Journal of Uncertain Systems
Vol.12, No.4, pp.292-308, 2018
Online at: www.jus.org.uk

Imprecise Probabilistic Inference for Software Run Reliability Growth Models

Lev V. Utkin[1], Frank P.A. Coolen[2,*]

[1] *Telematics Department, Central Scientific Research Institute of Robotics and Technical Cybernetics*
Peter the Great Saint-Petersburg Polytechnic University, St. Petersburg, Russia

[2] *Department of Mathematical Sciences, Durham University, UK*

Received 8 January 2018; Revised 12 October 2018

Abstract

This paper presents the application of an inferential statistical approach which combines imprecise Bayesian methods with likelihood inference, to a standard software run reliability growth model. The main idea of the approach is to divide the set of model parameters into two subsets related to fundamentally different aspects of the overall model, and to combine an imprecise Bayesian method related to one of the subsets of the model parameters with maximum likelihood estimation for the other subset. In accordance with the first subset and statistical data, the imprecise Bayesian model is constructed, which provides lower and upper predictive probability distributions depending on the second subset of parameters. These further parameters are then estimated by a maximum likelihood method. This method is applied to a basic software run reliability growth model and it is shown to perform better than a standard model. Several aspects related to the method are discussed, including its advantages, its wider applicability and the possibility to include relevant expert judgements.

Keywords: Bayesian inference, imprecise probabilities, lower and upper probability distributions, maximum likelihood estimation, software run reliability growth

1 Introduction

One of the main goals of analysing software system development is to predict its future reliability based on past experience, for which one typically constructs a statistical model to quantify uncertainties and to enable learning from data. There is a variety of statistical theories and methods for such inference, and researchers often strongly advocate one specific general theory, e.g. the Bayesian approach, whilst rejecting other approaches that also have their merits. In this paper we explore combined use of imprecise Bayesian methods [1, 11], where sets of prior distributions are used, with maximum likelihood estimation, both on different subsets of all parameters appearing in a statistical model. At first look, these methods may appear to have little in common and one may favour either a complete (imprecise) Bayesian approach or maximum likelihood estimation of all parameters. However, if one considers a Bayesian approach as using a weighted likelihood function, with weights reflecting prior knowledge, then the two methods are less contradictory and exploration of the opportunity to combine both into a hybrid method is of interest. In this paper we develop this hybrid method for a generalization of a well-known software run reliability model. Detailed fundamental analysis and further exploration of this hybrid approach will be important for its full justification, in particular with regard to possible interpretations of the resulting inferences; this is left as a topic for future research.

An important feature of many systems is change of some of their characteristics over time, which has to be taken into account when constructing a statistical model for the system. For example, a common approach for measuring software reliability [15] is by using a statistical model whose parameters are generally estimated from available data on software failures, and the model may be obtained by observing the overall trend of reliability growth during the debugging process. In other words, a software reliability growth model describes

*Corresponding author.
Email: frank.coolen@durham.ac.uk (F.P.A. Coolen).

how observation of failures, and correcting the underlying faults, such as occurs in software development when the software is being tested and debugged, affect the reliability of software. The word "growth" is rather conventional to describe reliability models with important characteristics changing over time, it does not restrict use of such models to systems whose reliability actually improves. In other words, a growth model can be regarded to be a mathematical expression which fits experimental data from systems with some important changes over time.

Suppose that $X_1, ..., X_n$ is a series of random variables with X_i the number of successful software runs between the $(i-1)$-th and i-th software failures, for $i = 1, \ldots, n$. We suppose that variable X_i is governed by a probability distribution function $p_i(x \mid \mathbf{b}, \mathbf{d})$ depending on two vectors of parameters $\mathbf{b}$ and $\mathbf{d}$. The vector $\mathbf{b}$ contains parameters of the probability distribution under consideration. The vector $\mathbf{d}$ of parameters characterizes the growth, i.e., the growth is modelled by a function $f(i, \mathbf{d})$ which characterizes the change of the system behaviour ('growth'). For example in software reliability analysis, the function f mainly shows how parameters $\mathbf{b}$ of the probability distribution p_i change with the number i of corrected errors or faults. Generally, the vector $\mathbf{b}$ depends on $\mathbf{d}$ and the number i, which is the index of the random variable X_i under consideration.

It should be noted that the growth function is explicitly stated in some models. For instance, Littlewood and Verrall [6] suggest software reliability models with linear and quadratic forms for the function f with two parameters $\mathbf{d} = (d_0, d_1)$: $f(i, \mathbf{d}) = d_0 + d_1 i$ or $f(i, \mathbf{d}) = d_0 + d_1 i^2$. In these models, the growth function is included as parameter of a gamma distribution, which changes with the number of corrected errors in the software.

Clearly, the growth function f may model different characteristics. In software reliability models, it typically enables possible changes of the parameters $\mathbf{b}$ of the probability distribution of random variables X_i to reflect actual changes to software systems, mostly due to fault corrections. In such models, we assume a form of f and wish to learn about the parameters $\mathbf{d}$ of f from data.

There are several approaches for inference about growth models on the basis of statistical data. Nowadays, the most popular inferential methods tend to use the likelihood function as main mechanism to link model parameters and statistical data. For models such as reliability growth models, estimation is required both for parameters of the basic probability model and parameters explicitly modelling the growth behaviour. This may involve a substantial number of parameters, with possibly relatively few data available. In this paper, we explore a possible way for dealing with this, by considering imprecise Bayesian inference for one subset of parameters, and a maximum likelihood approach to estimate the other subset of parameters. Such imprecise Bayesian inference has been presented, without a link to maximum likelihood for further parameters, by Walter, Augustin and Peters [14] with application to linear regression models. Typically, a precise parametric model is assumed, with imprecision following through the use of sets of conjugate prior distributions [2, 8, 13]. It is theoretically feasible to use sets of priors for all parameters combined, but this may well lead to very wide posterior intervals for inferences of interest. Furthermore, if one can estimate some of the parameters by means of maximum likelihood methods, it could also be attractive with regard to not having to assign informative (sets of) prior distributions, in particular if they are on a feature about which no clear expert judgement is available or which one strongly wishes to infer from the data. Exploring this possibility to combine imprecise Bayesian inference for some of the model parameters with maximum likelihood estimation for other parameters, is the main aim of this paper. It is particularly studied for software reliability growth models as their parameters naturally divide into two groups, as will be discussed in Section 4.

The approach we propose in this paper is as follows. By using imprecise Bayesian inference, we can exclude all the parameters of the vector $\mathbf{b}$ from the model, and derive a set of predictive cumulative distribution functions (CDFs) such that their lower and upper bounds are conditional on all the parameters of the vector $\mathbf{d}$. This is followed by estimation of the parameters of the vector $\mathbf{d}$, for which we use the modified maximum likelihood estimation method which is explained in Section 2. This approach allows us to reduce the number of parameters in the model and to maximize the likelihood function only over parameters of the vector $\mathbf{d}$ without considering the parameters of vector $\mathbf{b}$. Even further, it can be applied if one explicitly wishes to take expert judgement into account on the part of the model corresponding to parameters $\mathbf{b}$, and this expert judgement is best reflected by imprecise probabilities, while no such prior information is available for the model aspects related to parameters $\mathbf{d}$, for which, however, one can use process data.

This paper is organized as follows. Section 2 explains the modified maximum likelihood estimation. Section 3 presents the imprecise Bayesian models used as part of the full model. Section 4 presents a general scheme for the full model construction. Section 5 presents the software run reliability growth model considered in this

paper, which is illustrated in Section 6 and compared to a standard model in Section 7. Section 8 explains how additional prior information can be included in the model. The paper ends with concluding remarks in Section 9.

2 Modified Maximum Likelihood Estimation

Let $\mathbf{K} = (k_1, ..., k_n)$ be a realization of $X_1, ..., X_n$, where the k_i, $i = 1, \ldots, n$, are non-negative integers. If probability distributions $p_i(k_i \mid \mathbf{b}, \mathbf{d})$ of the random variables X_i, $i = 1, ..., n$, are known or assumed, then the standard way for obtaining the parameters $\mathbf{b}$ and $\mathbf{d}$ of a growth model is to maximize the likelihood function

$$L(\mathbf{K} \mid \mathbf{b}, \mathbf{d}) = \prod_{i=1}^{n} p_i(k_i \mid \mathbf{b}, \mathbf{d})$$

over a set of parameters $\mathbf{b}$ and $\mathbf{d}$. Values of the parameters $\mathbf{b}$ and $\mathbf{d}$ should be chosen in order to maximize $L(\mathbf{K} \mid \mathbf{b}, \mathbf{d})$.

Many well-known software reliability growth models presented in the literature have been implemented with such standard maximum likelihood estimation. Such models differ only by assumptions about the probability distributions p_i and the growth function f. For example, for p_i the exponential distribution is used in the Jelinski-Moranda model [5], while the Rayleigh distribution is used in the Schick-Wolverton model [10] and the Beta distribution in the Littlewood-Verrall model [6].

Suppose that the random variable X_i is governed by an unknown CDF $F_i(k)$ which is only known to belong to the set $\mathcal{M}_i(\mathbf{d})$ defined by the lower and upper CDFs

$$\underline{F}_i(k \mid \mathbf{d}) = \inf_{\mathcal{M}_i(\mathbf{d})} F(k), \tag{1}$$

$$\overline{F}_i(k \mid \mathbf{d}) = \sup_{\mathcal{M}_i(\mathbf{d})} F(k). \tag{2}$$

It should be noted that the set $\mathcal{M}_i(\mathbf{d})$ is the set of *all* CDFs bounded by $\underline{F}_i(k \mid \mathbf{d})$ and $\overline{F}_i(k \mid \mathbf{d})$, so it is *not* the set of parametric distributions having the same parametric form as the bounding distributions. This is an important feature of the proposed approach for combined imprecise Bayesian and likelihood inference in this paper. Moreover, the bounds $\underline{F}_i(k \mid \mathbf{d})$ and $\overline{F}_i(k \mid \mathbf{d})$ are assumed not to depend on the parameters $\mathbf{b}$, which is achieved by taking the predictive CDFs resulting from the imprecise Bayesian approach applied with regard to the parameters $\mathbf{b}$.

The likelihood function can be written in the following form:

$$L(\mathbf{K} \mid \mathbf{d}) = \Pr\{X_1 = k_1, \cdots, X_n = k_n\}.$$

Proposition 1 explains how the above likelihood function is maximized over all distributions belonging to $\mathcal{M}_1(\mathbf{d}), ..., \mathcal{M}_n(\mathbf{d})$.

Proposition 1. *Suppose that discrete random variables $X_1, ..., X_n$ are governed by a probability distribution $F(k)$ from sets $\mathcal{M}_i$ defined by bounds (1)-(2), respectively. If $X_1, ..., X_n$ are independent, then*

$$\max_{\mathcal{M}_1, ..., \mathcal{M}_n} \Pr\{X_1 = k_1, \cdots, X_n = k_n\} = \prod_{i=1}^{n} \left\{\overline{F}_i(k_i) - \underline{F}_i(k_i - 1)\right\}. \tag{3}$$

Proof. Denote $N = \{1, 2, ..., n\}$, $\mathbf{M} = (m_1, ..., m_n)$. Let $I_{\{1,...,k_i\}}(m)$ be the indicator function taking the value 1 if $m \leq k_i$. The indicator functions are used in order to represent all probabilities in the form of expectations of indicator functions and to write the natural extension for computing the maximal value of $\Pr\{X_1 = k_1, \cdots, X_n = k_n\}$ in its standard form. The upper bound for the joint probability $\Pr\{X_1 = k_1, \cdots, X_n = k_n\}$ can be found by solving the following optimization problem:

$$\max \sum_{m_1=1}^{\infty} \cdots \sum_{m_n=1}^{\infty} I_{\{k_1,...,k_n\}}(\mathbf{M}) \prod_{i=1}^{n} p_i(m_i),$$

subject to

$$\sum_{m=1}^{\infty} p_i(m) = 1,$$

$$\underline{F}_i(j) \le \sum_{m=1}^{\infty} I_{\{1,...,j\}}(m) p_i(m) \le \overline{F}_i(j),\ i = 1, ..., n,\ j = 1, 2...$$

The objective function can be rewritten as follows:

$$\prod_{i=1}^{n} \sum_{m_i=1}^{\infty} \left(I_{\{1,..,k_i\}}(m_i) - I_{\{1,..,k_i-1\}}(m_i) p_i(m_i) \right).$$

We introduce new variables

$$F_i(j) = \sum_{m_i=1}^{\infty} I_{\{1,...,j\}}(m_i) p_i(m_i).$$

Then we can rewrite the optimization problem as

$$\max \prod_{i=1}^{n} \left\{ F_i(j) - F_i(j-1) \right\},$$

subject to

$$\underline{F}_i(j) \le F_i(j) \le \overline{F}_i(j),\ \underline{F}_i(j-1) \le F_i(j-1) \le \overline{F}_i(j-1),\ i = 1, ..., n.$$

By using the known rules of interval analysis, we obtain (3), which completes the proof. □

Proposition 1 generalizes the standard likelihood estimation for precise probability models.

3 Imprecise Bayesian Models as a Way for Obtaining the Set $\mathcal{M}$

We now consider how to derive the set $\mathcal{M}(\mathbf{d})$. A straightforward way is to use ideas similar to Walley's imprecise Bayesian approach [12, 13].

3.1 Standard Bayesian Analysis

One of the efficient approaches to estimation of the model parameters is Bayesian analysis [3, 9]. It treats parameters of concern as random variables which are assigned a prior probability distribution before observations become available. If we assume that the random variable has a probability distribution with vector of unknown parameters $\mathbf{b}$, then these parameters would be regarded as random variables with a prior probability density $\pi(\mathbf{b} \mid \mathbf{c})$, characterized by (hyper-)parameters $\mathbf{c}$. In this case, the Bayesian approach can be applied for computing the CDF for the random variable of interest, with the parameter $\mathbf{b}$ integrated out:

$$F(k \mid \mathbf{c}) = \int_{\Omega} F(k \mid \mathbf{b}) \cdot \pi(\mathbf{b} \mid \mathbf{c}) \mathrm{d}\mathbf{b}.$$

Here Ω is the set of values of $\mathbf{b}$.

Central to the Bayesian approach is the derivation of the posterior distribution of the unknown parameters, given both the data and the assumed prior density for these parameters, and achieved by application of Bayes' theorem. Suppose that the prior distribution $\pi(\mathbf{b} \mid \mathbf{c})$ represents our uncertainty with regard to $\mathbf{b}$ prior to collecting information in the form of a set $\mathbf{K} = (k_1, ..., k_n)$ of observed values of independent random variables $X_1, ..., X_n$. Let $p(k)$ be the probability mass function for the observed data k given $\mathbf{b}$. Then the posterior distribution $\pi(\mathbf{b} \mid \mathbf{K}, \mathbf{c})$, which is the conditional distribution of $\mathbf{b}$ given the observed data $\mathbf{K}$ and prior parameters $\mathbf{c}$, is computed as

$$\pi(\mathbf{b} \mid \mathbf{K}, \mathbf{c}) \propto p(k_1) \cdot \cdots p(k_n) \cdot \pi(\mathbf{b} \mid \mathbf{c}).$$

Here $\pi(\mathbf{b} \mid \mathbf{K}, \mathbf{c})$ represents updated beliefs about $\mathbf{b}$, with information $\mathbf{K}$ taken into account.

The prior distribution is often chosen to facilitate calculation of the posterior, especially through the use of *conjugate priors* [3]. If the posterior distribution $\pi(\mathbf{b} \mid \mathbf{K}, \mathbf{c})$ and the prior distribution $\pi(\mathbf{b} \mid \mathbf{c})$ both belong to the same family of distributions, then π is called a conjugate prior for p.

3.2 Imprecise Prior Models

A critical feature of any Bayesian analysis is the choice of a prior distribution, which is often done by considering the choice of (hyper-)parameters of an assumed parametric prior probability distribution. This is both important if one aims at modelling prior information and if one aims to choose a prior distribution in order to reflect the absence of prior information about the parameters. In this paper we focus on the latter case, where a so-called non-informative prior has to be constructed. Many criteria for non-informativeness, and methods to determine non-informative priors, have been proposed in the literature [3, 9], with many methods applying the Bayes-Laplace postulate or the principle of insufficient reason. However, this choice meets some difficulties or problems. In particular, Walley [13] provides examples illustrating possible problems and shortcomings of the principle of insufficient reason.

An alternative way for using the Bayesian approach if one wishes not to take prior knowledge into account is through the use of a class $\mathcal{P}$ of (non-informative) prior distributions π [1, 12], which can overcome most problems that can occur when single non-informative priors are used. Such a class of priors can be considered through the lower and upper probabilities of an event A, given by

$$\underline{P}(A) = \sup\{P_\pi(A) : \pi \in \mathcal{P}\},$$
$$\overline{P}(A) = \inf\{P_\pi(A) : \pi \in \mathcal{P}\},$$

respectively. As pointed out by Walley [13], the class $\mathcal{P}$ is "not a class of reasonable priors, but a reasonable class of priors". This means that each single member of the class is not a reasonable model for prior ignorance, because no single distribution can model ignorance satisfactorily, but the whole class is a reasonable model for prior ignorance. When we have little prior information, the upper probability of a non-trivial event should be close to one and the lower probability should be close to zero. This means that the prior probability of the event may be arbitrary from 0 to 1.

Quaeghebeur and de Cooman [8] proposed a class of imprecise probability models in the framework of the so-called exponential families of probability distributions [3]. These models significantly extend a set of Bayesian imprecise models and give a possibility to develop a framework for imprecise growth models. In our approach, the set $\mathcal{P}$ is used in the imprecise Bayesian framework to take data into account with regard to parameters $\mathbf{b}$, and thus to generate the set $\mathcal{M}$ of predictive distributions with lower and upper bounds which allow us to apply Proposition 1 for maximum likelihood estimation of the parameters $\mathbf{d}$.

4 A General Scheme of the Model Construction

We now present a general scheme for our proposed method which combines imprecise Bayesian inference and maximum likelihood estimation. We present it using the setting of reliability growth models discussed earlier in this paper, but the general idea is more widely applicable. The first task is to define the sets $\mathcal{M}_1(\mathbf{d}), ..., \mathcal{M}_n(\mathbf{d})$, or their bounds, by using an appropriate imprecise Bayesian model. It consists of four steps.

1. We divide the set of parameters into two subsets. The first subset contains the parameters $\mathbf{b}$ of the assumed probability distribution p of the random variables $X_1, ..., X_n$. The second subset consists of the growth parameters $\mathbf{d}$.

2. For the assumed probability distribution p of the random variables, we choose an appropriate conjugate prior $\pi(\mathbf{b} \mid \mathbf{c})$ with parameters $\mathbf{c}$.

3. We construct the corresponding imprecise Bayesian model on the basis of results of Walley [13] or Quaeghebeur and de Cooman [8]. At that point we replace the parameters $\mathbf{c}$ by new parameters including the hyperparameter s (see [8, 13] and examples below). The produced set $\mathcal{P}$ depends on the hyperparameter s.

4. By using n observations $k_1, ..., k_n$, we write the lower and upper predictive CDFs, $\underline{F}_i(k \mid \mathbf{d}, s)$ and $\overline{F}_i(k \mid \mathbf{d}, s)$, respectively, as functions of the parameters $\mathbf{d}$ and the hyperparameter s for every debugging period. These functions form the sets $\mathcal{M}_1(\mathbf{d}), ..., \mathcal{M}_n(\mathbf{d})$. (Note that the set $\mathcal{M}_i(\mathbf{d})$ also depends on the hyperparameter s, but we omit this parameter for ease of notation.)

After completing the four steps of the first task, the sets $\mathcal{M}_1(\mathbf{d}), ..., \mathcal{M}_n(\mathbf{d})$ have been derived and these sets do not depend on the parameters $\mathbf{b}$ or $\mathbf{c}$. They depend only on the growth parameters $\mathbf{d}$, the hyperparameter s for the imprecise prior class, and the numbers i of the debugging periods. The second task is to estimate the parameters $\mathbf{d}$, this consists of two steps.

1. The likelihood function $L(\mathbf{K} \mid \mathbf{d}, s)$ is derived by applying Proposition 1.
2. Values of the parameters $\mathbf{d}$, for fixed s, should be chosen in order to maximize $L(\mathbf{K} \mid \mathbf{d}, s)$.

Note that the parameters $\mathbf{b}$ do not appear in the process, as they have been integrated out with the use of a class of priors to derive predictive distributions, and this process also implicitly replaced the parameters $\mathbf{c}$ by s. Clearly, the step to get the parameters $\mathbf{b}$ out of the model, without explicitly estimating their values, is imprecise and leads to imprecise predictive probabilities for the random variables of interest. For example, if we construct a software reliability model, then we are looking for the predictive behaviour of the analyzed software after n corrections of errors. In other words, we have to compute the probability measures of time to the $(n+1)$-th failure, in particular, the lower and upper probability distributions of time to the $(n+1)$-th failure. These bounds are totally determined by the parameters $\mathbf{d}$ and s in our approach, with s chosen to specify the class of priors, and $\mathbf{d}$ to be estimated by our proposed maximum likelihood approach in the second stage of our method.

In the following sections, we illustrate our method by considering some special cases which apply known imprecise Bayesian models and consider well-known software reliability growth models.

5 A Software Run Reliability Growth Model

A detailed description of software run reliability models is given in [4]. A run is a minimum execution unit of software. Any software execution process can be divided into a series of runs. When a run is executed, the software either passes or fails. Usually it is assumed that after observing a software failure, the software is corrected and that this action actually removes the software error that caused the failure, hence the software improves due to this action and therefore the term "reliability growth" tends to be used. There are many variations to this basic scenario in the literature, we do not address these here.

Let X be a run lifetime of software, that is, X is a discrete random variable taking the value k if the software fails during the k-th run, so after $k-1$ successful runs. The run lifetime distribution (probability mass function) is denoted by $p(k) = \Pr\{X = k\}$.

5.1 The Imprecise Beta-geometric Model

If we assume that the random variable X (run lifetime of software) is governed by the geometric distribution with parameter r and probability mass function

$$p(k \mid r) = (1-r)^{k-1} r, \ k = 1, 2, ...,$$

then the set $\mathcal{M}$ can be constructed by using an imprecise model that is similar to the beta-binomial model proposed by Walley [13]. The Beta prior distribution of the random variable r, with parameters $\alpha > 0$ and $\beta > 0$ and denoted by $\text{Beta}(\alpha, \beta)$, has probability density function

$$\pi(r) = \frac{1}{\text{B}(\alpha, \beta)} r^{\alpha-1}(1-r)^{\beta-1}, \ 0 \le r \le 1.$$

Here $\text{B}(\alpha, \beta)$ is the standard beta function.

Using the general notation introduced before in this paper for our new method, we write $\mathbf{b} = (r)$, $\mathbf{c} = (\alpha, \beta)$. If we observe k runs of software between the $(i-1)$-th and i-th software failures, and we assume that the number of such runs is geometrically distributed with parameter r, then the posterior distribution $\pi(r \mid k, \mathbf{c})$ is again a beta distribution, namely

$$\pi(r \mid k, \mathbf{c}) = \text{Beta}(\alpha + 1, \beta + k).$$

Here Bayesian analysis leads to the probability distribution of the number of events with parameters α and β. We can call this a beta-geometric model. In the beta-binomial model, Walley proposed to replace

these parameters by introducing s and γ, with $\alpha = s\gamma$ and $\beta = s - s\gamma$, and then the parameter γ is allowed to take on any value in the interval from 0 to 1, hence a set of prior distributions is created which only depends on the choice of $s > 0$, and which trivially leads to a corresponding set of posterior distributions. The hyperparameter s determines the influence of the prior distribution on posterior probabilities [13]. The beta-geometric model proposed here can be given the same imprecise Bayesian treatment, resulting in what we call the imprecise beta-geometric model. The lower and upper bounds can be obtained by minimizing and maximizing the probabilities of events over all values γ in $[0, 1]$.

5.2 The Imprecise Bayesian Growth Model

In contrast to Walley's imprecise beta-binomial model, we use a similar model by assuming that the i-th run lifetime of software X_i is governed by the geometric distribution with parameter r_i, $i = 1, ..., n$ (in place of the binomial distribution). Suppose that the probability r_i is a random variable having the beta distribution with prior parameters α and $\beta + f(i, \varphi)$. Here $f(i, \varphi)$ is a function characterizing the software reliability growth (see Section 1). In other words, the growth behaviour of the software during the debugging process is modeled by changing the parameter β of the beta distribution. In particular, if we accept $f(i, \varphi) = (i-1) \cdot \varphi$, then this model takes some features of the Littlewood-Verall model [6]. For simplicity, we will use this form of the function $f(i, \varphi)$. In this case, we get a model with three parameters, including two parameters α and β of the distribution and one parameter φ of the reliability growth. The general notation introduced above can be used by defining $\mathbf{c} = (\alpha, \beta)$ and $\mathbf{d} = (\varphi)$.

The construction of the model is based on the idea of dividing the set of parameters α, β, φ into two subsets and to consider the imprecise Bayesian model on the set $\mathcal{M}_i(\varphi)$ of CDFs bounded by lower CDFs $\underline{F}_i(k \mid \varphi, \alpha, \beta)$ and upper CDFs $\overline{F}_i(k \mid \varphi, \alpha, \beta)$, which are defined by the set of parameters $\mathbf{c} = (\alpha, \beta)$ for a fixed parameter φ, for $i = 1, ..., n$. In other words, we fix φ and construct the sets of CDFs $F_i(k)$ with bounds depending on $f(i, \varphi)$ by using the imprecise beta-geometric model.

After constructing the set $\mathcal{M}_i(\varphi)$ of CDFs $F_i(k \mid \varphi)$ through $\underline{F}_i(k \mid \varphi, \alpha, \beta)$ and $\overline{F}_i(k \mid \varphi, \alpha, \beta)$ for every $i = 1, ..., n$, and by assuming that the random variables $X_1, ..., X_n$ are independent, the likelihood function can be written and maximized by application of Proposition 1, leading to the value φ_0 that maximises this likelihood and which we consider to be an appropriate estimate of φ.

The parameters of the i-th posterior beta distribution after n observations are

$$\alpha^* = \alpha + n, \quad \beta_i^* = \beta + D_i(\varphi),$$

where

$$D_i(\varphi) = K_n + f(i, \varphi), \; K_n = \sum_{j=1}^{n} (k_j - 1).$$

Note that the posterior parameter β_i^* for the i-th posterior beta distribution is $\beta_i^* = \beta + f(i, \varphi)$. In addition, we get K_n runs of the software during n periods of observations. This implies that the posterior parameter β_i^* for the i-th period of debugging is defined by n periods of observations. This is an important feature and it is the reason for using the index i for the posterior parameter β_i^*.

It can be also seen from the above that the posterior parameters depend on $\mathbf{d}$. In the considered special case, β_i^* depends on $f(i, \varphi)$. Now we can write the predictive CDF for the i-th step of the software debugging process, after n observations, as

$$\begin{aligned} F_i(k \mid \varphi, \alpha, \beta) &= \int_0^1 (1 - (1-p)^k) \cdot \mathrm{Beta}(\alpha^*, \beta_i^*) \mathrm{d}p \\ &= 1 - \int_0^1 \frac{\Gamma(\alpha^* + \beta_i^*)}{\Gamma(\alpha^*)\Gamma(\beta_i^*)} p^{\alpha^* - 1} (1-p)^{\beta_i^* + k - 1} \mathrm{d}p \\ &= 1 - \frac{\Gamma(\alpha^* + \beta_i^*)}{\Gamma(\beta_i^*)} \frac{\Gamma(\beta_i^* + k)}{\Gamma(\alpha^* + \beta_i^* + k)} \\ &= 1 - \frac{\mathrm{B}(\alpha^* + \beta_i^*, k)}{\mathrm{B}(\beta_i^*, k)}. \end{aligned}$$

Similarly, the probability mass function is

$$\begin{aligned} p_i(k \mid \varphi, \alpha, \beta) &= \int_0^1 p(1-p)^{k-1} \cdot \frac{\Gamma(\alpha^* + \beta_i^*)}{\Gamma(\alpha^*)\Gamma(\beta_i^*)} p^{\alpha^*-1}(1-p)^{\beta_i^*-1} \mathrm{d}p \\ &= \frac{\Gamma(\alpha^* + \beta_i^*)}{\Gamma(\alpha^*)\Gamma(\beta_i^*)} \frac{\Gamma(\alpha^*+1)\Gamma(\beta_i^* + k - 1)}{\Gamma(\alpha^* + \beta_i^* + k)}. \end{aligned}$$

The expected number of successful runs after the i-th step of the software debugging process is

$$\mathbb{E}X_{i+1} = \frac{\alpha^* + \beta_i^* - 1}{\alpha^* - 1}.$$

We introduce new parameters $s > 0$ and $\gamma \in [0, 1]$ such that

$$\alpha = s\gamma, \ \beta = s - s\gamma.$$

Here the parameter s plays the same role as the hyperparameter in Walley's imprecise beta-binomial model [13]. Then the predictive CDF for the i-th step of the software debugging process can be written as

$$F_i(k \mid \varphi, \gamma, s) = 1 - \frac{\mathrm{B}(s + n + D_i(\varphi), k)}{\mathrm{B}(s - s\gamma + D_i(\varphi), k)}. \tag{4}$$

The above representation of the predictive CDF in terms of beta functions shows that it increases as γ increases in the interval $[0, 1]$, because the beta function $\mathrm{B}(x, y)$ is decreasing in x for $x > 0$. Hence, the lower bound for the function $F_i(k \mid \varphi, \gamma, s)$ is

$$\underline{F}_i(k \mid \varphi, s) = \inf_{0 \le \gamma \le 1} F_i(k \mid \varphi, \gamma, s) = F_i(k \mid \varphi, 0, s) = 1 - \frac{\mathrm{B}(s + n + D_i(\varphi), k)}{\mathrm{B}(s + D_i(\varphi), k)},$$

and the upper bound for the function $F_i(k \mid \varphi, \gamma, s)$ is

$$\overline{F}_i(k \mid \varphi, s) = \sup_{0 \le \gamma \le 1} F_i(k \mid \varphi, \gamma, s) = F_i(k \mid \varphi, 1, s) = 1 - \frac{\mathrm{B}(s + n + D_i(\varphi), k)}{\mathrm{B}(D_i(\varphi), k)}.$$

With these lower and upper CDFs, it follows from Proposition 1 that the likelihood function is of the form:

$$\begin{aligned} L^*(\mathbf{K} \mid \varphi, s) &= \prod_{i=1}^{n} \left(\overline{F}_i(k_i \mid \varphi, s) - \underline{F}_i(k_i - 1 \mid \varphi, s) \right) \\ &= \prod_{i=1}^{n} \left(\frac{\mathrm{B}(C_i(\varphi, s), \ k_i - 1)}{\mathrm{B}(s + D_i(\varphi), \ k_i - 1)} - \frac{\mathrm{B}(C_i(\varphi, s), \ k_i)}{\mathrm{B}(D_i(\varphi), \ k_i)} \right). \end{aligned}$$

Here $C_i(\varphi, s) = s + n + D_i(\varphi)$, and this likelihood function is to be maximized over $\mathcal{M}_i(\varphi)$ for given s. The optimal value φ_0 of φ can be found by numerically solving the equation $\partial \ln L^*(\mathbf{K} \mid \varphi, s)/\partial\varphi = 0$. Once we have calculated the estimate of the parameter φ, we can derive the lower and upper software run failure functions after the n-th software failure, i.e., we can compute the lower and upper CDFs of the $(n+1)$-th failure

$$\underline{F}_{n+1}(k, s) = 1 - \frac{\mathrm{B}(s + n + D_{n+1}(\varphi_0), \ k)}{\mathrm{B}(s + D_{n+1}(\varphi_0), \ k)},$$

$$\overline{F}_{n+1}(k, s) = 1 - \frac{\mathrm{B}(s + n + D_{n+1}(\varphi_0), \ k)}{\mathrm{B}(D_{n+1}(\varphi_0), \ k)}.$$

The lower and upper expected numbers of successful runs after the n-th software failure are

$$\underline{\mathbb{E}}^s X_{n+1} = \frac{n + s + K_n + f(n+1, \varphi) - 1}{s + n - 1},$$

$$\overline{\mathbb{E}}^s X_{n+1} = \frac{n + s + K_n + f(n+1, \varphi) - 1}{n - 1}.$$

It should be noted that computing the optimal value φ_0 is generally not a simple task. However, it can be simplified for some special cases of s. We will use the well-known property of beta functions $\mathrm{B}(a,b) = \mathrm{B}(b,a)$ and the following equality:

$$\mathrm{B}(a, b-1) = \frac{a+b-1}{b-1}\mathrm{B}(a,b).$$

Hence, we have

$$\frac{\mathrm{B}(C_i(\varphi,s),\ k_i-1)}{\mathrm{B}(s+D_i(\varphi),\ k_i-1)} = \frac{\mathrm{B}(C_i(\varphi,s),\ k_i)}{\mathrm{B}(s+D_i(\varphi),\ k_i)} \cdot \frac{C_i(\varphi,s)+k_i-1}{s+D_i(\varphi)+k_i-1}.$$

If $s=1$, then we can write

$$\mathrm{B}(s+D_i(\varphi),\ k_i) = \mathrm{B}(1+D_i(\varphi),\ k_i) = \frac{D_i(\varphi)}{D_i(\varphi)+k_i}\mathrm{B}(D_i(\varphi),\ k_i).$$

Hence

$$\begin{aligned}\frac{\mathrm{B}(C_i(\varphi,1),\ k_i-1)}{\mathrm{B}(1+D_i(\varphi),\ k_i-1)} &= \frac{\mathrm{B}(C_i(\varphi,1),\ k_i)}{\mathrm{B}(D_i(\varphi),\ k_i)} \cdot \frac{C_i(\varphi,1)+k_i-1}{D_i(\varphi)+k_i} \cdot \frac{D_i(\varphi)+k_i}{D_i(\varphi)} \\ &= \frac{\mathrm{B}(C_i(\varphi,1),\ k_i)}{\mathrm{B}(D_i(\varphi),\ k_i)} \cdot \frac{C_i(\varphi,1)+k_i-1}{D_i(\varphi)} \\ &= \frac{\mathrm{B}(C_i(\varphi,1),\ k_i)}{\mathrm{B}(D_i(\varphi),\ k_i)} \cdot \frac{n+D_i(\varphi)+k_i}{D_i(\varphi)}.\end{aligned}$$

By substituting the last expression into the i-th term of the likelihood function $L_i^*(\mathbf{K} \mid \varphi, s)$, we get

$$\begin{aligned}L_i^*(\mathbf{K} \mid \varphi, 1) &= \frac{\mathrm{B}(C_i(\varphi,1),\ k_i)}{\mathrm{B}(D_i(\varphi),\ k_i)} \cdot \left(\frac{n+D_i(\varphi)+k_i}{D_i(\varphi)} - 1\right) \\ &= \frac{\mathrm{B}(n+1+D_i(\varphi),\ k_i)}{\mathrm{B}(D_i(\varphi),\ k_i)} \cdot \left(\frac{n+k_i}{D_i(\varphi)}\right).\end{aligned}$$

Hence, the logarithm of the likelihood function is

$$\ln L^*(\mathbf{K} \mid \varphi, 1) = \sum_{i=1}^{n}\left(\ln\left(\frac{n+k_i}{D_i(\varphi)}\right) + \ln\left(\frac{\mathrm{B}(n+1+D_i(\varphi),\ k_i)}{\mathrm{B}(D_i(\varphi),\ k_i)}\right)\right).$$

Since k_i is integer, we can use the equality

$$\mathrm{B}(a,n) = \frac{(n-1)!}{\prod_{i=0}^{n-1}(a+i)}.$$

Then

$$\frac{\mathrm{B}(n+1+D_i(\varphi),\ k_i)}{\mathrm{B}(D_i(\varphi),\ k_i)} = \frac{\prod_{j=0}^{k_i-1}(D_i(\varphi)+j)}{\prod_{j=0}^{k_i-1}(n+1+D_i(\varphi)+j)}.$$

Hence, the logarithm of the likelihood function is

$$\begin{aligned}\ln L^*(\mathbf{K} \mid \varphi, 1) &= \sum_{i=1}^{n}\ln\left(\frac{n+k_i}{D_i(\varphi)}\right) + \sum_{i=1}^{n}\sum_{j=0}^{k_i-1}\ln\left(D_i(\varphi)+j\right) - \sum_{i=1}^{n}\sum_{j=0}^{k_i-1}\ln\left(n+1+D_i(\varphi)+j\right) \\ &= \sum_{i=1}^{n}\ln\left(n+k_i\right) + \sum_{i=1}^{n}\sum_{j=1}^{k_i-1}\ln\left(D_i(\varphi)+j\right) - \sum_{i=1}^{n}\sum_{j=0}^{k_i-1}\ln\left(n+1+D_i(\varphi)+j\right).\end{aligned}$$

Now the optimal value φ_0 of the parameter φ can be found from the following equation:

$$\frac{\partial \ln L^*(\mathbf{K} \mid \varphi, 1)}{\partial \varphi} = 0.$$

By substituting the corresponding expression for $\ln L^*(\mathbf{K} \mid \varphi, 1)$ into the above equation, we get

$$\sum_{i=1}^{n} \frac{\mathrm{d}f(i,\varphi)}{\mathrm{d}\varphi} \left(\sum_{j=1}^{k_i-1} \frac{1}{D_i(\varphi)+j} - \sum_{j=0}^{k_i-1} \frac{1}{n+1+D_i(\varphi)+j} \right) = 0. \tag{5}$$

By taking into account the form of the growth function, $(i-1)\cdot\varphi$, we obtain the equation

$$\sum_{i=1}^{n} (i-1) \left(\sum_{j=1}^{k_i-1} \frac{1}{K_n+(i-1)\varphi+j} - \sum_{j=0}^{k_i-1} \frac{1}{n+1+K_n+(i-1)\varphi+j} \right) = 0.$$

It should be noted that the special case $s=2$ allows us to obtain also a rather simple equation for computing the optimal value φ_0 of the parameter φ. The main advantage of the above approach is that we have to find only one parameter. This allows us to solve equation (5) by arbitrary numerical methods without major difficulties.

6 An Illustrative Example

Suppose that the following three numbers of runs between software failures have been recorded: $k_1 = 10$, $k_2 = 30$, $k_3 = 60$. Assuming $s = 1$, equation (5) becomes

$$\left(\sum_{j=1}^{30-1} \frac{2}{97+2\varphi+j} - \sum_{j=0}^{30-1} \frac{2}{4+97+2\varphi+j} \right) + \left(\sum_{j=1}^{60-1} \frac{3}{97+3\varphi+j} - \sum_{j=0}^{60-1} \frac{3}{4+97+3\varphi+j} \right) = 0.$$

This equation has the solution $\varphi = 25.2$. Hence we get

$$\underline{F}_4^{(1)}(m) = 1 - \frac{\mathrm{Beta}(1+3+97+3\cdot 25.2, m)}{\mathrm{Beta}(1+97+3\cdot 25.2, m)},$$

$$\overline{F}_4^{(1)}(m) = 1 - \frac{\mathrm{Beta}(1+3+97+3\cdot 25.2, m)}{\mathrm{Beta}(97+3\cdot 25.2, m)}.$$

The functions $\underline{F}_4^{(1)}(m)$ and $\overline{F}_4^{(1)}(m)$ are shown in Figure 1. The lower and upper mean runs to failure after the third step of the debugging process can be computed as

$$\underline{\mathbb{E}}^1 X_4 = \frac{3+1+97+3\cdot 25.2-1}{1+3-1} = 58.5,$$

$$\overline{\mathbb{E}}^1 X_4 = \frac{3+1+97+3\cdot 25.2-1}{3-1} = 87.8.$$

Let us compare the proposed model with a simple geometric SRM, note that this is not the geometric model proposed by Moranda [7]. We consider a model that can be regarded as a discrete analogue of the Jelinski-Moranda model [5], with the mean number of runs between the $(j-1)$-th and j-th software failures being equal to

$$\frac{1}{p_j} = \frac{1}{\phi(N-(j-1))},$$

where N is the initial (unknown) number of faults in the software, ϕ is the parameter, p_j is the parameter of the geometric distribution between the $(j-1)$-th and j-th software failures, i.e., the probability that software fails at a run after the $(j-1)$-th failure. The logarithmic likelihood function is of the form

$$\begin{aligned} \ln L(\mathbf{K}|\phi, N) &= \sum_{j=1}^{n} \left(\ln p_j + (k_j - 1) \ln(1-p_j) \right) \\ &= \sum_{j=1}^{n} \left(\ln \phi(N-j+1) + (k_j-1) \ln(1-\phi(N-j+1)) \right). \end{aligned}$$

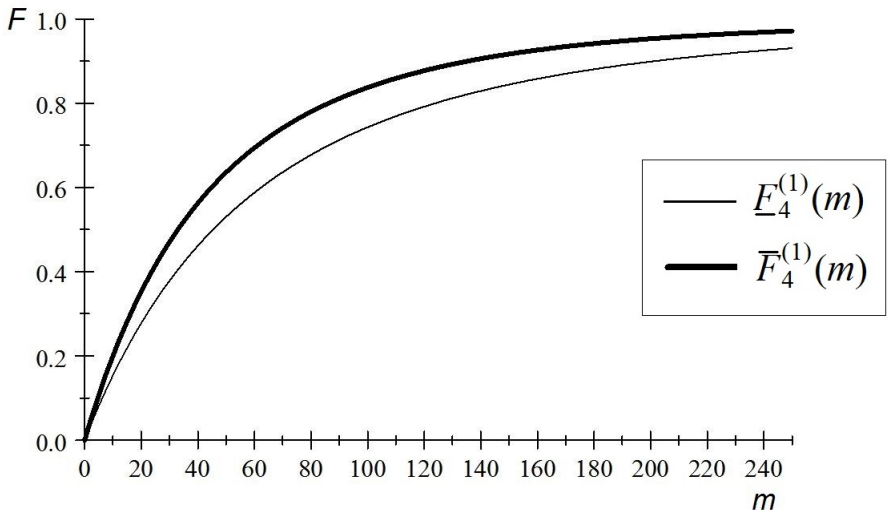

Figure 1: The functions $\underline{F}_4^{(1)}(m)$ and $\overline{F}_4^{(1)}(m)$

Hence, we get the following system of equations for computing the parameters N and ϕ:

$$\frac{\partial \ln L(\mathbf{K}|\phi,N)}{\partial N} = \sum_{j=1}^{n}\left(\frac{1}{N-j+1} + \frac{k_j-1}{(N-j+1)-1/\phi}\right) = 0,$$
$$\frac{\partial \ln L(\mathbf{K}|\phi,N)}{\partial \phi} = \frac{n}{\phi} + \sum_{j=1}^{n}\left(\frac{k_j-1}{\phi-1/(N-j+1)}\right) = 0.$$

By substituting the debugging data, we obtain

$$\frac{1}{N}+\frac{10-1}{N-1/\phi} + \frac{1}{N-1} + \frac{30-1}{N-1-1/\phi} + \frac{1}{N-2} + \frac{60-1}{N-2-1/\phi} = 0,$$
$$\frac{3}{\phi} + \frac{10-1}{\phi-1/N} + 2\frac{30-1}{\phi-1/(N-1)} + 3\frac{60-1}{\phi-1/(N-2)} = 0.$$

The above system has a set of solutions. However, only one solution satisfies the conditions

$$0 \le \phi(N-(j-1)) \le 1,\ j=1,2,3,4,$$

namely $N = 0.774$ and $\phi = -3.96 \times 10^{-2}$. Hence

$$p_{n+1} = \phi(N-n) = -3.96 \times 10^{-2} \cdot (0.774-3) = 8.82 \times 10^{-2},$$

and

$$F_{n+1}(m) = 1-(1-p_{n+1})^m = 1-(1-8.82 \times 10^{-2})^m.$$

It can be seen from Figure 2 that the predicted results based on this discrete analogue of Jelinski-Moranda model appear to be too optimistic.

7 Comparison of the Proposed SRGM with a Standard SRGM

We compare the proposed model and the discrete analogue of Jelinski-Moranda model by using a software reliability data set introduced by Jelinski and Moranda [5], it is shown in Table 1. The data consist of the number of days between the 26 failures that occurred during the production phase of specific software. The data represent the calendar time data corresponding to the continuous random times to failures. However,

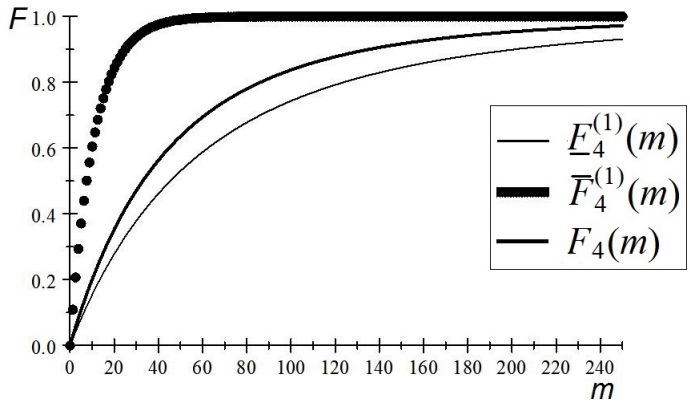

Figure 2: The functions $\underline{F}_4^{(1)}(m)$, $\overline{F}_4^{(1)}(m)$ and $F_4(m)$ (dotted)

to illustrate the new model presented in this paper, we regard the days to failure as the discrete number of successful runs between software failures.

Table 1: A software reliability data set

i	k_i	i	k_i	i	k_i
1	9	11	1	21	11
2	12	12	6	22	33
3	11	13	1	23	7
4	4	14	9	24	91
5	7	15	4	25	2
6	2	16	1	26	1
7	5	17	3		
8	8	18	3		
9	5	19	6		
10	7	20	1		

In order to evaluate the proposed model we try to make a prediction of the $(i+1)$-st discrete mean time to software failure starting from $i = 3$, we set $s = 1$. By having the mean times to failures, we can compare these with the actual times to failures given in Table 1. Moreover, we compare the results for the proposed model and for the discrete analogue of the Jelinski-Moranda model as described in Section 6.

The thick curve with the cross markers in Figure 3 is the set of real software reliability data k_i. The curve with the triangle markers is the set of predicted expectations $\mathbb{E}X_{i+1}$, $i = 3, ..., 25$, obtained by using the discrete analogue of the Jelinski-Moranda model. The two curves with the square markers are the lower and upper predicted expectations, $\underline{\mathbb{E}}X_{n+1}$ and $\overline{\mathbb{E}}X_{n+1}$, respectively.

In order to illustrate the quality of the proposed model when the amount of statistical data is small, we provide the same curves for $i = 3, ..., 16$, in Figure 4. This shows that the first three times to failure, that is $k_1 = 9$, $k_2 = 12$ and $k_3 = 11$, have substantial impact on the predicted values from the discrete analogue of the Jelinski-Moranda model. In spite of the clear deterioration of the software ($k_4 = 4$, $k_5 = 7$, $k_6 = 2$), the Jelinski-Moranda model "remembers" the first three times to failure and provides overestimated values of predicted expectations of the numbers of runs. The method proposed in this paper reacts quicker to new observations.

In order to investigate the quality of the models, we introduce the following measures of model quality, based on the deviation of the predicted expectations $\mathbb{E}X_{i+1}$ from the observations k_{i+1}: the largest deviation, denoted by $R_1(\mathbb{E}X_{i+1})$; the mean value of the deviations, denoted by $R_2(\mathbb{E}X_{i+1})$; the standard deviation of the deviations, denoted by $R_3(\mathbb{E}X_{i+1})$. These measures are

$$R_1(\mathbb{E}X_{i+1}) = \max_{i=1,...,M} |\mathbb{E}X_{i+1} - k_{i+1}|,$$

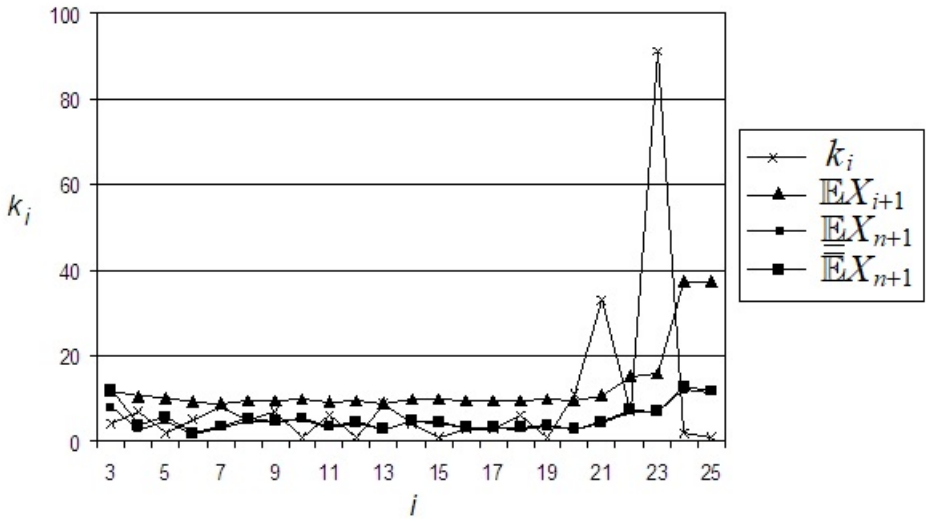

Figure 3: Predicted expected values of successful runs to failure for different models with $s = 1$

$$R_2\left(\mathbb{E}X_{i+1}\right) = M^{-1}\sum_{i=1}^{M}\left|\mathbb{E}X_{i+1} - k_{i+1}\right|,$$

$$R_3\left(\mathbb{E}X_{i+1}\right) = M^{-1}\sqrt{\sum_{i=1}^{M}\left(\mathbb{E}X_{i+1} - k_{i+1}\right)^2},$$

where M is the number of predicted times to failure.

Using the data set, we compute the corresponding measures of quality, with $s = 1$, after predicting 23 failures (from the 4-th test till the 26-th test; note that each prediction of a failure time is based only on the previous failure times). For our newly proposed model, we get

$$R_1\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right) = 84.398,\ R_2\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right) = 8.233,\ R_3\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right) = 3.988,$$

$$R_1\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right) = 84.098,\ R_2\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right) = 8.427,\ R_3\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right) = 3.988,$$

while for the discrete analogue of the Jelinski-Moranda model, we get

$$R_1\left(\mathbb{E}X_{i+1}\right) = 75.062,\ R_2\left(\mathbb{E}X_{i+1}\right) = 11.762,\ R_3\left(\mathbb{E}X_{i+1}\right) = 4.215.$$

These results show that the quality of the proposed imprecise Bayesian model appears to be better in comparison with the discrete analogue of the Jelinski-Moranda model when all the deviations are considered, but when only the maximum deviation is considered it performs a bit less. This is typical for our approach, as for small numbers of observations there is much imprecision which can lead to a large deviation for early data observations.

We compute the same measures after predicting only ten failures (from the 4-th test till the 13-th test, so just the first 10 of the 23 failures that had been predicted before). In this case, we get for the proposed model

$$R_1\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right) = 4.919,\ R_2\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right) = 3.199,\ R_3\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right) = 1.078,$$

$$R_1\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right) = 8.065,\ R_2\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right) = 3.530,\ R_3\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right) = 1.269,$$

and for the discrete analogue of the Jelinski-Moranda model

$$R_1\left(\mathbb{E}X_{i+1}\right) = 8.823,\ R_2\left(\mathbb{E}X_{i+1}\right) = 5.205,\ R_3\left(\mathbb{E}X_{i+1}\right) = 1.858.$$

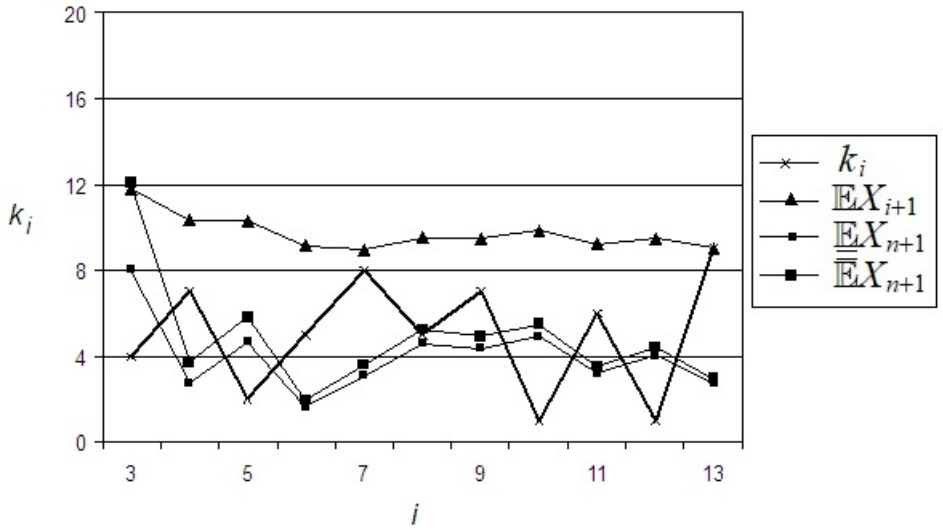

Figure 4: Predicted expected values of numbers of successful runs to failure for different models for $i = 3, ..., 16$ with $s = 1$

These values show that the predictive properties of the proposed imprecise Bayesian model are better in comparison to the discrete analogue of the Jelinski-Moranda model when the number of tests is rather small.

A remaining question is how the hyperparameter s influences the predictive quality of the model. Using the data set, we compute the quality measures with $s = 4$ for the proposed model after predicting 23 failures, these are

$$R_1\left(\underline{\mathbb{E}}^{(4)}X_{i+1}\right) = 84.885,\ R_2\left(\underline{\mathbb{E}}^{(4)}X_{i+1}\right) = 8.201,\ R_3\left(\underline{\mathbb{E}}^{(4)}X_{i+1}\right) = 4.004,$$

$$R_1\left(\overline{\mathbb{E}}^{(4)}X_{i+1}\right) = 83.774,\ R_2\left(\overline{\mathbb{E}}^{(4)}X_{i+1}\right) = 8.933,\ R_3\left(\overline{\mathbb{E}}^{(4)}X_{i+1}\right) = 4.050,$$

and after predicting 10 failures, these are

$$R_1\left(\underline{\mathbb{E}}^{(4)}X_{i+1}\right) = 5.249,\ R_2\left(\underline{\mathbb{E}}^{(4)}X_{i+1}\right) = 3.137,\ R_3\left(\underline{\mathbb{E}}^{(4)}X_{i+1}\right) = 1.045,$$

$$R_1\left(\overline{\mathbb{E}}^{(4)}X_{i+1}\right) = 18.925,\ R_2\left(\overline{\mathbb{E}}^{(4)}X_{i+1}\right) = 4.523,\ R_3\left(\overline{\mathbb{E}}^{(4)}X_{i+1}\right) = 2.165.$$

The corresponding curves of the predicted expectations are shown in Figure 5. Comparing these values with the corresponding values for $s = 1$, given above, we see that the increased imprecision for $s = 4$ leads to larger maximum deviations than for $s = 1$, which is again due to the large imprecision for the first prediction in the series, but the further results are pretty similar, hence the method learns quite quickly from the data.

8 Use of Prior Information

The proposed software reliability growth model is based on the assumption that there is no prior information about the reliability behavior of the software considered. However, we may have such information in various forms. Therefore, it is interesting to consider briefly how this information impacts on the software reliability prediction quality.

Let us consider one possible kind of expert prior information. Suppose that an expert provides the following judgement about the software reliability before the debugging process: 'The software will function without a failure on average longer than $V = 20$ runs (or days)'. By taking into account that the run lifetime of software is governed by the geometric distribution with parameter r, we can formalize the judgement as

$$\mathbb{E}X = \frac{1}{r} \geq V = 20.$$

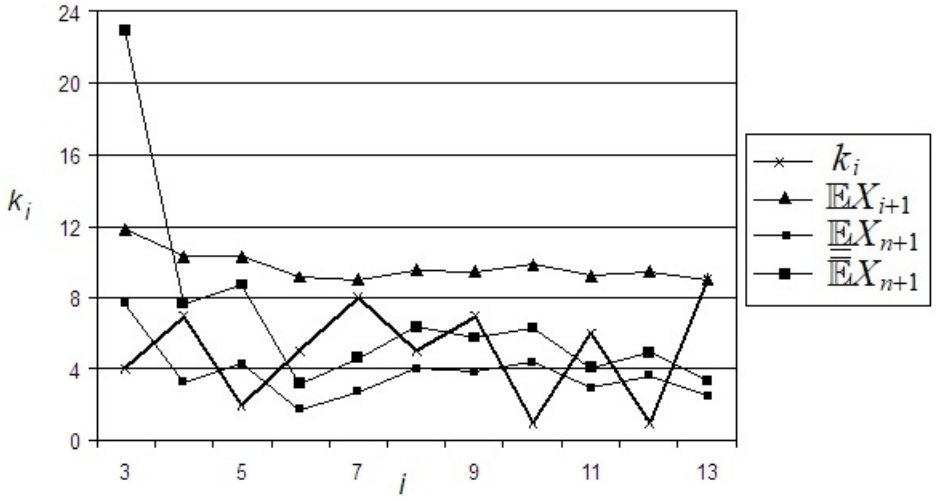

Figure 5: Predicted expected values of numbers of successful runs to failure for different models for $i = 3, ..., 16$ with $s = 4$

Hence $r \leq 1/V = 0.05$. It should be noted that we have used $r \in [0, 1]$ in the imprecise model. On the other hand, the parameter γ in the imprecise model is the expected value of the parameter r. This implies that we can now restrict the set of values of γ to the interval $[0, 1/V]$ instead of the interval $[0, 1]$.

We study how this prior information changes the measures of model quality. We denote the prior set of the parameter γ by $[\gamma_L, \gamma_U]$. Then

$$\underline{F}_i(k \mid \varphi, s) = F_i(k \mid \varphi, \gamma_L, s) = 1 - \frac{\mathrm{B}(s + n + D_i(\varphi), k)}{\mathrm{B}(s - s\gamma_L + D_i(\varphi), k)},$$

$$\overline{F}_i(k \mid \varphi, s) = F_i(k \mid \varphi, \gamma_U, s) = 1 - \frac{\mathrm{B}(s + n + D_i(\varphi), k)}{\mathrm{B}(s - s\gamma_U + D_i(\varphi), k)},$$

and

$$L^*(\mathbf{K} \mid \varphi, s) = \prod_{i=1}^{n} \left(\frac{\mathrm{B}(C_i(\varphi, s),\ k_i - 1)}{\mathrm{B}(s - s\gamma_L + D_i(\varphi),\ k_i - 1)} - \frac{\mathrm{B}(C_i(\varphi, s),\ k_i)}{\mathrm{B}(s - s\gamma_U + D_i(\varphi),\ k_i)} \right).$$

The lower and upper expected numbers of successful runs after the n-th software failure are

$$\underline{\mathbb{E}}^s X_{n+1} = \frac{n + s + K_n + f(n + 1, \varphi) - 1}{s\gamma_U + n - 1},$$

$$\overline{\mathbb{E}}^s X_{n+1} = \frac{n + s + K_n + f(n + 1, \varphi) - 1}{s\gamma_L + n - 1}.$$

Using the data set from Table 1, the corresponding measures of quality with $s = 1$ and $\gamma \in [0, 0.05]$ after predicting 23 failures (from the 4-th test till the 26-th test) by use of the newly proposed model, are

$$R_1\left(\underline{\mathbb{E}}^{(1)} X_{i+1}\right) = 84.113,\ R_2\left(\underline{\mathbb{E}}^{(1)} X_{i+1}\right) = 8.291,\ R_3\left(\underline{\mathbb{E}}^{(1)} X_{i+1}\right) = 3.981,$$

$$R_1\left(\overline{\mathbb{E}}^{(1)} X_{i+1}\right) = 84.098,\ R_2\left(\overline{\mathbb{E}}^{(1)} X_{i+1}\right) = 8.303,\ R_3\left(\overline{\mathbb{E}}^{(1)} X_{i+1}\right) = 3.982.$$

It can be seen from the numerical results that most measures of quality are approximately equal to the same measures obtained for the case $\gamma \in [0, 1]$. This results from the fact that quite a large number of data observations are used.

We compute the same measures after predicting 10 failures (from the 4-th test till 13-th test). In this case, we get the following measures for the proposed model

$$R_1\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right)=6.629,\ R_2\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right)=3.239,\ R_3\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right)=1.149,$$

$$R_1\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right)=6.895,\ R_2\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right)=3.259,\ R_3\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right)=1.116.$$

We can again see that the measures of quality are approximately equivalent to the same measures obtained for the case $\gamma\in[0,1]$.

Let us now consider $\gamma\in[0,0.01]$, corresponding to prior judgement $\mathbb{E}X\geq 100$. It can be clearly seen from Table 1 that this prior information conflicts with the data. The corresponding measures of quality with $s=1$ after predicting 23 failures are

$$R_1\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right)=84.101,\ R_2\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right)=8.646,\ R_3\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right)=4.003,$$

$$R_1\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right)=84.098,\ R_2\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right)=8.649,\ R_3\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right)=4.003.$$

These measures do not differ substantially from the same measures obtained in the previous cases. This can again be explained by the quite large number of observed statistical data. However, if we take only 10 failures, then the prediction quality is worse than in the previous case, reflected by the values

$$R_1\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right)=8.884,\ R_2\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right)=4.039,\ R_3\left(\underline{\mathbb{E}}^{(1)}X_{i+1}\right)=1.493,$$

$$R_1\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right)=8.885,\ R_2\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right)=4.045,\ R_3\left(\overline{\mathbb{E}}^{(1)}X_{i+1}\right)=1.497.$$

It follows from the above that the inclusion of incorrect, and quite precise, prior information can lead to the worse predictions in comparison to models with imprecision aimed at including little or no prior information.

9 Concluding Remarks

In this paper we have proposed a way towards development of statistical methods that combine imprecise Bayesian inference for one subset of all parameters with maximum likelihood estimation for the other parameters. The key to this approach is Proposition 1, which provides a generalization of maximum likelihood estimation for discrete variables with sets of distributions.

We presented the main idea of the new framework in this paper as an extension of the imprecise Bayesian models [8, 13] to situations where the process considered has some changeable behaviour, which we also wish to estimate using the data. In line with most reported developments in such imprecise Bayesian models, we presented it from the perspective of a non-informative set of prior distributions, but it may also be useful to apply this new method with an informative prior set of distributions. When such sets are also defined using conjugate priors in the same way as for these non-informative prior sets, that can be done in a relatively straightforward manner building on the approach presented in this paper. We chose to focus our presentation on software reliability growth models, as these typically have clear divisions of the parameters according to the different roles, for which we consider our proposed method particularly suitable.

The set $\mathcal{M}_i(\mathbf{d})$ used in this paper is the set of *all* CDFs bounded by $\underline{F}_i$ and $\overline{F}_i$. One could also consider the use of only a set of parametric distributions, all with the same parametric form as the bounding distributions. However, following this approach, maximization of the likelihood function over a set of distributions with parameters $\mathbf{c}$ would be reduced to its maximization over a set of parameters $\mathbf{c}$. In this case, we get the standard statistical model completely based on the maximum likelihood estimation, which does not differ from many well-known models of software reliability.

In this paper, we did not consider continuous random variables, but of course this case is very important. However, Proposition 1 can be extended to the continuous case, so the method can also be developed for continuous random variables $X_1,...,X_n$. In this case, the likelihood function can be written as

$$L(\mathbf{X})=\lim_{\triangle_1\to 0,\ldots,\triangle_n\to 0}\frac{\Pr\left\{x_1\leq X_1\leq x_1+\triangle_1,\cdots,x_n\leq X_n\leq x_n+\triangle_n\right\}}{\triangle_1\cdots\triangle_n},$$

and this suggests that maximum likelihood estimates for the parameters can be derived by maximizing

$$\max_{\mathcal{M}_1,\ldots,\mathcal{M}_n} L(\mathbf{X}) = \prod_{i=1}^{n} \left(\overline{F}_i(x_i) - \underline{F}_i(x_i)\right) \delta(x_i). \tag{6}$$

Here $\delta(x_i)$ is the Dirac function which has unit area concentrated in the immediate vicinity of the point x_i. $L(\mathbf{X})$ achieves its maximum by taking the probability density functions such that $\rho_i(x_i) = \left(\overline{F}_i(x_i) - \underline{F}_i(x_i)\right) \cdot\delta(x_i)$. However, whether or not condition (6) is fully correct is yet to be established, which is an important topic for further research.

Acknowledgements

Part of the research reported in this paper took place during a visit of Prof. Utkin to Durham University in November 2016, funded through a 'Research in Pairs' (Scheme 4) grant from the London Mathematical Society. We thank an anonymous reviewer for supportive comments that led to improved presentation.

References

[1] Augustin, T., Coolen, F.P.A., de Cooman, G., and M.C.M. Troffaes, *Introduction to Imprecise Probabilities*, Wiley, Chichester, 2014.

[2] Bernard, J.M., An introduction to the imprecise Dirichlet model for multinomial data, *International Journal of Approximate Reasoning*, vol.39, pp.123–150, 2005.

[3] Bernardo, J.M., and A.F.M. Smith, *Bayesian Theory*, Wiley, Chichester, 1994.

[4] Cai, K.Y., Towards a conceptual framework of software run reliability modeling, *Information Sciences*, vol.126, pp.137–163, 2000.

[5] Jelinski, Z., and P.B. Moranda, Software reliability research, *Statistical Computer Performance Evaluation*, Academic Press, New York, pp.464–484, 1972.

[6] Littlewood, B., and J. Verall, A Bayesian reliability growth model for computer software, *Applied Statistics*, vol.22, pp.332–346, 1973.

[7] Moranda, P.B., Event-altered rate models for general reliability analysis, *IEEE Transactions on Reliability*, vol.R-28, pp.376–381, 1979.

[8] Quaeghebeur, E., and G. de Cooman, Imprecise probability models for inference in exponential families, *Proceeding of the 4rd International Symposium on Imprecise Probabilities and Their Applications, ISIPTA'05*, Pittsburgh, Pennsylvania, 2005.

[9] Robert, C.P., *The Bayesian Choice*, Springer, New York, 1994.

[10] Schick, G.J., and R.W. Wolverton, An analysis of competing software reliability models, *IEEE Transactions on Software Engineering*, vol.SE-4, pp.104–120, 1978.

[11] Utkin, L.V., and F.P.A. Coolen, Imprecise reliability: an introductory overview, *Computational Intelligence in Reliability Engineering, Volume 2: New Metaheuristics, Neural and Fuzzy Techniques in Reliability*, Springer, Berlin, pp.261–306, 2007.

[12] Walley, P., *Statistical Reasoning with Imprecise Probabilities*, Chapman and Hall, London, 1991.

[13] Walley, P., Inferences from multinomial data: learning about a bag of marbles (with discussion), *Journal of the Royal Statistical Society, Series B*, vol.58, pp.3–57, 1996.

[14] Walter, G., Augustin, T., and A. Peters, Linear regression analysis under sets of conjugate priors, *Proceedings of the Fifth International Symposium on Imprecise Probabilities and Their Applications*, Prague, Czech Republic, pp.445–455, 2007.

[15] Xie, M., *Software Reliability Modeling*, World Scientific, 1991.