1-2-2019

Critical Fault-Detecting Time Evaluation in Software with Discrete Compound Poisson Models

Min-Hsiung Hsieh

Shuen-Lin Jeng

Paul Kvam
University of Richmond, pkvam@richmond.edu

This is a pre-publication author manuscript of the final, published article.

Recommended Citation

Hsieh, Min-Hsiung, Shuen-Lin Jeng, and Paul Kvam. “Critical Fault-Detecting Time Evaluation in Software with Discrete Compound Poisson Models.” *Journal of Quality Technology* 51, no. 1 (January 2, 2019): 94–108. https://doi.org/10.1080/00224065.2018.1545494.

Critical Fault-Detecting Time Evaluation in Software with Discrete Compound Poisson Models

Min-Hsiung Hsieh, Shuen-Lin Jeng [*]

Department of Statistics, National Cheng Kung University, Tainan,Taiwan

Paul Kvam

Department of Mathematics & Computer Science, University of Richmond

April 4, 2018

Final copy published in the *Journal of Quality Technology,* vol. 51, no. 1, pp. 94–108

Abstract

Software developers predict their product's failure rate using reliability growth models that are typically based on nonhomogeneous Poisson (NHP) processes. In this paper, we extend that practice to a nonhomogeneous discrete-compound Poisson process that allows for multiple faults of a system at the same time point. Along with traditional reliability metrics such as average number of failures in a time interval, we propose an alternative reliability index called Critical Fault-Detecting Time in order to provide more information for software managers making software quality evaluation and critical market policy decisions. We illustrate the significant potential for improved analysis using wireless failure data as well as simulated data.

Keywords: Goodness-of-Fit, Maximum Likelihood, l-fold Convolution, Nonhomogeneous Compound Poisson Process, Reliability Growth Model

[*]Corresponding Author: Dr. Jeng is an Associate Professor in the Department of Statistics. His email address is sljeng@mail.ncku.edu.tw

1 Introduction

In software quality evaluation, a variety of software reliability growth (SRG) models for software fault-failure (SF) processes have been developed to predict failure rate for new software by assuming a failure event is concurrent with a single fault. Nonhomogeneous Poisson (NHP) processes are effective for many applications, but in some modern software systems, each failure event may be induced by two or more faults (Debroy and Wong (2009), Hamill and Goševa-Popstojanova (2009)). In such cases, the SRG model of an NHP process becomes inadequate because it may not be able to properly describe the number of faults that leads to the occurrence of a failure event. This study proposes a more general modeling approach using a *nonhomogeneous discrete-compound Poisson* (NHDCP) model. The number of fault increments is described by a discrete random quantity (for example, Bernoulli or Poisson random variable) and the occurrences of failure events are described by an NHP process. When the number of fault increments is a constant quantity, the NHDCP model will simplify to an NHP process.

In the literature, there are several evaluation metrics that are commonly applied to software reliability. Four primary examples (Musa (1998), Pham (2000), Zachariah (2012), Zachariah (2015)) include: (1) The average number of failures experienced at any point in time; (2) The average number of failures in a time interval; (3) The failure intensity at any point in time; (4) The probability distribution of the time interval between failures.

These metrics provide fundamental levels for software quality evaluation. Furthermore, in the testing phase of software development, the software manager may also want to estimate the cost of testing time when a critical number of cumulative faults has been reached. This cost estimation is needed for the development of each new release version of the software. Our study proposes an alternative reliability index called the *critical fault-detecting time* (CFDT) for software quality evaluation. CFDT indicates the time when a certain critical fault-detected number (CFDN) is detected or found in a software test. The CFDN will be determined by the software manager to indicate the quality level threshold of the released software. The setting of the CFDN may also be referred to as the “quality level in the previous released version” or as a “standard of similar software”. This approach supplies a rich complement to the traditional metrics of software quality evaluation.

Because CFDT varies from system to system due to varying usage conditions, it is a random quantity (its distribution is called the critical fault-detecting time distribution). Using the test data, we seek to provide accurate estimates or predictions of CFDTs for software testing. CFDT distribution evaluation will help software managers more sensibly decide market policies. For example, insufficient testing of software in the manufacturer’s test environment may lead to exorbitant costs when the software is implemented in the field environment.

This research was motivated by the following wireless-network example. We do not fully adopt the assumptions of the example, instead aiming at slightly more pragmatic test schemes. Our specific model assumptions are explained at the end of the example.

Example of the Wireless Software Failure (WSF) Data

Figure 1 shows an example of the software faults for a wireless network system test given by Jeske et al. (2005). They performed a system test for the initial software release and there was interest in being able to predict the field failure rate for the software’s second release at various stages of testing. In that test, each failure is induced by a single fault. The software runs on an element within a wireless network switching center. Its main functions are routing voice channels and signaling messages to relevant radio resource & processing entities within the switching center.

The initial release test data are observed during a test that combines feature testing and load testing. Feature testing is designed to verify that the required features have been properly implemented and is often done with extremely focused test cases as opposed to test cases that would more naturally align with a user’s operational profile. Load testing consists of test scenarios designed to stress the system for the purpose of observing performance characteristics under peak or even overload conditions.

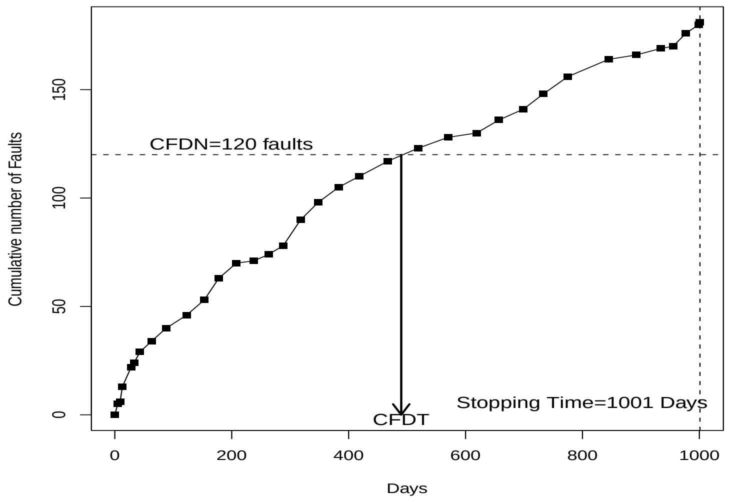

Figure 1: 181 Software faults are observed in the wireless network system and are aggregated by Jeske et al. (2005) The stopping time of the test is set at 1001 days and the arrow points at the CFDT under the CFDN of 120 faults.

Four systems were used in parallel to test the software and the data are obtained across multiple systems by aggregating the test time and the number of failures. That is, the observed fault numbers for individual software were not recorded. As shown in Figure 1, the cumulative total of days for the four systems is over 1,001 days and a total of 181 failures are observed under the planned inspection times.

Jeske et al. (2005) assumed that faults found in any test system were reported to all other test systems. As a result, a fault could be discovered and removed by two different system managers. Because the software's initial release had been used in the field for different systems under different environments, and one purpose of their test is to study the faults and failures in the field usage, our approach does not directly follow the assumptions of the Jeske et al. (2005) model. Instead, we consider the different test schemes that mimic the field-usage environment, where system-to-system variability may be critical and the individual systems are monitored separately.

In this paper, we focus on test schemes based on multiple tested systems with separate fault and failure information provided by individual systems (instead of aggregated data). Specifically, we extend the NHP model by Jeske et al. (2005) to a more flexible NHDCP model in order to capture the fault-failure structure in the observed data. The process of analysis based on the NHDCP model is demonstrated by a simulation study.

Brief Literature Review

Based on the NHP process, many SRG models have been developed in the literature. Yamada and Osaki (1983) survey several important SRG models for hardware and software systems including the Duane, Weibull, exponential, and gamma models. Kuo and Yang (1996) propose a unified approach to the NHP process in a SRG model and use a Bayesian approach for the reliability inference. In Zhang and Pham (2000), a SRG model incorporating imperfect debugging along with the learning process of software developers is evaluated and compared with other existing NHP process models for both descriptive and predictive power. Huang and Kuo (2002) investigate a SRG model with a logistic testing-effort function and show that the model has good predictive capability.

Huang et al. (2003) describe how several existing SRG models based on NHP processes can be derived by applying weighted means (geometric, harmonic or arithmetic). They show that these approaches cover a number of well-known models under different conditions. Teng and Pham (2006) establish a SRG model for predicting software reliability in random field environments. Lo and Huang (2006) propose a general framework for modeling the software fault detection and correction processes that cover a number of well-known SRG models. To build their SRG models, Zachariah and Rattihalli (2007) assume that the intensity of failure detection is proportional to the failure size. Wang et al. (2007) develop a moving average SRG model which includes the benefits of both the time-domain approach and the structure-based approach. That method overcomes the deficiency of existing NHP process techniques that fall short of addressing repair and internal system structures simultaneously.

Most SRG models assume that the fault removal time is negligible, and the fault can be removed immediately, but Huang and Huang (2008) argue that such an assumption is unrealistic for many industry problems. For this, they study how to incorporate finite and infinite server queueing models into software reliability modeling under both ideal and imperfect debugging conditions. Huang and Lin (2010) consider the testing compression factor and the quantified ratio of faults to failures and their effects on software reliability. Numerical examples based on recorded failure data show that the proposed framework has adequate prediction capability. Chatterjee and Singh (2014) investigate the optimal release policy with logistic exponential test coverage under imperfect debugging. Their software cost model incorporates testing coverage and an optimal release policy based on the number of remaining faults.

Besides these models, Sahinoglu (1992) discusses basic properties of compound distributions for Poisson-Geometric and Poisson-Logarithmic models. Goševa-Popstojanova and Trivedi (2000) model the software process with a Markov renewal process. Dohi and Yasui (2003) propose generalized binomial SRG models in both continuous and discrete time, based on cumulative Bernoulli trials. Pfefferman and Cernuschi-Frias (2002) propose a nonparametric and non-stationary procedure for software failure prediction. Wilson and Samaniego (2007) discuss a nonparametric analysis of the order statistic model in software reliability. Inoue and Yamada (2007) discuss a unified framework for discrete software reliability modeling with effect of program size in which the software failure occurrence times follow a discrete-time probability distribution. A more complete introduction of general theories and methods for this kind of reliability assessment can be found in the books of Meeker and Escobar (1998), Musa (1998), Pham (2000), Lawless (2003), and Yamada (2014).

Overview

Our study provides a general analysis for the evaluation of software reliability from measurements of software failure processes at planned inspection times. Based on a novel NHDCP model, we derive a critical fault detection time distribution and formulate the likelihood function for estimating the underlying model parameters. We rely on three well known goodness-of-fit (GOF) tests (Anderson-Darling, Cramér-von Mises, and Watson) based on the empirical distribution function (EDF) for assessing the NHDCP model adequacy. The WSF example is well suited to illustrate the procedure. Two random increment models, Bernoulli and Poisson, are used to describe the number of fault increments. We apply the G-O (Goel and Okumoto, (1979)) model to portray fault event occurrences and through the likelihood ratio test, we show the proposed NHDCP model is preferred over the one used in Jeske et al. (2005).

The rest of this article is organized as follows. In Section 2, we derive a generalized model for estimating CFDT and set up the likelihood framework for statistical inference, including confidence intervals and model fit assessment. In Section 3, we apply the proposed methods to the WSF data, comparing our results to those of Jeske et al. (2005). We provide model checking using the three mentioned GOF tests in Section 4, and conclusions are discussed in Section 5.

2 Modeling and Analysis

In this section, we construct a generalized procedure for assessing software reliability metrics. We first develop an NHDCP to model the fault-failure process, then we derive the CFDT distribution under a specified CFDN based on the method of maximum likelihood. Confidence intervals for CFDT distribution quantiles help demonstrate the uncertainty of the CFDT estimate and the tests for GOF can effectively assess model adequacy.

Nonhomogeneous Discrete-Compound Poisson Model

A stochastic process $\{Y(t) : t \geq 0\}$ is said to be an NHDCP process with cumulative failure event rate $m(t,\ \boldsymbol{\theta_2}) = \int_0^t \lambda(s;\ \boldsymbol{\theta_2})\mathrm{d}s$ for a non-decreasing non-negative intensity function $\lambda(t;\ \boldsymbol{\theta_2})$ (Ross, 2003) if

$$Y(t) = \begin{cases} X_1 + \cdots + X_{N(t)} & \text{if } N(t) \geq 1, \\ 0 & \text{if } N(t) = 0, \end{cases}$$

where $N(t)$ is a Poisson random variable with intensity $\lambda(t;\ \boldsymbol{\theta_2})$ which indicates the number of failure events up to time t and X_k represents the number of faults at the kth occurrence of failure event. $\{X_k, k \geq 1\}$ are discrete i.i.d. random variables with distribution $P_X(x; \boldsymbol{\theta_1})$ and independent of $N(t)$. When $\lambda(s;\ \boldsymbol{\theta_2})$ is a constant, this is a homogeneous Poisson model. Typical nonhomogeneous models, such as, Duane (1964), Cox and Lewis (1966), Musa and Okumuto (1984), and Goel and Okumoto (1979), are listed in Examples 3 and 4 below.

Zacks (2005) derived key properties of stopping times for a homogeneous discrete-compound Poisson (HDCP) process. We extend those results to a general NHDCP model as follows. Let $\{X_k\}$ be the positive increments with a probability mass function (PMF) $p_X(x; \boldsymbol{\theta_1})$. Then the PMF of $Y(t)$ on $\{0\ , 1, \cdots\ \infty\}$ is

$$p_{Y(t)}(y; \boldsymbol{\theta}) = \begin{cases} \exp\Big[-m(t; \boldsymbol{\theta_2})\Big], & y = 0, \\ \sum_{l=1}^{\infty} p_X^{(l)}(y; \boldsymbol{\theta_1}) \cdot p\Big(l;\ m(t; \boldsymbol{\theta_2})\Big), & y \in \{1, \cdots\ \infty\}, \end{cases} \tag{1}$$

where $p_X^{(l)}(y; \boldsymbol{\theta_1})$ is the l-fold convolution of $p_X(x; \boldsymbol{\theta_1})$, and

$$p\Big(l; m(t; \boldsymbol{\theta_2})\Big) = \exp[-m(t; \boldsymbol{\theta_2})] \frac{m(t;\ \boldsymbol{\theta_2})^l}{l!}$$

is the Poisson PMF, with parameters $\boldsymbol{\theta} = (\boldsymbol{\theta_1}, \boldsymbol{\theta_2})$, $\boldsymbol{\theta_1} = (\theta_{10},\ \theta_{11}, \cdots,\ \theta_{1p})$, and $\boldsymbol{\theta_2}$ $= (\theta_{20}, \theta_{21}, \cdots,\ \theta_{2q})$.

Zacks (2005) showed (1) for the special case when $m(t; \boldsymbol{\theta_2}) = \lambda t$. The derivation of (1) for the more general case of $m(t; \boldsymbol{\theta_2})$ has a similar proof. The corresponding cumulative distribution function (CDF) is

$$P_{Y(t)}(y; \boldsymbol{\theta}) = \exp\Big[-m(t; \boldsymbol{\theta_2})\Big] + \sum_{s=1}^{y} p_{Y(t)}(s; \boldsymbol{\theta}),$$

We offer two primary examples for obtaining the means and variances of $\Delta Y(t_{ij})$ and $Y(t_{ij})$ at time t_{ij}, where $\Delta Y(t_{ij}) = Y(t_{ij}) - Y(t_{ij-1})$, the index i represents the ith test system, and the index j represents the jth inspection time point. The numbers of fault increment are generated with Poisson and Bernoulli random variables. Calculations are straightforward, so details are omitted.

Example 1: Bernoulli increment model. Suppose that the fault number increment X_1 has the probability $\Pr(X_1 = 1) = 1 - \theta_{10}$ and $\Pr(X_1 = 2) = \theta_{10}$, and $\theta_{10} \in [0, 1]$. This model indicates that one or two faults induce a failure event. Then $\mathrm{E}[\Delta Y(t_{ij})|\ \Delta N(t_{ij}) = n_{ij}] = n_{ij}(1 + \theta_{10})$ and $\mathrm{Var}[\Delta Y(t_{ij})|\Delta N(t_{ij}) = n_{ij}] = n_{ij}(\theta_{10} - \theta_{10}^2)$. Hence, $\mathrm{E}[\Delta Y(t_{ij})] = \Delta m(t_{ij};\ \boldsymbol{\theta_2})(1 + \theta_{10})$ and $\mathrm{Var}[\Delta Y(t_{ij})]$=$\mathrm{Var}[\mathrm{E}[\Delta Y(t_{ij})|$ $\Delta N(t_{ij})]] + \mathrm{E}[\mathrm{Var}[\Delta Y(t_{ij})|\Delta N(t_{ij})]] = \Delta m(t_{ij};\ \boldsymbol{\theta_2})(1 + 3\theta_{10})$, where $\Delta N(t_{ij}) = N(t_{ij}) - N(t_{ij-1})$ and $\Delta m(t_{ij};\ \boldsymbol{\theta_2}) = m(t_{ij};\ \boldsymbol{\theta_2}) - m(t_{ij-1};\ \boldsymbol{\theta_2})$. The mean and variance for the accumulated number of faults are $\mathrm{E}[Y(t_{ij})] = m(t_{ij};\ \boldsymbol{\theta_2})(1+\theta_{10})$ and $\mathrm{Var}[Y(t_{ij})] = m(t_{ij};\ \boldsymbol{\theta_2})(1 + 3\theta_{10})$.

For this example, the l-fold convolution of $p_X(x; \boldsymbol{\theta_1})$, $p_X^{(l)}(y; \boldsymbol{\theta_1})$ is

$$p_X^{(l)}(y; \boldsymbol{\theta_1}) = \binom{l}{y-l} \theta_{10}^{y-l} \cdot (1 - \theta_{10})^{2l-y},\ l \le y \le 2l.$$

The details of the derivations are relegated to Appendix A. Following expression (1), the probabilities of $Y(t)$ on $\{0\ , 1, \cdots\ \infty\}$ become

$$p_{Y(t)}(y; \boldsymbol{\theta}) = \begin{cases} \exp\Big[- m(t; \boldsymbol{\theta_2})\Big], & y = 0, \\ \sum\limits_{l=y^*}^{y} \binom{l}{y-l} \theta_{10}^{y-l} \cdot (1 - \theta_{10})^{2l-y} \cdot p\Big(l; m(t;\ \boldsymbol{\theta_2})\Big), & y \in \{1, \cdots\ \infty\}, \end{cases}$$

where y^* is the smallest positive integer greater than or equal to $y/2$.

Example 2: Poisson increment model. Suppose that the the fault number increment $X_1 = X_1^* + 1$, where X_1^* has a Poisson distribution with parameter $\theta_{10} \in (0, \infty)$, so the number of faults ranges over $\{1, \infty\}$. This indicates that each failure event may be induced by one or more faults. As a result, $\mathrm{E}[\Delta Y(t_{ij})|\ \Delta N(t_{ij}) = n_{ij}] = n_{ij}(1 + \theta_{10})$ and $\mathrm{Var}[\Delta Y(t_{ij})|\Delta N(t_{ij}) = n_{ij}] = n_{ij}\theta_{10}$. From this, we have $\mathrm{E}[\Delta Y(t_{ij})]$

$= \Delta m(t_{ij};\ \boldsymbol{\theta_2})(1+\theta_{10})$ and $\mathrm{Var}[\Delta Y(t_{ij})] = \Delta m(t_{ij};\ \boldsymbol{\theta_2})(\theta_{10}^2+3\theta_{10}+1)$. Moreover, $\mathrm{E}[Y(t_{ij})] = m(t_{ij};\ \boldsymbol{\theta_2})(1+\theta_{10})$ and $\mathrm{Var}[Y(t_{ij})] = m(t_{ij};\ \boldsymbol{\theta_2})(\theta_{10}^2+3\theta_{10}+1)$.

The l-fold convolution $p_X^{(l)}(y;\boldsymbol{\theta_1})$ of $p_X(x;\boldsymbol{\theta_1})$ for the Poisson increment model is

$$p_X^{(l)}(y;\boldsymbol{\theta_1}) = \frac{\exp(-l\theta_{10})\cdot(l\theta_{10})^{y-l}}{(y-l)!},\ y = l, l+1, \cdots, \infty.$$

Appendix A displays details of the derivations. The probabilities of $Y(t)$ on $\{0\ , 1, \cdots\ \infty\}$ are

$$p_{Y(t)}(y;\boldsymbol{\theta}) = \begin{cases} \exp\Big[-m(t;\boldsymbol{\theta_2})\Big], & y = 0, \\ \sum\limits_{l=1}^{y}\Big[\frac{\exp(-l\theta_{10})\cdot(l\theta_{10})^{y-l}}{(y-l)!}\Big]\cdot p\Big(l; m(t;\boldsymbol{\theta_2})\Big), & y\in\{1,\cdots\ \infty\}. \end{cases}$$

Critical Fault-Detecting Time Distribution

The distribution time of when the CFDN will reach a critical level is an important index of the software quality. The information can also help managers determine the optimal release time of the software in order to achieve maximum profit. A critical number $\xi \in \mathcal{N}$ is used to denote the CFDN, where $\mathcal{N}$ is the set of positive integers. The critical fault-detecting time (denoted by T_ξ) is defined as the time when the actual path of the number of cumulative faults $Y(t)$ first reaches the CFDN of ξ, i.e.,

$$T_\xi = inf\{t:\ Y(t)\geq \xi\}.$$

Let F_{T_ξ} and f_{T_ξ} denote the CDF and probability density function (PDF) of T_ξ. Using $1 - P_{Y(t)}(\xi;\boldsymbol{\theta})$, F_{T_ξ} and f_{T_ξ} can be expressed as

$$\begin{aligned} F_{T_\xi}(t;\ \boldsymbol{\theta}) = \Pr(Y(t)\geq\xi;\ \boldsymbol{\theta},\xi>0) = 1 - \exp\Big[-m(t;\ \boldsymbol{\theta_2})\Big] - \\ \sum_{y=1}^{\xi-1}\sum_{l=1}^{\infty}\Big[p_X^{(l)}(y;\boldsymbol{\theta}_1)\Big]\cdot p(l\ ; m(t;\boldsymbol{\theta}_2)), \qquad (2) \\ f_{T_\xi}(t;\boldsymbol{\theta}) = m'(t;\boldsymbol{\theta_2})\exp\Big[-m(t;\boldsymbol{\theta_2})\Big] + m'(t;\boldsymbol{\theta_2})\cdot \\ \sum_{y=1}^{\xi-1}\sum_{l=1}^{\infty}\Big[p_X^{(l)}(y;\boldsymbol{\theta_1})\Big]\cdot d_p\Big(l; m(t;\boldsymbol{\theta_2})\Big), \end{aligned}$$

where $m'(t;\boldsymbol{\theta_2}) = \mathrm{d}m(t;\boldsymbol{\theta_2})/\mathrm{d}t$ is the intensity function at time t, i.e., $m'(t;\boldsymbol{\theta_2}) = \lambda(t;\boldsymbol{\theta_2})$. $p_X^{(l)}(y;\boldsymbol{\theta}_1)$ is defined in the previous subsection, and $d_p\Big(l; m(t;\boldsymbol{\theta_2})\Big)$ is the difference of Poisson probability at l and $l-1$ with rate $m(t;\boldsymbol{\theta_2})$, i.e., $p\Big(l; m(t;\boldsymbol{\theta_2})\Big) - p\Big(l-1; m(t;\boldsymbol{\theta_2})\Big)$.

The p^{th} quantile (t_p) of the CFDT distribution can be obtained by solving the equation $p = F_{T_\xi}(t_p;\boldsymbol{\theta}) = 1 - F_{Y(t_p)}(\xi;\boldsymbol{\theta})$ using (2). In applications, under a specific CFDN, t_p is a helpful guide for determining software release dates. For example, $t_{0.5}$ denotes the median time needed to detect the CFDN. Because debugging costs during field use are usually higher than costs incurred during the test phase (Inoue and Yamada 2007), time-schedule plans for future detecting processes in the field phase are crucial.

Let $M = \lim_{t\to\infty} m(t;\boldsymbol{\theta_2})$, which is not assumed to be finite. The derivations of $\mathrm{E}(T_\xi)$ and $\mathrm{Var}(T_\xi)$, computed using the proposition below, will depend on this limit value.

Proposition 1: If M is infinite, then

$$\mathrm{E}[m(T_\xi;\ \boldsymbol{\theta_2})] = 1 + \sum_{y=1}^{\xi-1}\sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1}),$$

and

$$\mathrm{E}[m^2(T_\xi;\ \boldsymbol{\theta_2})] = 2\Big[1 + \sum_{y=1}^{\xi-1}\sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1}) + \sum_{y=1}^{\xi-1}\sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1})\cdot l\Big].$$

If M is finite, then

$$\mathrm{E}[m(T_\xi;\ \boldsymbol{\theta_2})] = 1 - (1+M)\cdot \mathrm{e}^{-M} + \sum_{y=1}^{\xi-1}\sum_{l=1}^{\infty}\Bigg\{p_X^{(l)}(y;\ \boldsymbol{\theta_1})\cdot \Bigg[1 - \frac{\mathrm{e}^{-M}\cdot M^{l+1}}{l!} - \mathrm{e}^{-M}\cdot \mathrm{e}_l(M)\Bigg]\Bigg\}, \tag{3}$$

$$\mathrm{E}[m^2(T_\xi;\ \boldsymbol{\theta_2})] = 2 - \mathrm{e}^{-M}\cdot\Big[1 + (1+M)^2\Big] + \sum_{y=1}^{\xi-1}\sum_{l=1}^{\infty}\Bigg\{p_X^{(l)}(y;\ \boldsymbol{\theta_1})\cdot \Bigg[2(l+1) - \frac{\mathrm{e}^{-M}\cdot M^{l}}{l!}\cdot\Big[(1+M)^2 + l - 1\Big] - 2(l+1)\cdot \mathrm{e}^{-M}\cdot \mathrm{e}_l(M)\Bigg]\Bigg\}, \tag{4}$$

where $\mathrm{e}_l(M)$ is the exponential sum function, i.e., $\mathrm{e}_l(M) = \sum_{k=0}^{l} M^k/k!$. The derivations for the Proposition 1 are given in Appendix B.

Example 3: Suppose the occurrence of failure events follow a homogenous Poisson process with cumulative rate λt, i.e., $m(t;\ \boldsymbol{\theta_2}) = \lambda t \to \infty$ as $t \to \infty$. Then $\mathrm{E}(m(T_\xi;\ \boldsymbol{\theta_2}))$ $=\mathrm{E}(\lambda T_\xi) = 1 + \sum_{y=1}^{\xi-1}\sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1})$, and

$$\mathrm{E}(T_\xi) = \frac{\mathrm{E}[m(T_\xi;\ \boldsymbol{\theta_2})]}{\lambda}, \quad \mathrm{Var}(T_\xi) = \frac{\mathrm{Var}[m(T_\xi;\ \boldsymbol{\theta_2})]}{\lambda^2}.$$

The other cases of $M = \infty$ for the NHP process are the Duane (1964) process with $\lambda(t;\ \boldsymbol{\theta_2}) = \theta_{20}\theta_{21}t^{\theta_{21}-1}$, the Cox and Lewis (1966) process with $\lambda(t;\ \boldsymbol{\theta_2}) = \exp(\theta_{20} + \theta_{21}t)$, and the Musa and Okumuto (1984) process with $\lambda(t;\ \boldsymbol{\theta_2}) = \theta_{20}/(t + \theta_{21})$.

The complex form of $m(t;\ \boldsymbol{\theta_2})$ inhibits our ability to find the mean and variance of $\mathrm{E}(T_\xi)$ directly, so we rely on numerical approximation. If $m'(\mu_{T_\xi};\ \boldsymbol{\theta_2})$ and $m^{-1}(\cdot;\ \boldsymbol{\theta_2})$ exist, then a first-order Taylor expansion of $m(t;\ \boldsymbol{\theta_2})$ about $t = \mu_{T_\xi}$ is given by

$$m(t;\ \boldsymbol{\theta_2}) \approx m(\mu_{T_\xi};\ \boldsymbol{\theta_2}) + m'(\mu_{T_\xi};\ \boldsymbol{\theta_2})(t - \mu_{T_\xi}). \tag{5}$$

Replacing t by T_ξ and taking expectation on both sides, we have $\mathrm{E}[m(T_\xi;\ \boldsymbol{\theta_2})] \approx m(\mu_{T_\xi};\ \boldsymbol{\theta_2})$, so that $\mu_{T_\xi} \approx m^{-1}[\mathrm{E}[m(T_\xi;\ \boldsymbol{\theta_2})];\ \boldsymbol{\theta_2}]$. Next, modify the expression of (5) to a square form as

$$\Big[m(t;\ \boldsymbol{\theta_2}) - m(\mu_{T_\xi};\ \boldsymbol{\theta_2})\Big]^2 \approx m'(\mu_{T_\xi};\ \boldsymbol{\theta_2})^2(t - \mu_{T_\xi})^2,$$

and

$$\mathrm{E}\Bigg[\Big[m(T_\xi;\ \boldsymbol{\theta_2}) - m(\mu_{T_\xi};\ \boldsymbol{\theta_2})\Big]^2\Bigg] \approx m'(\mu_{T_\xi};\ \boldsymbol{\theta_2})^2\mathrm{E}\Big[(T_\xi - \mu_{T_\xi})^2\Big], \text{ i.e.,}$$

$$\mathrm{Var}\Big[m(T_\xi;\ \boldsymbol{\theta_2})\Big] \approx m'(\mu_{T_\xi};\ \boldsymbol{\theta_2})^2\mathrm{Var}(T_\xi).$$

Hence,

$$\sigma^2_{T_\xi} \approx \frac{\mathrm{Var}\Big[m(T_\xi;\ \boldsymbol{\theta_2})\Big]}{m'(\mu_{T_\xi};\ \boldsymbol{\theta_2})^2}.$$

Example 4: Suppose $m(t;\boldsymbol{\theta_2})$ follows the form of the G-O model as $\theta_{20}\Big(1-\exp[-\theta_{21}t]\Big)$. As $t \to \infty$, $m(t;\boldsymbol{\theta_2}) \to \theta_{20}$, i.e., $M = \theta_{20}$. Moreover, $m'(t;\boldsymbol{\theta_2}) = \theta_{20}\theta_{21}\exp[-\theta_{21}t]$ and

$m^{-1}(t;\boldsymbol{\theta_2}) = -1/\theta_{21}\log(1-t/\theta_{20})$. Then,

$$\mu_{T_\xi} \approx \frac{-1}{\theta_{21}} \log\left[1 - \frac{\mathrm{E}[m(T_\xi;\boldsymbol{\theta_2})]}{\theta_{20}}\right], \text{ and } \sigma^2_{T_\xi} \approx \frac{\mathrm{Var}\Big[m(T_\xi;\boldsymbol{\theta_2})\Big]}{\Big[\theta_{20}^2 \cdot \theta_{21}^2 \cdot \exp(-2\theta_{21}\cdot\mu_{T_\xi})\Big]}.$$

Replacing M by θ_{20} in the expressions of (3) and (4), we evaluate $\mathrm{Var}[m(T_\xi;\boldsymbol{\theta_2})]$. We note that the G-O model of $m(t;\boldsymbol{\theta_2})$ for an NHP process will be used in Section 3. When $m(t;\boldsymbol{\theta_2})$ is far from linear, we calculate μ_{T_ξ} and $\sigma^2_{T_\xi}$ via numerical integration, which can be computationally challenging.

For the other models of $m(t;\boldsymbol{\theta_2})$ with finite M, one can refer to the Pareto, Weibull and extreme value NHP processes (stated in Kuo and Yang, 1996) and the gamma NHP process (Yamada et al., 1983). The basic structure of these models is $m(t;\boldsymbol{\theta_2}) = \theta_{20}F(t;\boldsymbol{\theta_2})$ where $F(t;\boldsymbol{\theta_2})$ is the specified CDF (Musa et al., 1987). For the G-O model, $F(t;\boldsymbol{\theta_2}) = 1-\exp[-\theta_{21}\cdot t]$ so it is also an exponential NHP process.

Likelihood Function of Model Parameters

In this section, the likelihood function is constructed for estimating the underlying model parameters $\boldsymbol{\theta}$ with the data observed at planned inspection times. Following the likelihood function for the NHCP model from Hsieh et al. (2009), we extend their approach to the NHDCP model as follows. Let $\{Y_{ij}\}$ denote the observed data and let t_j represent the inspection times , $i = 1,\cdots,I$, and $j = 1,\cdots,L$. I and L represent the number of test systems and the number of measurements of a test system, respectively. t_L denotes the stopping time of the test. All software systems are recorded separately at the inspection times (t_j, $j = 1,\cdots,L$) and the data $Y_{i1},\cdots,Y_{iL}$, $i = 1,\cdots,I$, are observed. ($Y_{i0} = 0$ at $t_0 = 0$).

Suppose the number of fault increments of system i between the time intervals $j-1$ and j is positive ($\Delta Y_{ij} > 0$) at u intervals of inspected times and $\Delta Y_{ij} = 0$ are observed at the remaining v intervals (for convenience, we suppress further indexing of u and v within system i):

$$1 \le k_1 < k_2 < \cdots < k_u \le L,\ Y(t_{k_1}) - Y(t_{k_1-1}) > 0, \cdots, Y(t_{k_u}) - Y(t_{k_u-1}) > 0 \text{ and}$$
$$1 \le l_1 < l_2 < \cdots < l_v \le L,\ Y(t_{l_1}) - Y(t_{l_1-1}) = 0, \cdots, Y(t_{l_v}) - Y(t_{l_v-1}) = 0,$$

where $k_r \neq l_j,\ r = 1, \cdots, u,\ j = 1, \cdots, v$, and $u + v = L$.

Based on the observed time-point data, the likelihood for test system i is the product of the probability mass functions:

$$L(\boldsymbol{\theta},\ \mathbf{y}_i) = \left\{ \prod_{r=1}^{u} \left[\sum_{\Delta s_{ik_r}=1}^{\infty} p_X^{(\Delta s_{ik_r})}(\Delta y_{ik_r}; \boldsymbol{\theta}_1) \cdot \ \frac{\exp[-\Delta m(t_{k_r}; \boldsymbol{\theta}_2)] \cdot \Delta m(t_{k_r}; \boldsymbol{\theta}_2)^{\Delta s_{ik_r}}}{\Delta s_{ik_r}!} \right] \right\} \cdot$$
$$\left[\prod_{j=1}^{v} \exp\Big(- \Delta m(t_{l_j}; \boldsymbol{\theta}_2) \Big) \right], \tag{6}$$

where $p_X^{(\Delta s_{ik_j})}$ is the Δs_{ik_j}-fold convolution of p_X.

Note that if $u = 0$, (6) is equivalent to the likelihood with $\boldsymbol{Y}_i = \mathbf{0}$ in which no fault event occurs over the test period: $L(\boldsymbol{\theta},\ \mathbf{y}_i) = \exp[-m(t_{iL}; \boldsymbol{\theta}_2)]$. If $u = L$, then $\boldsymbol{\Delta Y}_i > \mathbf{0}$ where fault events occur during each inspection interval, i.e.,

$$L(\boldsymbol{\theta},\ \mathbf{y}_i) = \left\{ \prod_{r=1}^{L} \left[\sum_{\Delta s_{ik_r}=1}^{\infty} p_X^{(\Delta s_{ik_r})}(\Delta y_{ik_r}; \boldsymbol{\theta}_1) \cdot \frac{\exp[-\Delta m(t_{k_r}; \boldsymbol{\theta}_2)] \cdot \Delta m(t_{k_r}; \boldsymbol{\theta}_2)^{\Delta s_{ik_r}}}{\Delta s_{ik_r}!} \right] \right\}.$$

Then the overall joint likelihood function of $\boldsymbol{\theta}$ for the data with I independent test systems is

$$\mathcal{L}(\boldsymbol{\theta}|\mathbf{data}) = \prod_{i=1}^{I} L(\boldsymbol{\theta},\ \mathbf{y}_i). \tag{7}$$

Interval Estimation of CFDT

In order to quantify the uncertainty of the point estimate of quantile t_p, we construct an approximate confidence interval for t_p using the bias-corrected bootstrap method. We implement the parametric bootstrap method using the following steps outlined by Meeker and Escobar (1998).

1. Use the observed data from the I sample paths to compute the estimate $\hat{\boldsymbol{\theta}}$ by ML approach based on the likelihood given in (7) (By the optim function in the software R).

2. Numerically solve the estimates $\hat{t}_p$ by $p = F_{T_\xi}(\hat{t}_p;\ \hat{\boldsymbol{\theta}}) = 1 - F_{Y(\hat{t}_p)}(\xi;\ \hat{\boldsymbol{\theta}})$.

3. Generate a large number B of bootstrap samples that mimic the original sample and compute the corresponding bootstrap estimates for t_p according to the following steps:

   (a) Generate fault-count paths with size I from $\hat{\boldsymbol{\theta}}$.

   (b) Use the I simulated paths to estimate model parameters in order to produce the bootstrap estimates $\hat{\boldsymbol{\theta}}^*$.

   (c) Use the expression $p = F_{T_\xi}(t_p;\ \boldsymbol{\theta}) = 1 - F_{Y(t_p)}(\xi;\ \boldsymbol{\theta})$ to solve the bootstrap estimates $\hat{t}_p^*$ using the bootstrap estimates $\hat{\boldsymbol{\theta}}^*$.

4. The bootstrap confidence interval for t_p is computed using the following steps:

   (a) Sort the B bootstrap estimates $\hat{t}_{p,1}^*, \cdots, \hat{t}_{p,B}^*$ in increasing order giving $\hat{t}_{p,(b)}^*$, $b = 1, \cdots, B$.

   (b) Following Efron and Tibshirani (1993), lower and upper bounds of the pointwise approximate $100(1-\alpha)\%$ bias-corrected confidence intervals for t_p are

$$\left[\underline{t_p},\quad \overline{t_p}\right] = \left[\hat{t}_{p,(l)}^*,\quad \hat{t}_{p,(u)}^*\right],$$

   where

$$\begin{aligned} l &= B \cdot \Phi[2\Phi^{-1}(q) + \Phi^{-1}(\alpha/2)], \\ u &= B \cdot \Phi[2\Phi^{-1}(q) + \Phi^{-1}(1-\alpha/2)], \end{aligned}$$

   and q is the proportion of the B values of $\hat{t}_p^*$ that are less than $\hat{t}_p$ ($q = 0.5$ is the percentile bootstrap method), and Φ is the CDF of the standard normal distribution.

Goodness-of-Fit (GOF) Test

Checking model adequacy is a crucial step in software evaluation. Here we explore the application of three popular GOF tests under the assumed NHDCP model:

the Anderson-Darling statistic (A^2), the Cramér-von Mises statistic (W^2), and the Watson statistic (U^2). The Anderson-Darling test generally reveals greater power for continuous and discrete data among these quadratic empirical distribution function (EDF) statistics (Kvam and Vidakovic, 1998). These GOF procedures are based on assuming the software failure process follows an NHDCP model with the expected cumulative fault number $m(t;\boldsymbol{\theta_2})\mu_X$, which essentially represents a null hypothesis we are testing.

The corresponding estimates of the three statistics for the i^{th} system are

$$A_i^2 = \sum_{j=1}^{L} \frac{[Y_{ij} - m(t_j;\hat{\boldsymbol{\theta}}_{\boldsymbol{2}})\hat{\mu}_X]^2 \cdot [m(t_j;\hat{\boldsymbol{\theta}}_{\boldsymbol{2}}) - m(t_{j-1};\hat{\boldsymbol{\theta}}_{\boldsymbol{2}})]}{m(t_j;\hat{\boldsymbol{\theta}}_{\boldsymbol{2}})(\hat{R} - m(t_j;\hat{\boldsymbol{\theta}}_{\boldsymbol{2}})\hat{\mu}_X)}, \tag{8}$$

$$W_i^2 = \frac{1}{\hat{R}^2} \cdot \sum_{j=1}^{L} [Y_{ij} - m(t_j;\hat{\boldsymbol{\theta}}_{\boldsymbol{2}})\hat{\mu}_X]^2 \cdot [m(t_j;\hat{\boldsymbol{\theta}}_{\boldsymbol{2}}) - m(t_{j-1};\hat{\boldsymbol{\theta}}_{\boldsymbol{2}})]\hat{\mu}_X, \tag{9}$$

$$U_i^2 = \frac{1}{\hat{R}^2} \cdot \sum_{j=1}^{L} [Y_{ij} - m(t_j;\hat{\boldsymbol{\theta}}_{\boldsymbol{2}})\hat{\mu}_X - \sum_{j=1}^{L} [Y_{ij} - m(t_j;\hat{\boldsymbol{\theta}}_{\boldsymbol{2}})\hat{\mu}_X]/L]^2 \cdot [m(t_j;\hat{\boldsymbol{\theta}}_{\boldsymbol{2}}) - m(t_{j-1};\hat{\boldsymbol{\theta}}_{\boldsymbol{2}})]\hat{\mu}_X. \tag{10}$$

where R is the long-term expected number of cumulative faults for a software system (Jeske et al., 2008), i.e., $\hat{R} = \lim_{t\to\infty} m(t;\hat{\boldsymbol{\theta}}_{\boldsymbol{2}})) \cdot \hat{\mu}_X$. For example, if $m(t;\boldsymbol{\theta_2}))$ has the form in the G-O model, $R = \theta_{20} \cdot \mu_X$. The derivations of the formulas of the three statistics are given in Appendix C.

The statistics in (8), (9), and (10) represent a single sample path. Since the I test systems are independent, it is reasonable to aggregate those statistics, i.e., $A_I^2 = \sum_{i=1}^{I} A_i^2$, $W_I^2 = \sum_{i=1}^{I} W_i^2$, and $U_I^2 = \sum_{i=1}^{I} U_i^2$. Large values of the statistics A_I^2, W_I^2, and U_I^2 provide evidence against the hypothesized model. The critical values of A_I^2, W_I^2, and U_I^2 with the significance level may be obtained by Monte Carlo technique. Implementation details are given in Section 4.

3 Analysis of Wireless Software Failure

To illustrate our proposed methodology, we return to the WSF data illustrated in Figure 1. This section is divided into two subsections (for the probability model and for the statistical inference). Since we are not able to emulate the software test, we treat the aggregated data in Jeske et al. (2005) as if they were collected from a test of an individual system, and model these data directly. The simulation study for multiple systems in Section 4 is conducted based on the data generated from the best fitted model.

Modeling of the WSF Data

Jeske et al. (2005) considered a reliability growth model in an NHP process, where the number of fault increments was assumed with a constant quantity (Δ=1, referred to as a constant increment). To extend the procedure to applications that include multiple faults, we use the two discrete models featured in Examples 1 and 2.

The Bernoulli model (Example 1) assumes that either one or two faults occur per failure event: $X_1 = X_1^* + 1$, where X_1^* has a Bernoulli distribution with probability θ_{10}. Using this model, Zacks (2005) discussed the properties of the HDCP process. In this article, we extend the model to the nonhomogeneous (NHDCP) model.

In practice, a software system consists of many elements (program codes), and each occurrence of a failure event may be caused by a variety of faults. In this case, the Poisson model is appropriate (Example 2). Assume X_1 is the number of faults in a failure event where $X_1 = X_1^* + 1$ and X_1^* follows a Poisson distribution with rate θ_{10}. The mean and variance of the cumulative number of faults are given in Example 2. When $\theta_{10} = 0$, the Poisson fault increment model will reduce to the constant one.

For the cumulative fault event rate $m(t;\ \boldsymbol{\theta_2})$ of an NHP process, Jeske et al. (2005) used the G-O model, which is popular for assessing software reliability (Goel and Okumoto (1979), Kuo and Yang (1996), Pham (2000), and Huang et al. (2003)). In our analysis, it is also used to describe the occurrences of failure events. The expressions of the cumulative failure event rate $m(t;\ \boldsymbol{\theta_2})$ and the intensity function

$\lambda(t;\ \boldsymbol{\theta_2})$ for the G-O model are given respectively by

$$m(t;\ \boldsymbol{\theta_2}) = \theta_{20} \cdot \Big(1 - \exp[-\theta_{21} \cdot t]\Big) \text{ and } \lambda(t;\ \boldsymbol{\theta_2}) = \theta_{20} \cdot \theta_{21} \cdot \exp[-\theta_{21} \cdot t],\ \theta_{20},\ \theta_{21} > 0,$$

where θ_{20} is the expected total number of failure events to be eventually detected and θ_{21} represents the occurrence rate of the failure events. The expected total number of events is bounded by θ_{20} as $t \to \infty$ and the intensity function decreases in t and has a limit of zero.

Table 1: NHP process and NHDCP models.

Model	Fault increment model	Cumulative failure event rate of NHP Process
Model$_C$	Constant increments	G-O model
Model$_B$	Bernoulli model	G-O model
Model$_P$	Poisson model	G-O model

Table 1 lists the three models we are comparing. Model$_C$ represents the usual NHP process model and Model$_B$ and Model$_P$ belong to the class of NHDCP models. Note that the CFDT distribution and likelihood function for Model$_C$ can be obtained as a special case of Model$_B$. For the WSF data, Model$_B$ and Model$_P$ will be used to compare with the results of Model$_C$.

Critical Fault-Detecting Time Inference

The WSF data were aggregated from a multiple software systems. For our purposes of creating more broadly applicable results, we consider multiple (independent) systems without allowing for data aggregation. In this case, the system was inspected through the planned time points until 1001 days (see Figure 1). The MLEs of model parameters $\boldsymbol{\theta} = (\theta_{10}, \theta_{20}, \theta_{21})$ are obtained as (0.268, 189.271, $1.402 \cdot 10^{-3}$) for Model$_B$ and (0.300, 185.519, $1.387 \cdot 10^{-3}$) for Model$_P$. Model$_C$ is a special case of Model$_B$ when the probability of 2 faults of each failure event is set to 0, (i.e. one fault of each failure event). In Model$_B$, the MLE of the probability of 2 faults of each failure event

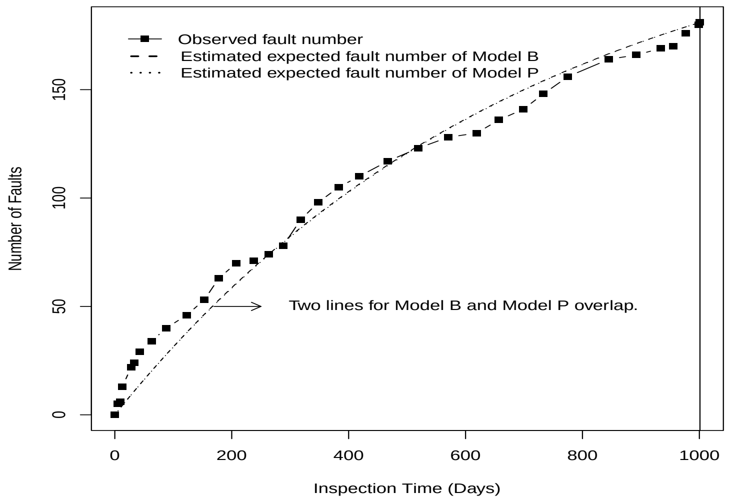

Figure 2: The plot of the estimated expected numbers of cumulative faults (Model$_B$ and Model$_P$, dotted lines) and the observed WSF data (squares).

is 0.268 which is not close to 0. This is an indication that Model$_C$ is not adequate for this data set.

Based on Examples 1 and 2, the estimated expected number of cumulative faults under the NHDCP models of Model$_B$ and Model$_P$ at time $t = 1001$ days is 181 which equals to the observed number. As $t \to \infty$, the expected numbers are estimated as 240 and 241 faults respectively for Model$_B$ and Model$_P$. Moreover, Figure 2 presents a visual comparison between the (estimated) expected and the observed cumulative numbers of faults. In this plot, the dotted lines denote the expected cumulative numbers of faults for Model$_B$ and Model$_P$ and the solid line represents the observed numbers. Figure 2 reveals that both NHDCP models fit the WSF data adequately.

Table 2 shows the log-likelihood values of Model$_C$, Model$_B$, and Model$_P$. The log-likelihood value of Model$_C$ is slightly smaller compared to those from Model$_B$ and Model$_P$. Moreover, Model$_B$ and Model$_P$ have similar log-likelihood values. The likelihood ratio test based on comparing the nested models between Model$_C$ and Model$_B$ produces a (chi-square) test statistic of $X^2 = 5.08$. If $Y(t)$ follows an NHP

Table 2: Log-likelihood values of Model$_C$, Model$_B$, and Model$_P$ and the likelihood ratio test between Model$_C$ and Model$_B$ (Model$_C$ is a sub-model of Model$_B$) for the WSF data.

model	log-likelihood	Comparison	− 2log-likelihood ratio	p-value
Model$_C$	-88.98	–	–	–
Model$_B$	-86.44	C vs B	5.08	0.02
Model$_P$	-86.09	–	–	–

process of Model$_C$, the test statistic has a chi-square distribution with one degree of freedom (χ^2_1), and $P(X^2 > 5.08) \approx 0.02$. This p-value provides strong evidence against the assumption that $Y(t)$ follows an NHP Process of Model$_C$; Model$_B$ provides a better description for the WSF data compared to Model$_C$. In other words, the more general NHDCP model represents a significant improvement on the WSF model provided by Jeske et al. (2005).

In a competitive market, it is essential for software producers to understand the most informative product characteristics related to its performance reliability. In this study, we focus on the evaluation of CFDT, which is defined as the time when the cumulative faults reach a critical threshold. Correspondingly, the fault detected efficiency (FDE) is defined as

$$\text{FDE} = \frac{\text{Expected detected number of faults}}{\text{Expected total number of faults}} \times 100\%.$$

For the wireless software systems, we consider the critical CFDNs as 120, 192, and 216 faults. These numbers come from the estimated FDE, respectively, at 50%, 80%, and 90%. For example, at the threshold of 216 faults, 90% (216/240) of the software faults have been detected and removed.

For Model$_B$ and Model$_P$, Figure 3 shows the estimates of the 0.10 and 0.50 quantiles of the CFDT distribution and their corresponding 95% bootstrap confidence intervals under the CFDNs of 120, 192, and 216 faults. The confidence intervals are obtained via 2,000 bootstrap simulations to quantify the variability of CFDT esti-

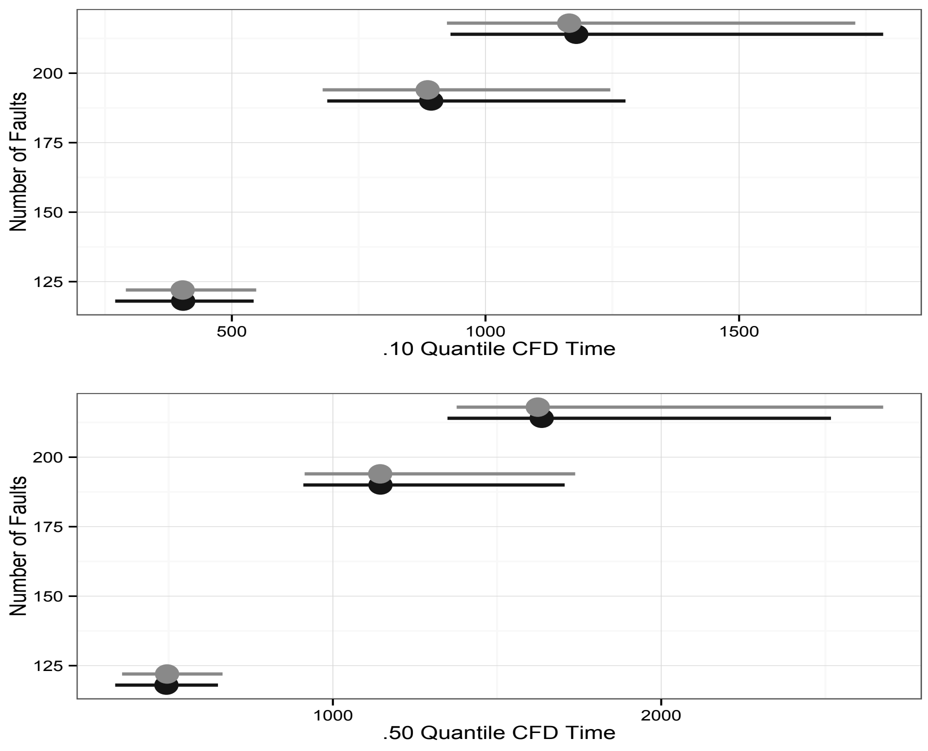

Figure 3: Estimates of p-th quantiles ($\hat{t}_p$) of the CFDT distribution and their corresponding 95% bootstrap confidence intervals with Model$_B$ (Black) and Model$_P$ (Gray) at the CFDNs of 120, 192, and 216 faults for the WSF data.

mates. Using the two NHDCP models (Model$_B$ in gray and Model$_P$ in black), we can see there is no significant difference between the models for the WSF data. At the CFDN of 120 faults, the estimated median of CFDT is about 500 days, i.e., 120 faults can be detected within 500 days in 50% of the tested software systems. To predict the CFDT in the field phase, 80% faults are detected at the time of about 1,145 days (i.e., after 144 days of the stopping time of 1001 days) in 50% of the tested software systems and 90% faults are detected at the time of about 1,630 days (i.e., after 629 days of the stopping time) in 50% of the tested software systems. The variability at the higher CFDN is relatively larger compared with that at the lower one. Because of small sample size, Figure 3 shows large variation between the CFDT estimates.

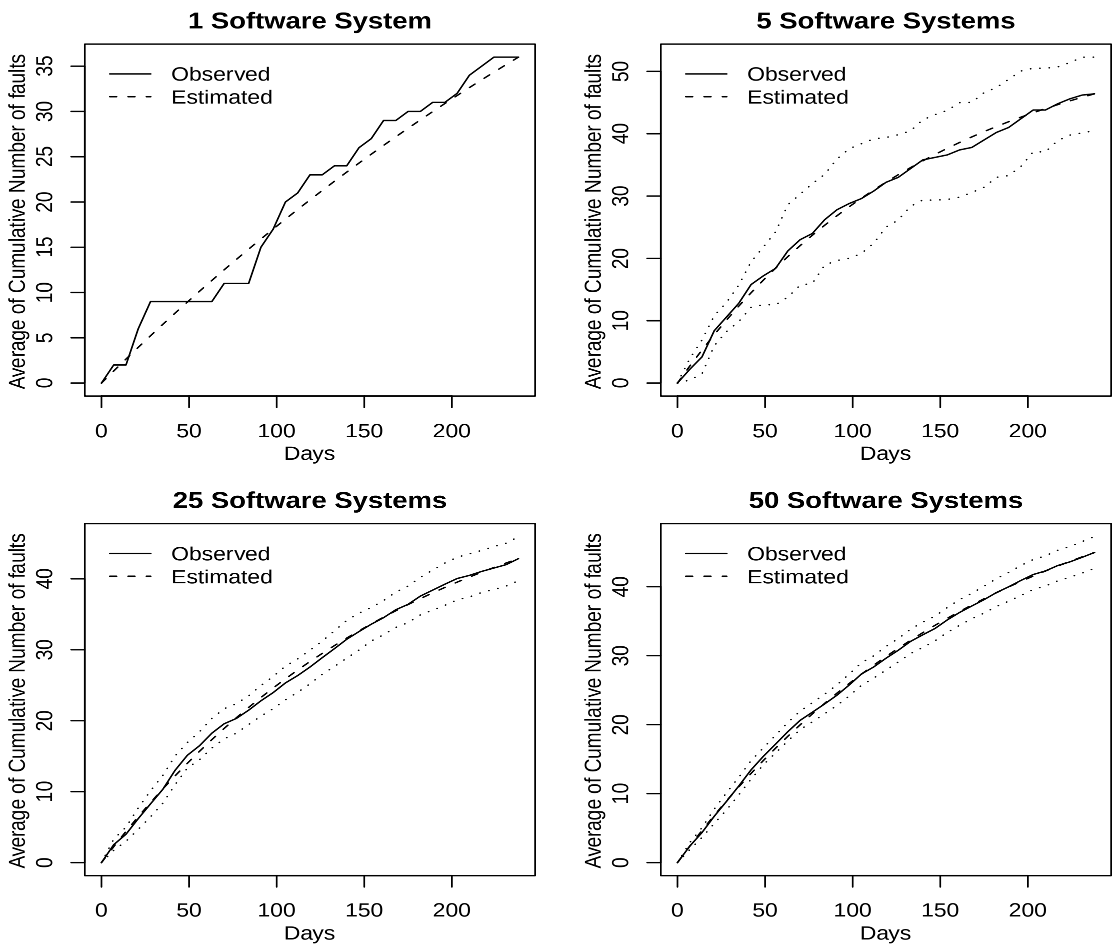

Figure 4: Plots of GOF test with sizes of 1, 5, 25, and 50 software systems under the WSFs of Section 3.

4 Model Checking

Model$_P$ is as effective as Model$_B$ for fitting the WSF data, and Model$_P$ is more flexible in fitting the data with a larger number of faults of each failure. We illustrate the procedure of model checking for Model$_B$. The procedure is the same for the other two models. The fitted plots and formal statistical tests are typical tools for model checking. We demonstrate the model checking procedure using the simulations based on the previous parameter estimates $\boldsymbol{\theta}$ =(0.300, 46.380, $5.548 \cdot 10^{-3}$) and the stopping time t_L, 238 days (34 weeks). The parameter values are the MLEs of Model$_P$ on the

WSF data. Four sets of data which are corresponding to the software tests of single and multiple (size 5, 25 and 50) systems are simulated.

Figure 4 provides the fitted plots for Model$_P$ via the plotted averages of the observed (simulated) cumulative numbers of faults and the estimated expected cumulative numbers of faults with single and multiple (size 5, 25 and 50) software systems. In Figure 4, the solid lines denote averages of the observed (simulated) cumulative numbers of faults (i.e., $\bar{Y}_j = \sum_{i=1}^{I} Y_{ij}/I$) which are calculated at weekly inspection times. The dashed line (centered) represents the estimated expected cumulative number of faults, $m(t_j; \hat{\boldsymbol{\theta}}_2) \cdot (1 + \hat{\theta}_{10})$, which is given in Example 2. The dotted lines in the cases of 5, 25 and 50 software systems show the variability of the sample average for the cumulative number of faults based on two standard errors.

Table 3: GOF test: p-values using the EDF statistics of A_I^2, W_I^2, and U_I^2 for checking the assumed NHDCP model.

	Number of Systems			
Statistic	1	5	25	50
A_I^2	0.254	0.471	0.685	0.805
W_I^2	0.230	0.427	0.734	0.837
U_I^2	0.371	0.482	0.865	0.921

Now we illustrate the application of the formal statistical tests. Here the null hypothesis states that the SF process $Y(t)$ follows an NHDCP model (Model$_P$), where the values of the parameters are set to the MLEs. The EDF-based statistics of A_I^2, W_I^2, and U_I^2 (from equations (8), (9) and (10), respectively) are calculated for the four simulated data sets corresponding to the four different number of systems (1, 5, 25, 50) and used as the the critical values for the GOF tests. The approximate distributions of these three statistics are obtained by another 10,000 simulations under the null hypothesis. Table 3 shows the corresponding p-values (calculated from the proportion of the simulated statistics which are larger than the critical values) for

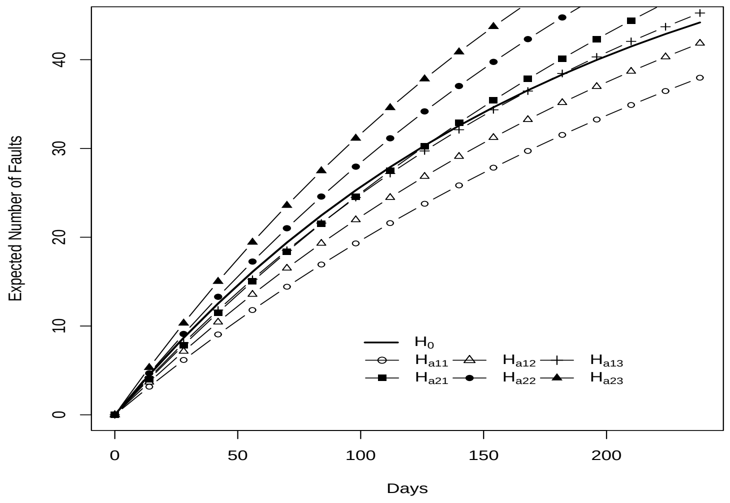

Figure 5: Plot of expected cumulative number of faults for the seven hypotheses.

the goodness of fit test using various system sizes. In all cases, there is no reasonable evidence against the null hypothesis; all p-values are greater than significance level α = 0.20.

Next, we examine statistical power for the three GOF tests. Data are simulated according to the corresponding seven hypotheses where H_0 assumes Model$_P$ with parameters $\boldsymbol{\theta^a_{11}} = \boldsymbol{\theta} = (0.300, 46.380, 5.548 \cdot 10^{-3})$, while six alternative hypotheses $(H_{a_{ij}}, i, j = 1, 2)$ are specified as:

$$
\begin{aligned}
&H_{a_{11}} : \text{ Model}_P \text{ with } \boldsymbol{\theta^a_{11}} = \boldsymbol{\theta} \times (1.2, 1.1, 0.6), \quad &&H_{a_{21}} : \text{ Model}_P \text{ with } \boldsymbol{\theta^a_{21}} = \boldsymbol{\theta} \times (1.2, 1.4, 0.6),\\
&H_{a_{12}} : \text{ Model}_P \text{ with } \boldsymbol{\theta^a_{12}} = \boldsymbol{\theta} \times (1.2, 1.1, 0.7), \quad &&H_{a_{22}} : \text{ Model}_P \text{ with } \boldsymbol{\theta^a_{22}} = \boldsymbol{\theta} \times (1.2, 1.4, 0.7),\\
&H_{a_{13}} : \text{ Model}_P \text{ with } \boldsymbol{\theta^a_{13}} = \boldsymbol{\theta} \times (1.2, 1.1, 0.8), \quad &&H_{a_{23}} : \text{ Model}_P \text{ with } \boldsymbol{\theta^a_{23}} = \boldsymbol{\theta} \times (1.2, 1.4, 0.8).
\end{aligned}
\tag{11}
$$

Figure 5 shows plots of the expected number of faults for these various hypotheses. Note that the number of faults of $H_{a_{13}}$ and $H_{a_{21}}$ are closer to H_0 compared to that of the numbers of $H_{a_{11}}$, $H_{a_{12}}$, $H_{a_{22}}$, and $H_{a_{23}}$.

Figure 6 shows the power graphs for the three different goodness of fit tests. The data are generated from the alternative hypotheses and sample size varies from one to 100 test systems. Through 2,000 simulations, the power is obtained by counting the proportion of the tests which reject the null hypothesis under the significant level 0.05. This procedure is repeated 5 times and the average of power is determined. The results show that test power is similar between A_I^2 and W_I^2 tests. However, the power of U_I^2 test is significantly different from A_I^2 and W_I^2 tests. When the expected cumulative number of faults under the the alternative hypothesis is lower than what is expected under H_0 (i.e., $H_{a_{11}}$ and $H_{a_{12}}$ in Figure 5), the power for U_I^2 test is smaller than A_I^2 and W_I^2 tests, otherwise, U_I^2 test has greater power ($H_{a_{13}}$, $H_{a_{21}}$, $H_{a_{22}}$ and $H_{a_{23}}$). There is no clear preference among these three tests. If a tolerance of testing power can be given, Figure 6 serves as a useful guideline for the selection of sample size.

5 Conclusion and Discussion

For software fault-failure processes, we have developed a more applicable nonhomogeneous discrete-compound Poisson model to resolve the issue of a failure event induced by two or more faults. This procedure improves measurably on the NHP process models in this domain of failure event problems. Critical fault-detecting times under a specific critical fault number proposed in this study have great potential to aid in policy decisions with test data or field data. This is currently an important challenge in software reliability that has yet to be adequately addressed in the literature.

The study was motivated, in part, by the wireless software data given by Jeske, et al. (2005), and we featured this data in order to illustrate our new methods, including the estimation, confidence statements and goodness of fit procedure. Through the likelihood ratio test, our proposed NHDCP models based on the Goel-Okumoto model with Binomial and Poisson fault increments can provide better descriptions for the fault-failure characteristics compared to the common use model based on NHP processes by Jeske, et al. (2005). We provide further insight into the effectiveness of the models by looking at scenarios with a varying number of systems under test.

There are some limitations to our model that we may characterize in three different issues. First, it is likely that not all the faults at a failure event are detected if it is dependent on the troubleshooting time and the resource level. In this case, we are inhibited in estimating the total number of cumulative faults. This problem can be avoided if the software managers focus on a prespecified number of detected faults as suggested by this paper. For the second issue, the rate of occurrence of failure events and the compounding random variable may be statistically dependent. As the rate decreases over time, the number of faults is likely to decrease too. However, there are cases in which this concern about dependence may be alleviated with our consideration of approximate models. For example, the cumulative number of faults is strictly increasing but still not very close to the bound when the test is stopped (this is the case in our study). Finally, the testing procedure might also have an effect because there will be more faults detected if the time gaps between sample points are too large. The time gaps between observations are modeled by the NHP process, and the sampling scheme or the inspection scheme is not treated separately in the NHDCP model. Along with the first two issues, the theoretical development for different sampling schemes may be of interest for future research, but will inevitably contain numerous modeling challenges.

Appendix A: l - Fold Convolution

Example 1: For the l-fold convolution $p_X^{(l)}(y;\theta_{10})$ of $p_X(x;\theta_{10})$ of the Bernoulli increment model, let $X_i = X^* + 1$, where X^* has a binomial distribution with parameters (β, θ_{10}). Using the additive property of the binomial,

$$\begin{aligned} p_X^{(l)}(y;\theta_{10}) &= \Pr(X_1 + X_2 + \cdots + X_l = y) = \Pr(X_1 - 1 + X_2 - 1 + \cdots + X_l - 1 = y - l) \\ &= \binom{\beta l}{y-l}\theta_{10}^{y-l}(1-\theta_{10})^{\beta l-(y-l)},\ y = l, l+1, \cdots, (\beta+1)l. \end{aligned}$$

If $\beta = 1$, this reduces to the Bernoulli model.

Example 2: For the l-fold convolution $p_X^{(l)}(y;\theta_{10})$ of $p_X(x;\theta_{10})$ of the Poisson increment model, let $X_i = X^* + 1$, where X^* has a Poisson distribution with rate θ_{10}. Then, using the additive property of the Poisson distribution,

$$\begin{aligned} p_X^{(l)}(y;\theta_{10}) &= \Pr(X_1 + X_2 + \cdots + X_l = y) = \Pr(X_1 - 1 + X_2 - 1 + \cdots + X_l - 1 = y - l) \\ &= \frac{\exp(-l\theta_{10}) \cdot (l\theta_{10})^{y-l}}{(y-l)!},\ y = l, l+1, \cdots \end{aligned}$$

Appendix B: Proof of the Proposition 1

Let $f^*(z;\boldsymbol{\theta})$ define the transform of $f_{T_\xi}(t;\ \boldsymbol{\theta})$ as

$$f^*(z;\boldsymbol{\theta}) = \int_0^\infty \exp[-z \cdot m(t;\ \boldsymbol{\theta_2})] \cdot f_{T_\xi}(t;\ \boldsymbol{\theta})\ \mathrm{d}t.$$

With $m(0;\ \boldsymbol{\theta_2}) = 0$ and $m(t;\ \boldsymbol{\theta_2})$ approaching the value M (finite or infinite) as $t \to \infty$, we have

$$\begin{aligned} f^*(z;\boldsymbol{\theta}) = \int_0^M \mathrm{e}^{-m(t;\ \boldsymbol{\theta_2})(1+z)}\ \mathrm{d}m(t;\ \boldsymbol{\theta_2}) + \sum_{y=1}^{\xi-1}\sum_{l=1}^{\infty} \left[p_X^{(l)}(y;\ \boldsymbol{\theta_1})\right] \cdot \\ \int_0^M \left[\frac{\mathrm{e}^{-m(t;\ \boldsymbol{\theta_2})(1+z)} \cdot m(t;\ \boldsymbol{\theta_2})^l}{l!} - \frac{\mathrm{e}^{-m(t;\ \boldsymbol{\theta_2})(1+z)} \cdot m(t;\ \boldsymbol{\theta_2})^{l-1}}{(l-1)!}\right] \mathrm{d}m(t;\ \boldsymbol{\theta_2}). \end{aligned} \tag{12}$$

If M is an infinite value, the expression (12) implies that

$$\begin{aligned} f^*(z;\boldsymbol{\theta}) &= \frac{1}{1+z} + \sum_{y=1}^{\xi-1}\sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1}) \cdot \left[\frac{\Gamma(l+1)/(1+z)^{l+1}}{l!} - \frac{\Gamma(l)/(1+z)^l}{(l-1)!}\right] \\ &= \frac{1}{1+z} - \frac{z}{1+z}\sum_{y=1}^{\xi-1}\sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1}) \cdot \Big(\frac{1}{1+z}\Big)^l, \end{aligned}$$

where $\Gamma(l,\alpha)$ is the (upper) incomplete gamma function, i.e., $\Gamma(l,\alpha) = \int_\alpha^\infty t^{l-1}\mathrm{e}^{-t}\ \mathrm{d}t$.

If M is a finite value, the expression (12) becomes

$$\begin{aligned} f^*(z;\boldsymbol{\theta}) = \frac{1}{1+z}\Big[1 - \mathrm{e}^{-M(1+z)}\Big] - \frac{z}{1+z}\sum_{y=1}^{\xi-1}\sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1}) \cdot \Big(\frac{1}{1+z}\Big)^l \cdot \\ \left[1 - \mathrm{e}^{-M(1+z)} \cdot \mathrm{e}_{l-1}(M(1+z)) + \mathrm{e}^{-M(1+z)} \cdot \frac{[M(1+z)]^l}{l!z}\right], \end{aligned}$$

where $\mathrm{e}_l(\alpha)$ is the exponential sum function, i.e., $\mathrm{e}_l(\alpha) = \sum_{k=0}^{l} \alpha^k/k!$. Under standard regularity conditions, equations $-\frac{\partial}{\partial z} f^*(z; \boldsymbol{\theta})|_{z=0}$ and $\frac{\partial^2}{\partial z^2} f^*(z; \boldsymbol{\theta})|_{z=0}$ lead to solutions for $\mathrm{E}[m(T_\xi;\ \boldsymbol{\theta_2})]$ and $\mathrm{E}[m^2(T_\xi;\ \boldsymbol{\theta_2})]$, respectively. In the case of infinite M,

$$\mathrm{E}[m(T_\xi;\ \boldsymbol{\theta_2})] = 1 + \sum_{y=1}^{\xi-1} \sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1}),$$

and

$$\mathrm{E}[m^2(T_\xi;\ \boldsymbol{\theta_2})] = 2\Big[1 + \sum_{y=1}^{\xi-1} \sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1}) + \sum_{y=1}^{\xi-1} \sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1}) \cdot l\Big],$$

so that the variance of $m(T_\xi;\ \boldsymbol{\theta_2})$ is

$$\mathrm{Var}[m(T_\xi;\ \boldsymbol{\theta_2})] = 1 + 2\sum_{y=1}^{\xi-1} \sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1}) \cdot l - \Big(\sum_{y=1}^{\xi-1} \sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1})\Big)^2.$$

For the case of finite M,

$$\mathrm{E}[m(T_\xi;\ \boldsymbol{\theta_2})] = 1 - \mathrm{e}^{-M}(1+M) + \sum_{y=1}^{\xi-1} \sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1}) \cdot \left[1 - \frac{\mathrm{e}^{-M} M^{l+1}}{l!} - \mathrm{e}^{-M}\mathrm{e}_l(M)\right].$$

and

$$\begin{aligned}\mathrm{E}[m^2(T_\xi;\ \boldsymbol{\theta_2})] =& 2 - \mathrm{e}^{-M}\Big[1 + (1+M)^2\Big] + \sum_{y=1}^{\xi-1} \sum_{l=1}^{\infty} p_X^{(l)}(y;\ \boldsymbol{\theta_1}) \cdot \\ & \left[2(l+1) - \frac{\mathrm{e}^{-M} M^l}{l!} \cdot \Big[(1+M)^2 + l - 1\Big] - 2(l+1)\mathrm{e}^{-M}\mathrm{e}_l(M)\right],\end{aligned}$$

and the variance of $m(T_\xi; \boldsymbol{\theta_2})$ is derived from $\mathrm{Var}(X) = \mathrm{E}(X^2) - [\mathrm{E}(X)]^2$.

Appendix C: Derivations of the GOF test statistics

Choulakian et al. (1994) gave the formulas of Anderson-Darling statistic (A^2), Cramer-von Mises statistic (W^2), and Waston statistic (U^2) for testing discrete distributions. The given forms are quite general, we extent them to the NHDCP model.

For a discrete distribution with L cells, each with cell (frequency) probability p_j, $j \in \{1, 2, \cdots, L\}$, suppose we have R independent observations. Let o_j be the observed number of outcomes in cell j, and let $R \cdot p_j = e_j$ be the expected number in cell j. Define $S_j = \sum_{k=1}^{j} o_k$, $V_j = \sum_{k=1}^{j} e_k$, $Z_j = S_j - V_j$, $\bar{Z} = \sum_{j=1}^{L} Z_j/L$, and $H_j = V_j/R$. Then S_j/R and V_j/R correspond to the EDF $F_R(x)$ and the CDF $F(x)$ in the continuous case. For discrete distributions, the test statistics given by Choulakian et al. (1994) are

$$A^2 = \frac{1}{R} \cdot \sum_{j=1}^{L} \frac{Z_j^2 p_j}{H_j(1-H_j)},\ W^2 = \frac{1}{R} \cdot \sum_{j=1}^{L} Z_j^2 p_j,\ U^2 = \frac{1}{R} \cdot \sum_{j=1}^{L} (Z_j - \bar{Z})^2 p_j. \qquad (13)$$

In the setting of the fault-detection test, notice that S_j is the cumulative number of observed faults, V_j is the expected number of cumulative faults at time t_j, and R is the long-term expected number of cumulative faults for a software system (see Jeske et al. (2008)), i.e., $R = \lim_{t\to\infty} m(t; \boldsymbol{\theta}_2) \cdot \mu_X$. For the NHDCP model, (S_j, V_j, R) in (13) are correspoinding to $(Y_{ij}, m(t_j; \hat{\boldsymbol{\theta}}_{\mathbf{2}})\hat{\mu}_X, \hat{R})$ and p_j is replaced by $e_j/\hat{R} = [m(t_j; \hat{\boldsymbol{\theta}}_{\mathbf{2}}) - m(t_{j-1}; \hat{\boldsymbol{\theta}}_{\mathbf{2}})]\hat{\mu}_X/\hat{R}$.

Using these notations, (13) becomes

$$\begin{aligned}
A_i^2 &= \sum_{j=1}^{L} \frac{[Y_{ij} - m(t_j; \hat{\boldsymbol{\theta}}_{\mathbf{2}})\hat{\mu}_X]^2 \cdot [m(t_j; \hat{\boldsymbol{\theta}}_{\mathbf{2}}) - m(t_{j-1}; \hat{\boldsymbol{\theta}}_{\mathbf{2}})]}{m(t_j; \hat{\boldsymbol{\theta}}_{\mathbf{2}})(\hat{R} - m(t_j; \hat{\boldsymbol{\theta}}_{\mathbf{2}})\hat{\mu}_X)} \\
W_i^2 &= \frac{1}{\hat{R}^2} \cdot \sum_{j=1}^{L} [Y_{ij} - m(t_j; \hat{\boldsymbol{\theta}}_{\mathbf{2}})\hat{\mu}_X]^2 \cdot [m(t_j; \hat{\boldsymbol{\theta}}_{\mathbf{2}}) - m(t_{j-1}; \hat{\boldsymbol{\theta}}_{\mathbf{2}})]\hat{\mu}_X \\
U_i^2 &= \frac{1}{\hat{R}^2} \cdot \sum_{j=1}^{L} [Y_{ij} - m(t_j; \hat{\boldsymbol{\theta}}_{\mathbf{2}})\hat{\mu}_X - \bar{Z}_i]^2 \cdot [m(t_j; \hat{\boldsymbol{\theta}}_{\mathbf{2}}) - m(t_{j-1}; \hat{\boldsymbol{\theta}}_{\mathbf{2}})]\hat{\mu}_X,
\end{aligned}$$

where $\bar{Z}_i = \sum_{j=1}^{L} [Y_{ij} - m(t_j; \hat{\boldsymbol{\theta}}_{\mathbf{2}})\hat{\mu}_X]/L$.

References

Chatterjee, S. and Singh, J. B., (2014). "A NHPP based software reliability model and optimal release policy with logistic-exponential test coverage under imperfect debugging". *International Journal of System Assurance Engineering and Management*, 5, 3, pp. 399-406.

Choulakian, V., Lockhart, R. A., Stephens, M. A. (1994). "Cramer-von Mises Statistics for Discrete Distribution". *The Canadian Journal of Statistics*, 22, pp. 125-137.

Cox, D. R. and Lewis, P. A. (1966), *Statistical Analysis of Series of Events.* Methuen's Monographs on Applied Probability and Statistics, London: Springer.

Debroy, V. and Wong, E. E. (2009). "Insights on Fault Interference for Programs with Multiple Bugs". *20th International Symposium on Software Reliability Engineering*, pp. 165-174.

Dohi, T. and Yasui, K. (2003). "Software Reliability Assessment Models Based on Cumulative Bernoulli Trial Processes". *Mathematical and Computer Modelling*, 38, pp. 1177-1184.

Duane, J. T. (1964). "Learning Curve Approach to Reliability Monitoring". *IEEE Transactions on Aerospace*, AS-2, pp. 563-566.

Efron, B. and Tibshirani, R. J. (1993). *An Introduction to the Bootstrap.* New York: Chapman & Hall.

Goel, A. L. and Okumoto, K. (1979). "Time-Dependent Error Detection Rate Model for Software Reliability and Other Performance Measures". *IEEE Transactions on Reliability*, 28, pp. 206-211.

Goševa-Popstojanova, K. and Trivedi, K. S. (2000). "Failure Correlation in Software Reliability Models". *IEEE Transactions on Reliability*, 49, pp. 37-48.

Hamill, M. and Goševa-Popstojanova, K. (2009). "Common Trends in Software Fault and Failure Data". *IEEE Transactions on Software Engineering*, 35, pp. 484-496.

Hsieh, M. H., Jeng, S. L., and Shen, P. S. (2009). "Assessing Device Reliability Based on Scheduled Discrete Degradation Measurements". *Probabilistic Engineering Mechanics*, 24, pp. 151-158.

Huang, C. Y. and Kuo, S. Y. (2002). "Analysis of Incorporating Logistic Testing-Effort Function into Software Reliability Modeling". *IEEE Transactions on Reliability*, 51, pp. 261-270.

Huang, C. Y., Lyu, M. R., and Kuo, S. Y. (2003). "A Unified Scheme of Some Nonhomogeneous Poisson Process Models for Software Reliability Estimation". *IEEE Transactions on Software Engineering*, 29, pp. 261-269.

Huang, C. Y. and Huang, W. C. (2008). "Software Reliability Analysis and Measurement Using Finite and Infinite Server Queueing Models". *IEEE Transactions on Reliability*, 57, pp. 192-203.

Huang, C. Y. and Lin, C. T. (2010). "Analysis of Software Reliability Modeling Considering Testing Compression Factor and Failure-to-Fault Relationship". *IEEE Transactions on Computers*, 59, pp. 283-288.

Inoue, S. and Yamada, S. (2007). "Generalized Discrete Software Reliability with Effect of Program Size". *IEEE Transactions on Systems, Man, and Cybernetics–Part A: Systems and Humans*, 37, pp. 170-179.

Jeske, D. R., Zhang, X., and Pham, L. (2005). "Adjusting Software Failure Rates that are Estimated from Test Data". *IEEE Transactions on Reliability*, 54, pp. 107-114.

Jeske, D. R., Lockhart R. A., Stephens, M. A., and Zhang, Q. (2008). "Cramer-von Mises Tests for the Compatibility of Two Software Operating Environments". *Technometrics*, 50, pp. 53-63.

Kuo, L. and Yang, T. Y. (1996). "Bayesian Computation for Nonhomogeneous Poisson Processes in Software Reliability". *Journal of the American Statistical Association*, 91, pp. 763-773.

Kvam, P. H. and Vidakovic, B. (1998). *Nonparametric Statistics with Applications to Science and Engineering.* New York: John Wiley & Sons.

Lawless, J. F. (2003). *Statistical Models and Methods for Lifetime Data.* Hoboken, N. J.: Wiley-Interscience.

Lo, J. H. and Huang, C. Y. (2006). "An Integration of Fault and Correction Process in Software Reliability Analysis". *The Journal of Systems and Software*, 79, pp. 1312-1323.

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

Musa, J. D. and Okumoto, K. (1984). "A Logarithmic Poisson Execution Time Model for Software Reliability Measurement". *Proceedings of the Seventh International Conference on Software Engineering*, Florida, March 26-29, pp. 230-238.

Musa, J. D., Lannino, A., and Okumoto, K. (1987). *Software Reliability: Measurement, Prediction, Application.* New York: McGraw-Hill.

Musa, J. D. (1998). *Software Reliability Engineering*, New York: McGraw-Hill.

Pfefferman, J. D. and Cernuschi-Frias, B. (2002). "A Non-Parametric Non-Stationary Procedure for Failure Prediction". *IEEE Transactions on Reliability*, 51, pp. 434-442.

Pham, H. (2000). *Software Reliability*, Singapore: Springer-Verlag.

Ross, S. M. (2003). *Introduction to Probability Models*, San Diego: Academic Press.

Sahinoglu, M. (1992). "Compound-Poisson Software Reliability Model". *IEEE Transactions on Software Engineering*, 18, pp. 624-630.

Teng, X. and Pham, H. (2006). "A New Methodology for Predicting Software Reliability in the Random Field Environments". *IEEE Transactions on Reliability*, 55, pp. 458-468.

Wang, W. L., Hemminger, T. L., and Tang, M. H. (2007). "A Moving Average Non-Homogeneous Poisson Process Reliability Growth Model to Account for Software with Repair and System Structures". *IEEE Transactions on Reliability*, 56, pp. 411-421.

Wilson, S. P. and Samaniego, F. J. (2007). "Nonparametric Analysis of the Order-Statistic Model in Software Reliability". *IEEE Transactions on Software Engineering*, 33, pp. 198-208.

Yamada, S. (2014). *Software Reliability Modeling, Fundamentals and Applications*, Springer-Verlag.

Yamada, S., Ohba, M., and Osaki, S. (1983). "S-Shaped Reliability Growth Modeling for Software Error Detectio"". *IEEE Transactions on Reliability*, 32, pp. 475-478.

Yamada, S. and Osaki, S. (1983). "Reliability Growth Models for Hardware and Software Systems Based on Nonhomogeneous Poisson Process: A Survey". *Microelectron Reliability*, 23, pp. 91-112.

Zachariah, B. and Rattihalli, R. N. (2007). "Failure Size Proportional Models and an Analysis of Failure Detection Abilities of Software Testing Strategies". *IEEE Transactions on Reliability*, 56, pp. 246-253.

Zachariah, B. (2012). "Analysis of software testing strategies through attained failure size". *IEEE Transactions on Reliability*, 61, 2, pp. 569-579.

Zachariah, B. (2015). "Optimal Stopping Time in Software Testing Based on Failure Size Approach". *Annals of Operations Research*, 235, pp. 771-784.

Zacks, S. (2005). "Some Recent Results on the Distributions of Stopping Times of Compound Poisson Processes with Linear Boundaries". *Journal of Statistical Planning and Inference*, 130, pp. 95-109.

Zhang, X. and Pham, H. (2000). “Comparisons of Nonhomogeneous Poisson Process Models and its Applications”. *International Journal of Systems Science*, 31, pp. 1115-1123.

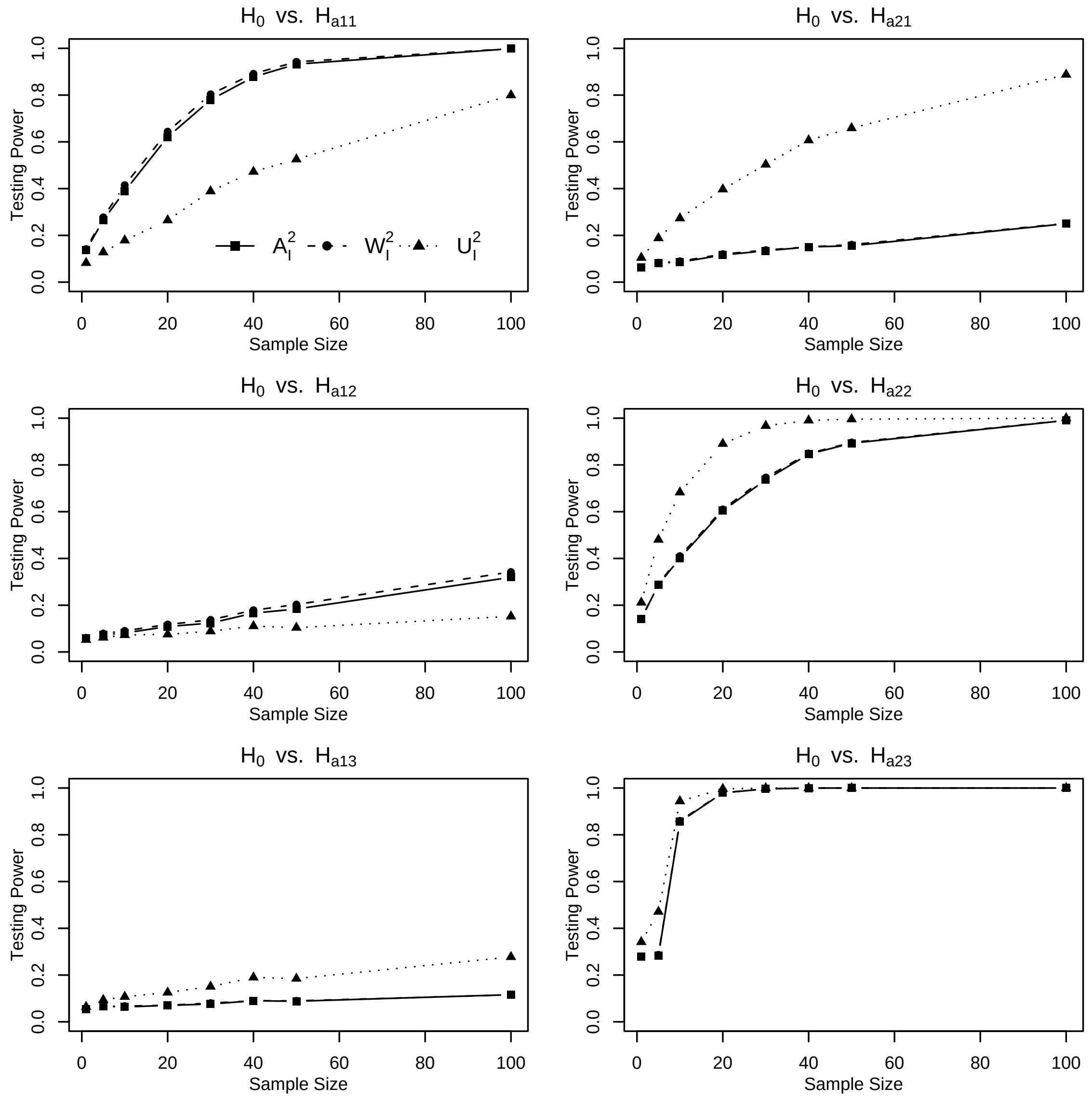

Figure 6: Testing Power for six alternative hypotheses in (11) using three GOF tests: Anderson-Darling (solid line), Cramér von-Mises (dashed line) and Watson (dotted line).