A Documentation Structure for Software Reliability Growth Models

A. Di Bucchianico[*] and I. Corro Ramos
Eindhoven University of Technology
Eindhoven
The Netherlands

E. Brandt and R. Henzen
Refis
Bilthoven
The Netherlands

Abstract

In spite of the rich literature on software reliability growth models, it is not always easy to find correct or sufficiently detailed information on these models. As an attempt to improve upon this situation, we present a general documentation structure for software reliability growth models based on best practices from statistics. To illustrate our documentation structure, we sketch the details for the Jelinski-Moranda and Goel-Okumoto models. Even for these well-studied, simple models there are not widely known as well as not yet available results. Finally, we report on the status of a new tool that we are developing to incorporate our approach.

Keywords software reliability growth models, software reliability, software testing, NHPP models, model selection matrix, trend test, goodness-of-fit test, ML estimation, LS estimation, statistical software `R`.

1 Introduction

In spite of the rich literature on software reliability growth models including monographs like Lyu (1996), it is not always easy to find correct or sufficiently detailed information on these models. Common problems include vague mathematical descriptions, incorrect use of asymptotic results from mathematical statistics, lack of universal agreement on assumptions behind these models, and lack of attention for numerical instabilities in parameter estimation algorithms. As an attempt to improve upon this situation, we present a general documentation structure for software reliability growth models based on best practices from statistics. Our documentation structure contains standard elements like model assumptions, allowed data formats, trend tests etc. as well as less common elements like existence results for parameter estimates and numerically stable procedures to compute estimates for parameters. In this paper we sketch details of our documentation structure for the well-known Jelinski-Moranda and Goel-Okumoto models. Finally, we report on a new tool that we are developing to incorporate our approach. The tool is built around the free open source package `R` (`www.r-project.org`), which is rapidly becoming the standard within the statistics community.

This paper is organized as follows. In Section 2 we establish standard notation. Our documentation structure is presented in Section 3, while specifics for the Jelinski-Moranda and Goel-Okumoto models can be found in

2 Notation

In this section we establish some notation. We adopt the general notation that N is the unknown initial number of errors and that $T_1 < T_2 < \ldots < T_n$ $(n < N)$ are error detection times (failure times), so $T_1, T_2 - T_1, \ldots, T_n - T_{n-1}$ are times between detection of errors. In this case we speak of exact

[*]Corresponding author: email `a.d.bucchianico@tue.nl`

or ungrouped data. Defect reports often only contain weekly summaries of defects. In such cases, the observations are defect counts m_i in intervals (T_{i-1}, T_i) for $1 \leq i \leq n$ (and hence, $\sum_{i=1}^{n} m_i$ is the total number of detected errors). This form of data is called interval or grouped data. It is often convenient to define $T_0 = 0$ (often implicitly assumed in many papers), so that the times between errors can be written as in terms of the cumulative times as $X_1 = T_1 - T_0, X_2 = T_2 - T_1, \ldots, X_n = T_n - T_{n-1}$. Software reliability growth models arise when one attaches probability distributions to these quantities.

If we assume perfect immediate repair of defects, then like in hardware reliability systems with minimal repair we must interpret the times $T_1, T_2, \ldots, T_N$ as event times of a point process. A point process $(N(t))_{t>0}$ is a collection of random variables with non-negative integer values such that $N(t)$ is the number of events up to time t, so $N(t) = \{\#i \mid T_i \leq t\}$. The case of failure counts as discussed above perfectly fits in if we distinguish between (possibly hidden) detection events described by $(N(t))_{t>0}$ and the available observations T_i or (T_i, m_i).

In hardware reliability it is common to define distributions by failure rates instead of densities or distribution functions. It is important to note that the concept of failure rate in software reliability models is the failure rate of the associated point process $(N(t))_{t>0}$ rather than the failure rate concept of a random variable as is used in hardware reliability in the case of non-repairable systems (see Thompson (1981) and Thompson (1988) for a detailed discussion).

3 Model documentation structure

In this section we describe a documentation structure for software reliability growth models. For several programming languages there exist explicit coding standards. Although such explicit standards are hard to find in statistics, there are best practices commonly agreed upon by statisticians (like using both graphical and formal ways to analyze data). We tried to make these practices explicit in our documentation structure. Since we focus on model documentation, we left out important practical issues like data cleaning.

Our documentation structure is targeted at a classical statistical analysis for such models. A similar documentation structure could be set up for Bayesian analyses.

Table 1: Documentation structure

1. Probability model (a) joint distribution of $(N(t))_{t>0}$ and/or $T_1, T_2, \ldots, T_n$ and/or $X_1, X_2, \ldots, X_n$ (including (in)dependence structure) (b) model assumptions (c) interpretation of model parameters
2. Trend analysis (a) graphical methods i. interpretation of graph ii. algorithm to produce graph (b) formal tests i. test statistics (if necessary, algorithm to compute test statistics) ii. description of hypotheses associated to the tests iii. distribution of test statistics under null hypothesis iv. algorithms to compute significance values v. power results of tests
continued on next page

Table 1: (continued)

3. Parameter estimation
   - (a) point estimation procedures for parameters (Maximum Likelihood and/or Least Squares)
     - i. data requirements
     - ii. existence results for parameter estimates
     - iii. performance of parameter estimators (bias, efficiency)
     - iv. algorithms to compute parameter estimates
   - (b) confidence interval procedures for parameters (Maximum Likelihood and/or Least Squares)
     - i. distributional description of underlying estimators
     - ii. algorithms to compute confidence intervals

4. Model validation
   - (a) graphical methods
     - i. interpretation of graphs
     - ii. algorithm to produce graphs
   - (b) goodness-of-fit tests
     - i. test statistics (if necessary, algorithm to compute test statistics)
     - ii. description of hypotheses associated to the goodness-of-fit tests
     - iii. distribution of test statistics under null hypothesis
     - iv. algorithms to compute significance values
     - v. power results of tests

5. Prediction
   - (a) list relevant predictions for model (should preferably include reliability/time to next error, intensity function, remaining number of errors)
   - (b) point estimates for items mentioned in a)
     - i. performance of estimators (bias, efficiency)
     - ii. algorithms to compute estimates
   - (c) confidence intervals for items mentioned in a)
     - i. distributional description of confidence intervals
     - ii. algorithms to compute confidence intervals

Remarks:

- existence results of parameter estimates are useful in order to avoid misinterpreting convergence problems
- the model assumptions should preferably be translations of the model structure to practical assumptions; it is worthwhile to mention both active requirements (constraints like all errors are of the same type) as well as lack of requirements (errors may be repaired in an imperfect way)
- all items should preferably be for both exact (ungrouped) data and interval (grouped) data
- distributional results should state whether they are exact or asymptotic; if results are asymptotic, then it should be indicated which parameters go to infinity

- algorithms should be numerically stable (cf. Yin and Trivedi (1999));
- all details must be accompanied by references to publicly available literature.

4 Two examples: the Jelinski-Moranda and Goel-Okumoto models

4.1 Jelinski-Moranda

1. Probability model
   (a) the X_i's are independent exponentially distributed random variables with mean $\lambda/(N-i+1)$ or equivalently the T_i's are the order statistics of a sample of size N of exponentially distributed random variables with parameter λ (see Miller (1986)). The random process $(N(t))_{t>0}$ cannot be characterized easily, but the random variable $N(t)$ for fixed t follows a binomial distribution with parameters N and $F(t)$, where F is the distribution function of an exponential random variable with mean λ.

   (b) The underlying assumptions are:
      i. number of initial errors is unknown but fixed and finite
      ii. immediate and perfect repair
      iii. all errors are of the same type
      iv. errors are being detected independently of each other.

   (c) N is the number of initial errors, λ is the rate at which each individual error will be detected during testing.

2. Trend analysis
   (a) For exact data, a plot of the cumulative error times versus the cumulative number of detected errors or a running average of it is appropriate. A concave plot indicates software reliability growth. For interval data, similar plots can be constructed.

   (b) For exact data, the Laplace test can be used. Under the hypothesis that the X_i's are i.i.d. exponentially distributed, its distribution is the sum of i.i.d. uniform distributions, hence asymptotically normal. Another test is the Military Handbook Test, which under the same null distribution is exactly χ^2_{2n}-squared distributed. Details can be found in *e.g.*, Rigdon and Basu (2000). Similar tests for interval data seem not to have been studied in the software reliability literature (but see Weller and Ryan (1998) for trend tests in another context).

3. Parameter estimation
   (a) Estimators exist for both exact and interval data.

   (b) i. Existence criteria for the ML estimators for exact data can be found in Finkelstein et al. (1999), Moek (1984) and Osborne and Severini (2000). Non-existence or degeneracy in this case is related to software deterioration as shown in Littlewood and Verrall (1981). Existence criteria for the ML estimators for interval data can be found in Knafl (1992).
      ii. The standard estimator for N for exact data is severely biased. Bias corrections have been introduced (Joe and Reid (1985), Osborne and Severini (2000)).
      iii. Finkelstein et al. (1999) contains a simple algorithm for ML estimators, improving on an earlier attempt in Joe and Reid (1985).

   (c) Only results for exact data are known:
      i. Standard asymptotics do not work since the parameter N causes the standard assumptions to be violated. Joe (1989) and Van Pul (1992) contain correct asymptotics for $N \to \infty$ and $T_n \to \infty$, respectively.

ii. Van Pul (1992) shows by simulation that asymptotic confidence intervals based on the Wilks likelihood ratio test are superior with respect to convergence to the asymptotic distribution. Finkelstein et al. (1999) contains an algorithm to approximate exact confidence intervals.

4. Model validation

   (a) Since the T_i's can be interpreted as order statistics, standard graphical procedures for right-censored samples like quantile plots can be used.

   (b) Although the T_i's can be interpreted as order statistics, standard goodness-of-fit procedures for right-censored samples should be used with caution since N is a parameter. See (Van Pul, 1993, Chapter 6) for a rigorous asymptotic approach. Convergence, however, is very slow.

5. Prediction

   (a) Reliability (the time to the next error) is exponentially distributed with mean $\lambda/(N-n+1)$, from which one can easily determine point estimators as well as confidence intervals.

   (b) The intensity function of the associated order statistic process is $N/\mu\, e^{-t/\mu}$, so by the Invariance Principle the ML estimator equals $\widehat{N}/\widehat{\mu}\, e^{-t/\widehat{\mu}}$.

   (c) A point estimator for the remaining number of errors is $\widehat{N}-n$. Confidence intervals can be derived from those for N.

4.2 Goel-Okumoto

1. Probability model

   (a) $(N(t))_{t\geq 0}$ is a non-homogeneous Poisson process with cumulative intensity function $\Lambda(t) = a\left(1-e^{-bt}\right)$. In particular, $\Lambda(t) = E(N(t))$ and

$$P\left(N(t)-N(s)=k\right) = e^{-(\Lambda(t)-\Lambda(s))}\left(\Lambda(t)-\Lambda(s)\right)^k/k!.$$

   (b) The underlying assumptions are:

      i. number of initial errors is unknown but finite
      ii. immediate and perfect repair of errors
      iii. all errors are of the same type
      iv. errors are being detected independently of each other
      v. the rate at which errors are detected is proportional to the number of remaining errors

   (c) The parameter a is the expected number of errors to be eventually detected, while b is the rate at which each individual error will be detected during testing.

2. Trend analysis

   (a) For exact data, a plot of the cumulative error times versus the cumulative number of detected errors or a running average of it is appropriate. A concave plot indicates software reliability growth. For interval data, similar plots can be constructed.

   (b) For exact data, the Laplace test can be used. Under the hypothesis that the X_i's are i.i.d. exponentially distributed, its distribution is the sum of i.i.d. uniform distributions, hence asymptotically normal. Another test is the Military Handbook Test, which under the same null distribution is exactly χ^2_{2n} distributed. Details can be found in *e.g.*, Rigdon and Basu (2000). Comparisons of these tests with other tests can be found in Bain et al. (1985), Cohen and Sackrowitz (1993) and Wang and Coit (2005). Variations on these tests for NHPP models can be found in Kvaløy and Linqvist (1998). Similar tests for interval data seem not to have been studied in the software reliability literature (cf. Weller and Ryan (1998)).

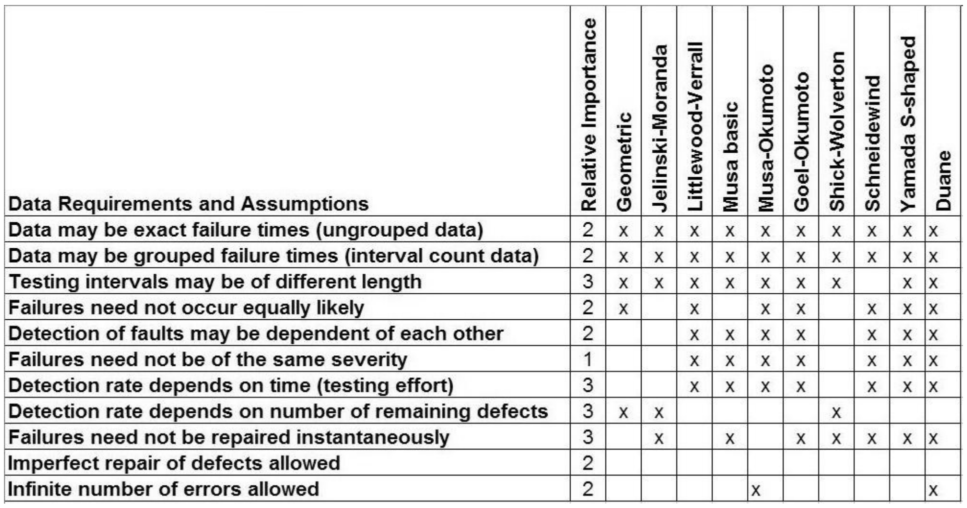
3. Parameter estimation
   (a) Estimators exist for both exact and interval data.
   (b) i. Existence results for ML estimators for exact and interval data can be found in Knafl and Morgan (1996) and Knafl (1992), respectively.
      ii. The ML estimators are not consistent as shown in Jeske and Pham (2001).
      iii. Numerically stable algorithms for ML estimators can be found in Hossain and Dahiya (1993) and Knafl and Morgan (1996) for exact data, and in Hossain and Dahiya (1993) and Knafl (1992) for interval data. (Weighted) Least Squares estimators are discussed in Kuhl et al. (1998).
   (c) Only results for exact data are known:
      i. Joe and Reid (1985) contains distributional results for ML-based intervals.
      ii. No stable algorithms are known.
4. Model validation
   (a) TTT plots may be used to assess goodness-of-fit (see *e.g.* Rigdon and Basu (2000)).
   (b) Specific goodness-of-fit tests for large classes of NHPP models (including the Goel-Okumoto model) can be found in Bhattacharjee et al. (2004) and Zhao and Wang (2005).
5. Prediction
   (a) The ML estimator for the probability that there there will be no error in the next t time units is given by
   $$e^{-\widehat{a}\, e^{-\widehat{b}t}\left(1-e^{-\widehat{b}t_n}\right)}.$$
   Approximate confidence intervals can be obtained by using formulas for error propagation as in Gokhale (2005).
   (b) The ML estimator of the intensity function is $\widehat{a}\widehat{b}\left(1-e^{-\widehat{b}t}\right)$.
   (c) An ML estimator for the remaining number of errors is $\max(0, \widehat{a}-n)$. Confidence intervals can be derived from those for a.

5 Our new tool

In this section we report on the status of a new tool that we are developing to incorporate our approach. Since we were not happy with existing tools, we decided to build a new tool that

- uses well-documented state-of-the-art algorithms
- is platform independent
- encourages to apply best practices from statistics
- can easily be extended to incorporate new models.

In order to meet these requirements we decided to use Java for the interface and the statistical programming language `R` (see `www.r-project.org`) for the statistical computations. `R` is open-source free software maintained by a group of top-level statisticians and is rapidly becoming the standard programming language within the statistical community. Our tool is written in Java and we have made use of readily available components from Java Resource Bundles. The statistical computations are performed by calling R (high-quality free open-source statistical software). The communication of Java with `R` uses JRI and JavaGD libraries developed by the Computational Statistics group of the University of Augsburg. Systematic approaches to use model assumptions and data requirements for initial model selection have not received much attention in the literature, Kharchenko et al. (2002) being an exception. Therefore, we have been developing a matrix-based procedure to support the choice of the models to work with. A simple version of this matrix can be found in Figure 1, while Figure 2 shows a screen shot of the model selection wizard in our tool. Since we wish to select rather than rule out applicable models, we state all assumptions as negations of restrictions. To select models, one first has to select relevant assumptions and weights to incorporate the available information on the testing project at hand. If all requirements and selected assumptions of a model are satisfied, then the score for this model is 100%. In all other cases the score of the model is defined using the relative importance (weight) of the applicable characteristics of the model.



Data Requirements and Assumptions	Relative Importance	Geometric	Jelinski-Moranda	Littlewood-Verrall	Musa basic	Musa-Okumoto	Goel-Okumoto	Shick-Wolverton	Schneidewind	Yamada S-shaped	Duane
Data may be exact failure times (ungrouped data)	2	x	x	x	x	x	x	x	x	x	x
Data may be grouped failure times (interval count data)	2	x	x	x	x	x	x	x	x	x	x
Testing intervals may be of different length	3	x	x	x	x	x	x	x		x	x
Failures need not occur equally likely	2	x		x		x	x		x	x	x
Detection of faults may be dependent of each other	2			x	x	x	x		x	x	x
Failures need not be of the same severity	1			x	x	x	x		x	x	x
Detection rate depends on time (testing effort)	3			x	x	x	x		x	x	x
Detection rate depends on number of remaining defects	3	x	x					x			
Failures need not be repaired instantaneously	3		x		x		x	x	x	x	x
Imperfect repair of defects allowed	2										
Infinite number of errors allowed	2					x					x

Figure 1: Assumption matrix.

Conclusion In this paper we presented a general documentation structure for software reliability growth models. We sketched the details for the Jelinski-Moranda and Goel-Okumoto models. It turns out that for interval (grouped) data less is known than for exact (ungrouped) data. Finally, we reported on a new tool that we are developing to incorporate our approach.

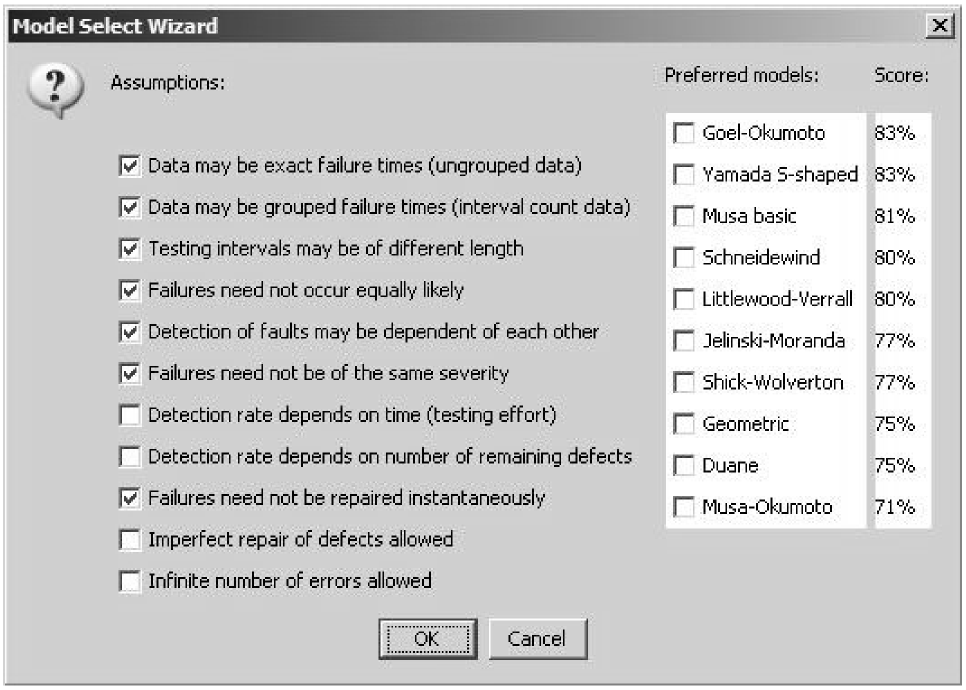
References

J.L. Bain, M. Engelhardt, and F.T. Wright. Tests for an increasing trend in the intensity of a Poisson process. *J. Amer. Stat. Assoc.*, 80:419–422, 1985.

M. Bhattacharjee, J. V. Deshpande, and U. V. Naik-Nimbalkar. Unconditional tests of goodness of fit for the intensity of time-truncated nonhomogeneous Poisson processes. *Technometrics*, 46(3):330–338, 2004.

A. Cohen and H. B. Sackrowitz. Evaluating tests for increasing intensity of a Poisson process. *Technometrics*, 35(4):446–448, 1993.

M. Finkelstein, H.G. Tucker, and J.A. Veeh. Confidence intervals for N in the exponential order statistics problem. *Comm. Statist. Theory Methods*, 28(6):1415–1433, 1999.



Model Select Wizard

Assumptions:

- [✓] Data may be exact failure times (ungrouped data)
- [✓] Data may be grouped failure times (interval count data)
- [✓] Testing intervals may be of different length
- [✓] Failures need not occur equally likely
- [✓] Detection of faults may be dependent of each other
- [✓] Failures need not be of the same severity
- [ ] Detection rate depends on time (testing effort)
- [ ] Detection rate depends on number of remaining defects
- [✓] Failures need not be repaired instantaneously
- [ ] Imperfect repair of defects allowed
- [ ] Infinite number of errors allowed

Preferred models:	Score:
Goel-Okumoto	83%
Yamada S-shaped	83%
Musa basic	81%
Schneidewind	80%
Littlewood-Verrall	80%
Jelinski-Moranda	77%
Shick-Wolverton	77%
Geometric	75%
Duane	75%
Musa-Okumoto	71%

Figure 2: Screen shot of model selection wizard.

S.S. Gokhale. Variance expressions for software reliability growth models. In *Proceedings of the Annual Reliability and Maintainability Symposium*, pages 628–633, 2005.

S.A. Hossain and R.C. Dahiya. Estimating the parameters of a non-homogeneous Poisson-process model for software reliability. *IEEE Trans. Rel.*, 42(4):604–612, 1993.

D.R. Jeske and H. Pham. On the maximum likelihood estimates for the Goel-Okumoto software reliability model. *Amer. Statist.*, 55(3):219–222, 2001.

H. Joe. Statistical inference for General-Order-Statistics and Nonhomogeneous-Poisson-Process software reliability models. *IEEE Trans. Software Eng.*, 15(11):1485–1490, 1989.

H. Joe and N. Reid. Estimating the number of faults in a system. *J. Amer. Stat. Assoc.*, 80(389):222–226, 1985.

V.S. Kharchenko, O.M. Tarasyuk, V.V. Sklyar, and V.Yu. Dubnitsky. The method of software reliability growth models choice using assumptions matrix. In *COMPSAC '02: Proceedings of the 26th International Computer Software and Applications Conference on Prolonging Software Life: Development and Redevelopment*, pages 541–546, Washington, DC, USA, 2002. IEEE Computer Society.

G.J. Knafl. Solving Maximum Likelihood equations for two-parameter software reliability models for using grouped data. In *Proceedings of the International Symposium on Software Reliability Engineering,*, pages 205–213. IEEE, 1992.

G.J. Knafl and J. Morgan. Solving ML equations for 2-parameter Poisson-process models for ungrouped software-failure data. *IEEE Trans. Rel.*, 45(1):42–53, 1996.

M.E. Kuhl, H. Damerdji, and J.R. Wilson. Least Squares estimation of nonhomogeneous Poisson processes. In D.J. Medeiros, E.F. E.F. Watson, J.S. Carson, and M.S. Manivannan, editors, *Proceedings of the 1998 Winter Simulation Conference*, pages 637–645, 1998.

J. T. Kvaløy and B.H. Linqvist. TTT-based tests for trend in repairable systems data. *Rel. Eng. System Safety*, 60:13–28, 1998.

B. Littlewood and J. L. Verrall. Likelihood function of a debugging model for computer software reliability. *IEEE Trans. Rel.*, 30:145–148, 1981.

M.R. Lyu, editor. *Handbook of Software Reliability Engineering*. McGraw-Hill and IEEE Computer Society, New York, 1996.

D.R. Miller. Exponential order statistic model of software reliability growth. *IEEE Trans. Softw. Eng.*, 12 (1):12–24, 1986.

G. Moek. Comparison of some software reliability models for simulated and real failure data. *Int. J. Modelling Sim.*, 4:29–41, 1984.

J.A. Osborne and T.A. Severini. Inference for exponential order statistic models based on an integrated likelihood function. *J. Amer. Statist. Assoc.*, 95(452):1220–1228, 2000.

S.E. Rigdon and A.P. Basu. *Statistical Methods for the Reliability of Repairable Systems*. Wiley, 2000.

W. A. Thompson, Jr. *Point Process Models with Applications to Safety and Reliability*. Chapman and Hall, New York, 1988.

W. A. Thompson, Jr. On the foundations of reliability. *Technometrics*, 23(1):1–13, 1981.

M.C.J. Van Pul. Simulations on the Jelinski-Moranda model of software reliability; application of some parametric bootstrap methods. *Stat. Computing*, 2:121–136, 1992.

M.C.J. Van Pul. *Statistical Analysis of Software Reliability Models*, volume 95 of *CWI Tract*. Centrum voor Wiskunde en Informatica, Amsterdam, 1993.

P. Wang and D.W. Coit. Repairable systems reliability trend tests and evaluation. In *RAMS 2005 Proceedings*, pages 416–421, 2005.

E.A. Weller and L.M. Ryan. Testing for trend with count data. *Biometrics*, 54:762–773, 1998.

L. Yin and K.S. Trivedi. Confidence interval estimation of NHPP-based software reliability models. In *Proc. 10th Int. Symp. Software Reliability Engineering (ISSRE 1999)*, pages 6–11, 1999.

J. Zhao and J. Wang. A new goodness-of-fit test based on the Laplace statistic for a large class of NHPP models. *Comm. Statist. Simulation Comput.*, 34(3):725–736, 2005.