{2} \left[F(t_j) - F(t_{j-1})\right] \tag{4.10}$$

According to this algorithm, $F^{(j)}(t_c)$ can be computed recursively as $F^{(j)}(t_m)$. The numerical convolution is implemented in C and called in R to improve the computational efficiency. The function `approxConvolution` calculates the desired convolution, that is, $F^{(n)}(t_c)$. The details are shown below:

```
void approxConvolution(int m, double Tc, int n, double bet, double eta)
  /*
    Args:
      m: Number of subintervals to divide (controls the accuracy)
      Tc: The end-of-observation time for the SRP
      n: To compute the n-fold convolution
      bet: Weibull shape parameter
      eta: Weibull scale parameter


    Returns:
      The n-fold convolution evalauted at Tc
  */
```

4.3.2 Estimation of p_{ij} by Simulation

We simulate a large number of SRP's to estimate the p_{ij} values. The R function `simulateDataConfig()` does the bookkeeping, stores all the counts in a two-level list and writes out the estimated p_{ij}'s for all partitions into .dat files. These files will be read in later steps and the saved p_{ij} values will be used in the objective function in the iterative procedure. Important inputs to this function include:

1. $nEvent$: The number of events we want to observe in the SRP. In the simulation, we only bookkeep simulated SRP's with exactly $nEvent$ events, corresponding to the number of events in the observed SRP.

2. $\boldsymbol{\theta}$: The parameters for the distribution used to simulate the SRP

3. t_c: The end-of-observation time for the SRP.

4. $nMin$: Stopping rule for the simulation. We stop the simulation whenever the second most dominant partition has more than $nMin$ counts. The larger $nMin$ is, the more counts are gained for all partitions, and as a result, the p_{ij} values should be more accurately estimated. But a larger $nMin$ will also take the simulation procedure longer time to complete.

We can enumerate all possible partitions of $nEvent$ by calling the `parts()` function in the R package `partitions`, according to Hankin (2013). Let the number of possible partitions be denoted by $nPart$. Then the returned list has $nPart$ components, with each component corresponding to a specific partition. Given partition $i = 1, \cdots, nPart$, we can obtain all of the unique data configurations, by calling the `setparts()` function in the R package `partitions`. Let the number of unique data configurations in partition i be s_i. Then the i^{th} component of the returned list is a list itself with s_i components, with each component recording the number of SRP's corresponding to each data configuration, as N_{ij}, for $j = 1, \cdots, s_i$. Then the Monte Carlo estimate of p_{ij} can be calculated as $N_{ij}/\sum_{j=1}^{s_i} N_{ij}$. Here $nMin$ controls when to stop the simulation. Because the most dominant partition will always have enough counts, what is more important is to get enough counts for less dominant partitions, in order to get accurate estimates of p_{ij}'s for these less dominant partitions. Currently we stop the simulation whenever the total count for the second dominant partition exceeds $nMin$. Larger $nMin$ will increase the accuracy, but will take more time for the simulation to complete.

Function `simulateDataConfig` also enables writing the estimated p_{ij}'s into external files. The file name is related to the partition because within one partition, $\sum_{j=1}^{s_i} p_{ij} = 1$. Here we use the example of $nEvent = 3$ and $nSlot = 3$ to illustrate the ideas. In this case, there are three partitions, with labels '3', '2.1', and '1.1.1', respectively. Here '3' refers to the partition where all the three events take place on the same slot, '2.1' indicates that two events occur on one slot and the other single event occurs on another slot, and '1.1.1' refers to the situation where the three events take place on three different slots. The corresponding partition name is part of the the file name. To accommodate the multi-system data structure and the iterative nature of the procedure, we also make the system label (`sysLab`) and iteration label (`iterLabel`) part of the file name. For example, if the system label is 1 and current iteration is 1, then the file name for the partition '3' will be system1_iter1_3.dat.

Because the p_{ij} values only depend on the number of events in the SRP (i.e., $nEvent$) and the end-of-observation time (e.g., t_c), if two SRP's have the same $nEvent$ and t_c, we only need to do simulation once and the estimated p_{ij}'s can be reused by other SRP's with same $nEvent$ and t_c.

An example of calling this function is shown below. Here $nEvent$ is 3 and $nMin$ is 20. The returned list, called `res`, is a list of three components, corresponding to the three partitions induced by nEvent. Within each component, we have a list (secondary level), corresponding to the data configurations within a partition (e.g. '1_2_1' in partition '2.1'). Here the data configuration '1_2_1' gives the slot index for the observed three events. That is, t_1 and t_3 occur in slot 1, t_2 takes place in slot 2. This partition has a total of $10 + 5 + 6 = 21$ counts, so the p_{ij}'s can be estimated as $(10/21, 5/21, 6/21) = (0.4761905, 0.2380952, 0.2857143)$, and are written to the .dat files.

```
> res <-
    simulateDataConfig(BETA = 3, ## Weibull shape parameter
                       ETA = 1, ## Weibull scale parameter
```

```
                                   sysLab = 1, ## System label
                                   iterLab = 1, ## Iteration label
                                   censorT = 1.2, ## End-of-observation time
                                   nSlot = 3, ## Number of slots in the SRP
                                   nEvent = 3, ## Number of events in the SRP
                                   nMin = 20 ## Miminum number of counts for the second
                                             ## dominant partition before stopping
                                             ## simulation
                                      )
```

```
> res
$‘3‘
$‘3‘$‘1_1_1‘
[1] 0



$‘2.1‘
$‘2.1‘$‘1_2_1‘
[1] 10


$‘2.1‘$‘1_1_2‘
[1] 5


$‘2.1‘$‘2_1_1‘
[1] 6
```

```
$`1.1.1`

$`1.1.1`$`1_2_3`

[1] 85
```

4.3.3 Main Function

The main R function, `idMLE()`, implements the incomplete data maximum likelihood estimation procedure.

4.3.3.1 Input and Output of the Main Function

The input of the main `idMLE()` function is described below:

- Data: a data frame with three columns, recording the system ID, event (recurrence/censoring) time and censoring status, respectively. Use 1 to denote replacement and 0 to denote the end of observation. The snapshot of the cylinder data looks like:

```
EngineID Time Censor
     806 1696      0
     809  870      1
     809  986      1
     809 1193      1
     809 1229      1
     809 1676      1
     809 1702      0
     812 1640      1
     812 1719      0
```

```
        815 1156      1
```

- p: Specifies which quantile (t_p) to estimate together with β.
- betInit: The initial value for β. The default value is 2. It can be changed according to user's knowledge about the failure mode.
- tpInit: The initial value for t_p. The algorithm computes a heuristic starting value for t_p automatically, but it also allows the users to specify this initial value according to their knowledge about the their data.
- $nSlot$: Number of slots for each system (SRP)
- m: A parameter controlling the accuracy of the numerical convolution in computing the slot combination probability, π_i. Larger values of m provide more accuracy, but require more computing time. Usually $m = 100$ is good enough.
- $nMin$: Stopping rule for the simulation for p_{ij}'s. We stop until the second most dominant partition has have more than $nMin$ counts.
- N_{iter}: Number of iterations in the iterative procedure.

The main function typically returns two components as the output.

- idML estimates
- the estimated variance-covariance matrix of the idML estimates (on the log scale), determined from the observed Hessian matrix evaluated at the last iteration of estimates.

4.3.3.2 Log-likelihood and Estimation

The function `logLikelihood.equalORgivenProb()` computes the log-likelihood function of the SRP, following Equation 4.1. This log-likelihood function is the objective function to be maximized in order to obtain the idML estimates and can be called in the following way:

```
logLikelihood.equalORgiveProb(para, nSlot, censorT, recurTime, m,
                                   equalProb = TRUE, sysLabel, iterLabel)
```

Here is a description of the arguments to the function `logLikelihood.equalORgivenProb()`.

1. `para`: a vector of parameters to estimate. For example, if the component failure time distribution is Weibull, then `para` will be the vector of $[\log(\beta), \log(t_p)]$
2. `nSlot`: the number of slots in the SRP
3. `censorT`: the end-of-observation time of the SRP
4. `recurTime`: the sequence of recurrence times for the SRP
5. `m`: the parameter that controls the accuracy of the numerical convolution approximation, as described in Section 4.3.1.
6. `equalProb`: if TRUE, then give equal p_{ij} to every unique data configuration within a partition, as described in Step 2 of *Algorithm 1.* If FALSE, then read in the p_{ij} values that are obtained by simulation from a previous step and saved in .dat files, as illustrated in Step 4 of *Algorithm 1.*
7. `sysLabel, iterLabel`: the system and iteration labels used in the .dat file name that we write p_{ij} values into.

4.4 Computational Efficiency of the Procedure

In this section we show how much computational time is needed for major steps of the proposed maximum likelihood estimation procedure. There are two steps that can be computationally expensive, including the likelihood maximization and the simulation (in order to estimate p_{ij} values).

Table 4.2 gives computation times for SRP's with number of events from $nEvent = 2$ to $nEvent = 8$ and $nSlot = 16$ (one SRP for each $nEvent$). For each $nEvent$ value, we record the time (sec) needed to evaluate a likelihood, to maximize the likelihood and to use the simulation to estimate p_{ij} values. As we can see when $nEvent$ gets larger, the time required to evaluate and maximize the likelihood increases, corresponding to the fact that the combinatorics becomes much more complicated. The overall trend for the time needed for simulation is increasing, but it is not monotonic. This is because for all the $nEvent$ values, we use the same simulation stopping criterion with $nMin = 20$. That is, we stop the simulation whenever the second most dominant partition has reached 20 counts. Note that when an SRP has two events, there are only two partitions, and one partition generally has a much larger probability than the other partition. Therefore we need to wait for a relatively long time in order to get enough counts for the second most dominant partition. When the SRP has four, five or six events, there are a lot more partitions and the probabilities for the most dominant and the second most dominant partitions are not significantly different, therefore it is much easier to get enough counts for the second dominant partition for cases with four to six events. Correspondingly the required time for simulation is shorter than the situation where $nEvent = 2$. Another finding is that the simulation is much more computationally intensive than the likelihood maximization.

Table 4.3 studies the effect of the parameter $nMin$ on the time and accuracy in estimating p_{ij}'s. In this study, we focus on one SRP with $nSlot = 3$ slots and $nEvent = 3$ events. The $nMin$ values from 20 to 2560 were used and for each $nMin$ value, we record the time needed to complete the simulation, and the computed p_{ij} values for the three data configurations for partition '2.1', denoted as '1_2_1', '1_1_2' and '2_1_1'. The p_{ij} values in Table 4.3 are rounded to three digits. The last row of the table, denoted by

Table 4.2 Computation time for likelihood evaluation, likelihood maximization and simulation for one SRP with different $nEvent$

$nEvent$	Likelihood	Maximization	Simulation for p_{ij}
2	0.007	0.351	89.6
3	0.016	1.257	30.6
4	0.033	2.487	16.5
5	0.064	4.741	12.9
6	0.148	10.222	16.4
7	0.416	32.891	96.9
8	2.062	138.093	6778.5

'True value', gives the p_{ij} values obtained from a large scale simulation, which is expected to give relatively accurate estimates of p_{ij}'s. The time needed for the simulation is close to being proportional to the $nMin$ value. This table gives some idea about the accuracy of the simulation for different stopping criteria (i.e., $nMin$).

Table 4.3 Effect of $nMin$ on the simulation result to estimate p_{ij} values

$nMin$	time (sec)	p1.2.1	p1.1.2	p2.1.1
20	1.06	0.619	0.190	0.190
40	1.78	0.610	0.293	0.098
80	3.23	0.617	0.235	0.148
160	6.36	0.602	0.242	0.155
320	11.97	0.583	0.259	0.159
640	24.20	0.583	0.259	0.158
1280	46.88	0.594	0.252	0.154
2560	96.91	0.601	0.246	0.152
True value		0.609	0.237	0.153

Figure 4.2 shows how the simulation time (top-left plot) and p_{ij} values (the other three plots) change as a function of $nMin$, for this realization of SRP. The red horizontal lines in the three p_{ij} plots represent the 'True value' shown in the last row of Table 4.3.

4.5 Some Results Based on Cylinder Data

This section illustrates the use of the main function `idMLE()` with the engine cylinder replacement data, as described in Figure 4.1. The returned results are shown below,

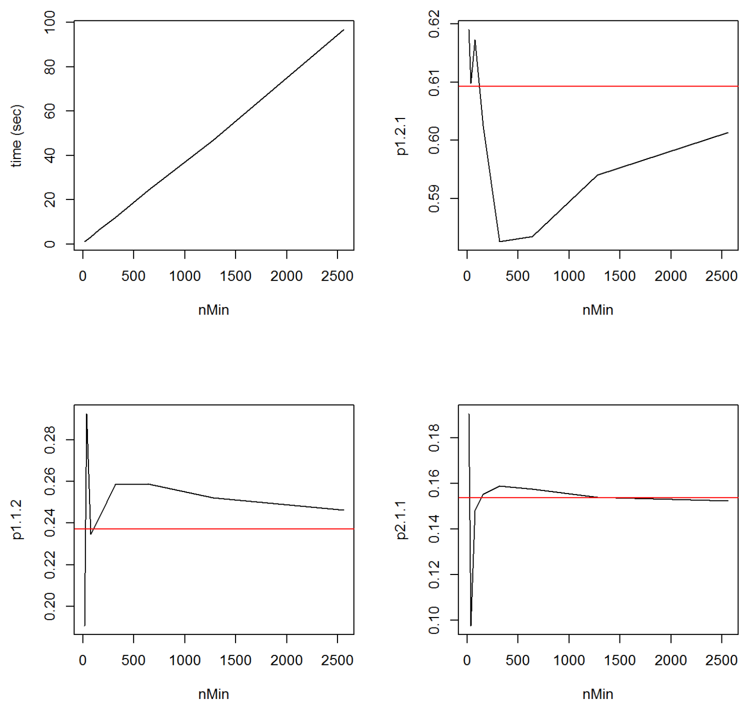

Figure 4.2 Effect of nMin on simulation efficiency and accuracy

including the idML estimates and the estimated covariance matrix of the estimators on the log scale.

```
> res.cylinder <-
    idMLE(dataFrame = dataCylinder, ## The data frame
          p = 0.1,                  ## Which quantile to estimate with beta
          betIni = 3,               ## Initial value for beta
          tpIni = 1600,             ## Intial value for tp
          m = 100,                  ## Controls accuracy for
                                    ## numerical convolution
          nMin = 20,                ## Simulation stopping rule
          nIter = 1                 ## Number of iterations needed
          )


> res.cylinder


$idMLE
       beta        t0.1
   3.945413 1556.835172


$covariance matrix
               log(beta)    log(t0.1)
  log(beta) 0.0148707362 0.0007170855
  log(t0.1) 0.0007170855 0.0011230124
```

4.6 Discussion

This paper provides an detailed description of how the idML estimation procedure is implemented. The procedure is implemented using a collection of R and C functions. This collection includes a main R function receiving the data and producing the idML estimates and estimated covariance matrix, and several building block R and C functions to perform different tasks. The functions allow users to specify different parameter settings in a flexible way. For example, the users can specify the number of iterations to proceed in *Algorithm 1* (i.e., N_{iter}), how accurate they would like the numerical convolution to be (i.e., m), and how efficient they expect the simulation be conducted to get the p_{ij} estimates (i.e., $nMin$). All of these parameters should be given according to user's knowledge about the problem and the time constraint.

References

Tian, Y., Escobar, L.A., Meeker, W.Q., (2013), Estimating a parametric component lifetime distribution from a collection of superimposed renewal processes, *in preparation*

Meeker, W.Q., Escobar, L.A., (1998), *Statistical Methods for Reliability Data* New York: John Wiley & Sons.

Hankin, R. K. S., (2013), Additive partitions of integers, *Manual for R package: partitions*

CHAPTER 5. GENERAL CONCLUSIONS

In this dissertation, we developed statistical models and methods for estimating component failure-time distribution from data structures that can be formulated as superposition of renewal processes (SRP) or a collection of independent SRP's.

In Chapter 2, we modeled the SRP data by a nonhomogeneous Poisson process (NHPP) with a particular MCF and recurrence rate function. We show that when the SRP has a large number of slots, the proposed NHPP estimator performs well and the interval estimation procedure based on transformed parametric bootstrap-t method has satisfactory coverage properties. By comparing the NHPP estimator with an alternative estimator, we make recommendations about which estimator to use for analyzing the SRP data. We hope the recommendations provide some useful tools for statisticians and engineers for better reliability analysis.

In Chapter 3, we derive the exact likelihood for the SRP data, by considering all possible data configurations that could lead to the observed event history. The likelihood can be computed as a weighted sum of conditional likelihoods corresponding to all unique data configurations, with weights being probabilities for each data configuration. We proposed an ML estimation procedure that starts from a crude estimate of the weights and updates iteratively. The bias and variance of the proposed ML estimator are analyzed. We show that when the number of events per slot in the SRP is relatively small, the ML estimator has relatively small bias, and the interval estimation procedure based on the likelihood ratio test has coverage probabilities close to the nominal values.

In Chapter 4, we developed an R function that implements the maximum likelihood estimation procedure described in Chapter 3. The estimation procedure is complicated and computationally intensive because of the need to enumerate all possible data configurations. The R function takes the recurrent event data as the input and returns the ML estimates and the estimated covariance matrix of the parameter estimates. Details about the implementation of the estimation procedure and how to use the function are discussed.