Software reliability prediction via two different implementations of Bayesian model averaging

Alex Sarishvili[1] and Gerrit Hanselmann[2]

[1] Fraunhofer Institute for Industrial Mathematics ITWM, 67663 Kaiserslautern, Germany, Alex.Sarishvili@itwm.fraunhofer.de
[2] Siemens AG, Corporate Technology 81739 München, Germany, Gerrit.Hanselmann@siemens.com

Abstract. The problem of predicting software reliability is strengthened by the uncertainty of selecting the right model. While Bayesian Model Averaging (BMA) provides a means to incorporate model uncertainty in the prediction, research on the influence of parameter estimation and the performance of BMA in different situations will further expedite the benefits of using BMA in reliability prediction. Accordingly, two different methods for calculating the posterior model weights, required for BMA, are implemented and benchmarked considering different data situations. The first is the Laplace method for integrals. The second is the Markov Chain Monte Carlo (MCMC) method using Gibb's and Metropolis within Gibb's sampling. For the last the explicit conditional probability density functions for grouped failure data are provided for each of the model parameters. With a number of different simulations of mixed grouped failure data we can show the robustness and superior performance of MCMC measured by mean squared error on long and short range predictions.

1 Introduction

There exists a large number of different reliability prediction models with a wide variety of underlying assumptions. The problem of selecting the single right model or combination of models is receiving considerable attention with the growing need of improved software reliability predictions [1], [2], [3], [4], [5], [6], [7]. A conceivably simple approach to integrate uncertainty about the right model in the prediction is the equally weighted linear combination (ELC) of models studied by [8]. ELC is defined by

$$\hat{f}_t^{ELC} = \frac{1}{k}\sum_{i=1}^{k} \hat{f}_i(t),$$

where $\hat{f}_i(t)$ is the prediction of $f(t)$ the number of accumulated faults found until time t, using model M_i and k is the number of models. It basically defines every prediction model as good as the other.

Instead of giving equal trust to each and every model a more sophisticated approach considers the model performance on the data to distribute trust on the different models. This can be done using Bayesian Model Averaging (BMA) that calculates posterior weights for every model and places trust amongst the models accordingly. Several realistic simulation studies comparing the performance of BMA [9, 10] showed that in general BMA has better performance. These studies were performed for a variety of situations, e.g., linear regression [11], Log-Linear models [12], logistic regression [13], and wavelets [14]. Other studies, especially regarding BMA out-of-sample performance have shown quite consistent results, namely BMA having better predictive performance than competing methods. Theoretical results on the performance of BMA are given by [15].

This paper investigates and describes two different approaches for estimating the posterior model probabilities for BMA, in the case were grouped failure data is given. One is the Laplace method for integrals. The other is the Markov Chain Monte Carlo (MCMC) method in its most general form, which allows implementation using Gibb's and Metropolis within Gibb's samplers.

Through the comparison of two different implementations of BMA on grouped and small simulated data sets we show the significance of the presented methods on the prediction performance of BMA. Furthermore we demonstrate the superiority of BMA over single prediction models and over the ELC combination technique. We decided to simulate the data and not to use standard data sets available online for the following reason: model performance estimated on only few data sets (each of them is one realization of a stochastic process) have the drawback of missing generality and in most cases have limited informative value.

For the combination four different non-homogeneous Poisson process (NHPP) models for grouped failure data with different mean value functions [16] have been used (Table 1). These models were selected because of their degree of popularity and convenience by the illustration of evaluation results. The doubt on validity of the assumption that the software reliability behaviour follows the NHPP is presented in [17].

The rest of this paper is organized as follows. Section 2 gives a short survey of software reliability growth modeling. Section 3 gives an introduction to Bayesian model averaging and describes the Laplace method and the MCMC method for calculating the posterior model weights. The simulation setup and the results are illustrated in Section 4. The conclusion and future work is given in Section 5. The models are introduced in [18], [19], [20], [21] respectively.

Table 1. Models under consideration

Model	***Mean value function***
Delayed S-Shaped **(DSS)**	$\mu(t) = a(1 - [1 + \beta t]e^{-\beta t})$
Goel-Okumoto **(GO)**	$\mu(t) = a(1 - e^{-bt})$
Goel Generalized **(GG)**	$\mu(t) = a(1 - e^{-bt^{\gamma}})$
Inflection S-Shaped **(ISS)**	$\mu(t) = \frac{a(1-e^{-bt})}{1+\beta e^{-bt}}$

2 Software Reliability Growth Models

Software reliability is defined as the probability of failure-free operation for a specified period of time under specified operating conditions [22]. Thereby, a software failure is an inconsistent behavior with respect to the specified behavior originating from a fault in the software code [23].

Software reliability modeling started in the early 70s. In contrast to hardware reliability, software reliability is concerned with design faults only. Software suffers not from deterioration nor does it fail accidentally if not in use. Design faults occur deterministically until they have been removed. Notwithstanding, software's failure behavior can be described using random models [24]. The reason is that even though the failures occur deterministically under the same conditions, their occurrence during usage may be random. A failure will no longer occur if its underlying fault has been removed. The process of finding and removing faults can be described mathematically by using software reliability growth models (SRGM) and most existing reliability models draw on this assumption of an improving reliability over time due to continuous testing and repair. An overview of these models can be found in [25], [26], [27], [28], [29]. This paper uses models of the NHPP class [16]. NHPP models have been used successfully in practical software reliability engineering. These models assume that $N(t)$, the number of observed failures up to time t, can be modeled as a NHPP, as Poisson process with a time varying intensity function. A counting process $\{N(t), t \geq 0\}$ is an NHPP if $N(t)$ has a Poisson distribution with mean value function $\mu(t) = E[N(t)]$, i. e.,

$$P(N(t) = n) = \frac{\mu(t)^n}{n!} e^{-\mu(t)},\ n = 0, 1, 2, \ldots .$$

The mean value function $\mu(t)$ is the expected cumulative number of failures in $[0, t)$. Different NHPP software reliability growth models have different forms of $\mu(t)$. (see also the table 1)

3 Bayesian Model Averaging

A software reliability growth model uses failures found during testing and fault removal to describe the failure behavior over time. Different models have different assumptions concerning the failure behavior of software. Let $M = \{M_1, ..., M_k\}$ be the k NHPP models that predict the cumulated number of failures, $f_i(t), i = 1, ..., k$, for each time t . The BMA model predicts the expected cumulated number of failures at time t, $\hat{\mu}(t)_{bma}$, by averaging over predictions of the models M_i, $i = 1 \ldots k$. Thereby the models are weighted using their posterior probabilities. This leads to the following BMA *pdf*

$$p(f(t)|\, d(t)) = \sum_{i=1}^{k} p(f_i(t)|\, M_i,\, d(t))p(M_i|\, d(t)), \tag{1}$$

where $p(f_i(t)|\, M_i,\, d(t))$ is the prediction *pdf* of $f_i(t)$ under model M_i and $p(M_i|\, d(t))$ is the posterior probability of model M_i given data $d(t)$[3]. The BMA point prediction of the cumulated number of experienced failures is

$$\hat{f}^{BMA}(t) = \sum_{i=1}^{k} \hat{f}_i(t)\, p(M_i|\, d(t)), \tag{2}$$

where the posterior probability for model M_i at time t is given by

$$p(M_i|\, d(t)) = \frac{p(d(t)|\, M_i)p(M_i)}{\sum_{i=1}^{k} p(d(t)|\, M_i)p(M_i)}, \tag{3}$$

with

$$p(d(t)|\, M_i) = \int p(d(t)|\, \theta_i, M_i)p(\theta_i|\, M_i)d\theta_i, \tag{4}$$

being the integrated likelihood of model M_i. Thereby, θ_i is the vector of parameters of model M_i, $p(\theta_i|M_i)$ is the prior density of θ_i under model M_i, $p(d(t)|\, \theta_i, M_i)$ is the likelihood, and $p(M_i)$ is the prior probability that M_i is the true model [10].

3.1 Laplace method for integrals

In this paper two methods for implementing BMA have been examined. The first is the approximation of the integral in (4) by the method of Laplace. In regular statistical models (roughly speaking, those in which the *maximum likelihood estimate* (MLE) is consistent and asymptotically normal) the best way to approximate the integral in (4) is usually using the Laplace method.

The integrated likelihood from (4) can be estimated in the following way. For simplicity the conditional information on models has been omitted from the equations. Let $g(\theta_i) = \log(p(d(t)|\, \theta_i)p(\theta_i))$ and let $\tilde{\theta}_i = arg\ \max_{\theta \in \Theta_i} g(\theta)$. After Taylor series expansion truncated at the second term the following is obtained:

$$g(\theta_i) \approx g(\tilde{\theta}_i) + 1/2(\theta_i - \tilde{\theta}_i)' g''(\tilde{\theta}_i)(\theta_i - \tilde{\theta}_i).$$

It follows

$$p(d(t)|M_i) = \int e^{(g(\theta_i))} d\theta_i = e^{g(\tilde{\theta}_i)} \int e^{(1/2(\theta_i - \tilde{\theta}_i)^T g''(\tilde{\theta}_i)(\theta_i - \tilde{\theta}_i))} d\theta_i. \tag{5}$$

By recognizing the integrand as proportional to the multivariate normal density and using the Laplace method for integrals

$$p(d(t)|M_i) = e^{g(\tilde{\theta}_i)}(2\pi)^{D_i/2}|A|^{-1/2}, \tag{6}$$

where D_i is the number of parameters in the model M_i, and $A_i = -g''(\tilde{\theta}_i)$. It can be shown that for large N which is the number of data available the

[3] Since the observed data changes with time it is denoted by the time dependent function $d(t)$.

$\tilde{\theta}_i \approx \hat{\theta}_i$, where $\hat{\theta}_i$ is the MLE, and $A_i \approx NI_i$. Thereby I_i is the expected Fisher information matrix. It is a $D_i \times D_i$ matrix whose (k,l)-th elements are given by:

$$I_{kl} = -E\left[\frac{\partial^2 \log(p(d(t)|M_i))}{\partial\theta_k \partial\theta_l}\Big|_{\theta=\hat{\theta}_i}\right].$$

Taking the logarithm of (6) leads to

$$\begin{aligned}\log(p(d(t)|M_i)) = {} & \log p(d(t)|\hat{\theta}_i) + \log p(\hat{\theta}_i) \\ & + D_i/2\log(2\pi) - D_i/2\log(N) \\ & - 1/2\log|I_i| + \mathcal{O}(N^{-1/2}). \end{aligned} \tag{7}$$

This is the basic Laplace approximation. In (7) the information matrix I_i is estimated as the negative of the Hessian of the log-likelihood at the maximum likelihood estimate. Furthermore the results can be compared with the *Bayesian information criterion* (BIC) approximation. If only the terms which are $\mathcal{O}(1)$ or less for $N \to \infty$ are retained in (7) the following can be derived:

$$\log(p(d(t)|M_i)) = \log p(d(t)|\hat{\theta}_i) - D_i/2\log(N) + \mathcal{O}(1). \tag{8}$$

This approximation is the well known BIC approximation. Its error $\mathcal{O}(1)$ does not vanish for an infinite amount of data, but because the other terms on the right hand side of (8) tend to infinity with the number of data, they will eventually dominate the error term. This is the case when software testing has progressed sufficiently and much failure data is available.

One choice of the prior probability distribution function of the parameters in (7) is the multivariate normal distribution with mean $\hat{\theta}_i$ and the variance equal to the inverse of the Fisher information matrix. This seems to be a reasonable representation of the common situation where there is only little prior information. Under this prior using (7) the posterior approximation is:

$$\log(p(d(t)|M_i)) = \log p(d(t)|\hat{\theta}_i) - D_i/2\log(N) + \mathcal{O}(N^{-1/2}). \tag{9}$$

Thus under this prior the error is $\mathcal{O}(N^{-1/2})$ which is much smaller for moderate to large sample sizes and which tends to zero as N tends to infinity. (9) was pointed out by [30]. This is a variant of the so called non-informative priors. Non-informative priors are useful in case when the analyst has no relevant experience to specify a prior, and that the subjective elicitation in multi-parameter problems is impossible. The Laplace method with normal prior (9) is used for the approximation of the posterior probabilities.

3.2 Markov Chain Monte Carlo method

Another way of calculating the marginal likelihoods which are needed for estimating the posterior weights of the models is MCMC. MCMC generates samples from the joint *pdf* of the model parameters. In this paper the Gibb's sampler is used for the MCMC implementation. The transition distribution of this Markov chain is the product of several conditional probability densities. The stationary distribution of the chain is the desired posterior distribution [31]. After the samples from the parameter joint *pdf* for any model i are generated, the integral (4) can be approximated by the sum

$$p(d(t)|\, M_i) = \sum_{j=1}^{T} p(d(t)|\, \theta_i^{(j)}, M_i) p(\theta_i^{(j)}|\, M_i), \tag{10}$$

where T is the number of Gibb's sampler iterations. Below the parameter conditional probability density functions are given, which are needed for the Gibb's sampler implementation. A similar MCMC implementation was ddescribed by [32] but for non-grouped data. Because the likelihood function for grouped data is different then for the non-grouped data the conditional densities needed for Gibb's implementation are different, as well.

The likelihood function of different NHPP models for interval failure count data is

$$p(d(t)|\theta_i, M_i) = \prod_{i=1}^{t} \frac{(\mu(i) - \mu(i-1))^{d_i}}{d_i!} e^{-\mu(t)}. \tag{11}$$

It is necessary to make an assumption about the prior distribution of the model parameters. It is convenient to choose the Gamma distribution as prior, for it supports the positivity of the parameters and is quite versatile to reflect densities with increasing or decreasing failures rates.

Under this assumption the posterior density, for instance for the delayed s-shaped model is

$$p(a, \beta | d(t)) \propto p(d(t)|a, \beta, DSS) a^{\alpha_1 - 1} e^{-a\alpha_2} \beta^{\beta_1 - 1} e^{-\beta\beta_2}.$$

Inserting the corresponding mean value function into (11) yields $p(d(t)|\, \alpha, \beta, DSS)$. Subsections 3.2.1 to 3.2.4 describe the gamma prior distributions of the parameters of the considered models.

Since not all conditional densities have convenient forms the Metropolis-within-Gibbs algorithm is used to approximate the joint *pdf* of the model parameters. One way to avoid this computationally intensive Metropolis-within-Gibbs sampling is data augmentation [31]. This is however not considered in this paper.

The following subsections describe the conditional densities for the different mean value functions of the NHPP models. These conditional densities can be used in the Gibbs sampler to generate the desired joint parameter probability distributions and therefore the corresponding posterior *pdf*'s for each model.

3.2.1 Conditional densities for DSS model parameters

For the DSS-model with Gamma a-priori distributed parameters
$a \sim \Gamma(\alpha_1, \alpha_2)$ and $\beta \sim \Gamma(\beta_1, \beta_2)$, the following conditional densities are sampled

$$p(\beta | a, d(t)) \propto e^{\left(-\beta \sum_{i=1}^{t} i d(i) - \bar{\mu}(t) - \beta_2 \beta\right)} \beta^{\beta_1 - 1} \prod_{i=1}^{t} A^{d(i)},$$

where $A = -(1+\beta i)+e^{\beta}(1+\beta i-\beta)$ and $\bar{\mu}(t) = -a(1+\beta t)e^{-\beta t}$ is the second term of the expectation function of the DSS model. The conditional density function for a is

$$a|\beta, d(t) \sim \Gamma\left(\sum_{i=1}^{t} d(i) + \alpha_1, 1-(1+\beta t)e^{-\beta t} + \alpha_2\right).$$

3.2.2 Conditional densities for GO model parameters

Considering the GO-model the conditional densities for the parameters a and b are

$$a|b, d(t) \sim \Gamma\left(\sum_{i=1}^{t} d(i) + a_1, 1-e^{-bt} + a_2\right)$$

and

$$p(b|a, d(t)) \propto e^{\left(-b\sum_{i=1}^{t} id(i)-\bar{\mu}(t)-b_2 b\right)}A,$$

where

$$A = (e^b-1)^{\sum_{i=1}^{t} d(i)} b^{b_1-1},$$

and $\bar{\mu}(t) = -ae^{-bt}$ is the second term of the expectation function of the GO model.

3.2.3 Conditional densities for GG model parameters

For the GG-model with a Gamma prior distribution the conditional densities are: For the parameter $a \sim \Gamma(a_1, a_2)$:

$$a|b, \gamma, d(t) \sim \Gamma\left(\sum_{i=1}^{t} d(i) + a_1, 1-e^{-bt^{\gamma}} + a_2\right).$$

For the parameter $b \sim \Gamma(b_1, b_2)$:

$$p(b|a, \gamma, d(t)) \propto Ae^{-\bar{\mu}(t)-b_2 b} b^{b_1-1},$$

where

$$A = \prod_{i=1}^{t}\left(-\frac{1}{e^{bi^{\gamma}}} + \frac{1}{e^{b(i-1)^{\gamma}}}\right)^{d(i)}.$$

For the parameter $\gamma \sim \Gamma(\gamma_1, \gamma_2)$:

$$p(\gamma|a, b, d(t)) \propto Ae^{-\bar{\mu}(t)-\gamma_2\gamma}\gamma^{\gamma_1-1},$$

where $\bar{\mu}(t) = -ae^{t^{\gamma}}$ is the second term of the expectation function of the GG model.

3.2.4 Conditional densities for ISS model parameters

For the parameter $a \sim \Gamma(a_1, a_2)$ of the ISS model the conditional density is

$$a|b, \beta, d(t) \sim \Gamma\left(\sum_{i=1}^{t} d(i) + a_1, \frac{1 - e^{-bt}}{1 + \beta e^{-bt}} + a_2\right).$$

For parameters $b \sim \Gamma(b_1, b_2)$ and $\beta \sim \Gamma(\beta_1, \beta_2)$ the conditional density is

$$p(\beta|a, b, d(t)) \propto (\beta + 1)^{\sum_{i=1}^{t} d(i)} A \beta^{\beta_1 - 1} e^{-\beta_2 \beta}$$

and

$$p(b|a, \beta, d(t)) \propto e^{b\left(\sum_{i=1}^{t} id(i) - b_2\right)} (e^b - 1)^{\sum_{i=1}^{t} d(i)} A b^{b_1 - 1}$$

respectively. Thereby,

$$A = \prod_{i=1}^{t} \left((e^{bi} + \beta)(e^{bi} + \beta e^b)\right)^{-d(i)} e^{-\mu(t)},$$

where $\mu(t)$ is the expectation function of the ISS model.

4 Evaluation

This paper examines two methods to calculate the posterior weights for the model combination using BMA. The evaluation compares the prediction performance of different combinations:

– BMA using MCMC
– BMA using Laplace
– ELC.

Besides comparing the prediction performance of the combinations, BMA using MCMC is also compared to the performance of individuals models. The comparison of model performances is done by measuring the mean squared error on long and short range software reliability predictions.

The experimental procedure begins with simulation of mixed NHPP realizations via thinning algorithm, which is described in the next section. The obtained data are then used for Gibb's sampler (Section 3.2) and Laplace procedure(Section 3.1) to estimate the posterior *pdf*s of selected NHPP models. Finally the performance comparison is made in section 4.4. The following sections present the details of the evaluation.

4.1 Simulation

To achieve this evaluation different data sets have to be simulated. In detail 100 mixed NHPP processes with uniformly distributed random mixing coefficients, and with random model parameters have been simulated. In [33] was shown that failure data can be much better described using several models instead of only one. Therefore simulating mixed NHPP processes is more realistic and used in this paper. The following distributions and parameters were used in the simulation

- for DSS Model $a \sim \mathcal{U}_{[50,125]}$ and $\beta \sim \mathcal{U}_{[0.05,0.15]}$
- for GO Model $a \sim \mathcal{U}_{[50,125]}$ and $b \sim \mathcal{U}_{[0.05,0.15]}$
- for ISS Model $a \sim \mathcal{U}_{[50,125]}$, $b \sim \mathcal{U}_{[0.05,0.15]}$ and $\beta \sim \mathcal{U}_{[2.5,3.5]}$
- for GG Model $a \sim \mathcal{U}_{[50,125]}$, $b \sim \mathcal{U}_{[0.05,0.15]}$ and $\gamma \sim \mathcal{U}_{[2.5,3.5]}$.

For the simulation of a single NHPP model an approach called *thinning* or *rejection method* [34] was used. It is based on the following observation. If there exists a constant $\bar{\lambda}$ such that $\lambda(t) \leq \bar{\lambda}$ for all t. Let $T_1^*, T_2^*, \ldots$ be the successive arrival times of a homogeneous Poisson process with intensity λ, and we accept the ith arrival time with probability $\lambda(T_i^*)/\bar{\lambda}$, then the sequence $T_1, T_2, \ldots$ of the accepted arrival times are the arrival times of a NHPP with rate function $\lambda(t)$ [35].

The generated data sets were divided in training and validation parts. The validation parts were used to calculate the predictive performance. All algorithms were implemented in Matlab environment.

4.2 Performance measure

The performance measure used for the evaluation is the standard measure mean squared error (MSE). The MSE for a specific model i can be expressed as

$$MSE_i = \frac{1}{N} \sum_{t=1}^{N} \left(\hat{f}_i(t) - f_i(t) \right)^2,$$

with $\hat{f}_i(t)$ the predicted number of failures of model i at time t and $f_i(t)$ is the actual number of observed (simulated) failures at time t.

The Bayes MSE is estimated by means of the Monte Carlo integration. Let $\theta_i^{(k,m)}$ be the variates of the parameters of model i drawn in the k-th replication and m-th iteration of the Gibbs sampler and let $f_i(\theta_i^{(k,m)}, t)$ be the model i output if we use the parameter vector $\theta_i^{(k,m)}$ estimated with the data till time t, then the MSE_i is calculated as follows:

$$\tilde{f}_i(t) = \frac{2}{KM} \sum_{k=1}^{K} \sum_{m=\frac{M}{2}+1}^{M} f_i(\theta_i^{(k,m)}, t), \tag{12}$$

$$MSE_i = \frac{1}{N} \sum_{t=1}^{N} \left(\tilde{f}_i(t) - f_i(t) \right)^2$$

where $\tilde{f}_i(t)$ is the Bayesian MCMC estimation of the accumulated number of failures by model i with the data from the time interval $[1, t]$, K is the number of replications and M is the number of iterations of the Gibbs sampler and N is the number of data.

4.3 Evaluation parameters

The a-priori densities of the parameters were chosen from the Gamma distribution function with following parameters:

– for DSS Model $a \sim \Gamma(1, 0.001)$ and $\beta \sim \Gamma(1, 0.001)$
– for GO Model $a \sim \Gamma(1, 0.001)$ and $b \sim \Gamma(1, 0.001)$
– for ISS Model $a \sim \Gamma(1, 0.001)$, $b \sim \Gamma(1, 0.001)$ and $\beta \sim \Gamma(2, 1)$
– for GG Model $a \sim \Gamma(1, 0.001)$, $b \sim \Gamma(1, 0.001)$ and $\gamma \sim \Gamma(1, 0.001)$

The number of MCMC iterations was $M = 1000$ and number of replications $K = 25$, see formula 12.

The BMA point estimate is given as follows:

– For BMA with MCMC: Inserting the outcome of (10) together with the equal model a-priori probabilities $P(M_i) = 1/4$ into (3), results in the BMA weighting factors. Inserting the product of this factors and the outcome of (12), into 2) results in the BMA MCMC point estimate of the system output.
– For BMA with Laplace: Inserting the exponential of the outcome of (9) together with the equal model a-priori probabilities $P(M_i) = 1/4$ into (3), results in the BMA weighting factors. Inserting the product of this factors and the ML estimates of the model outputs into (2) results in the BMA Laplace point estimate of the system output.

4.4 Evaluation results

4.4.1 Comparing model performance of MCMC BMA to single models

Figure 1 shows the performance comparison of the MCMC BMA Model vs. single models using 75% of the simulated data for the model parameter estimation and the remaining 25% of data for the model validation. In detail the Figure shows on the x-axis the MSE of BMA MCMC and on the y-axis the MSE of the respective single model. Therefore every point above the equal performance border line indicates that the performance of MCMC BMA was better than that of the respective single model. Figure 2 shows the same comparison but with only 50% of the data for the model parameter estimation and the remaining data for the validation.

In Figure 1 the DSS model MSE is near the equal performance border line for an $MSE < 16$, that means for simulated data which had a high weighting coefficient on DSS Model in the thinning algorithm. However the more interesting case of a longer term prediction, Figure 2 shows that the BMA MCMC model outperforms all single models by far.

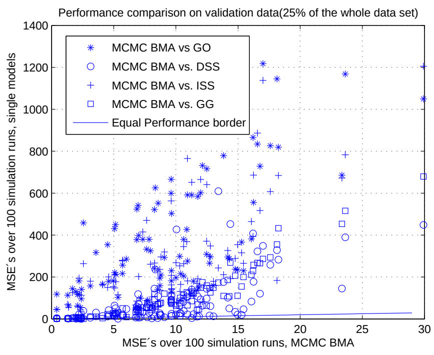

Fig. 1. Predictive performance comparison BMA MCMC vs single models, 25% of the data were used for the validation

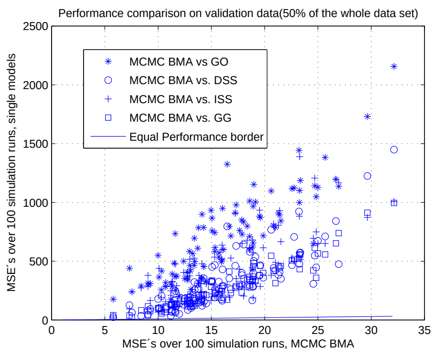

Fig. 2. Predictive performance comparison BMA MCMC vs single models, 50% of the data were used for the validation

4.4.2 Comparing model performance of MCMC BMA to Laplace BMA and ELC

The Figures 3 and 4 show the comparison results considering different combinations. Considering long term prediction BMA MCMC has better performance than ELC or BMA Laplace in almost every of the 100 simulation runs (Figure 3). However if 75% of the data were used for model fitting ELC had in 20% and BMA Laplace in 23% of the 100 simulation runs similar or better predictive performance than BMA MCMC (Figure 4). The BMA Laplace model was in 53 of 100 simulation cases better than the ELC on the small data set(50% of the data used for the parameter estimation) see the figure 3). For the bigger data set (75% of the data used for the parameter estimation) BMA Laplace had a smaller MSE than ELC in 64 out of the 100 simulation runs. This trend shows the better approximative results of the Laplace method for large data sets. Comparing the prediction performance of BMA MCMC and BMA Laplace revealed that

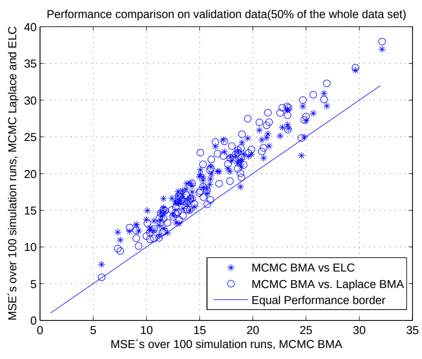

Fig. 3. Predictive performance comparison BMA MCMC vs combining models, 50% of the data were used for the validation

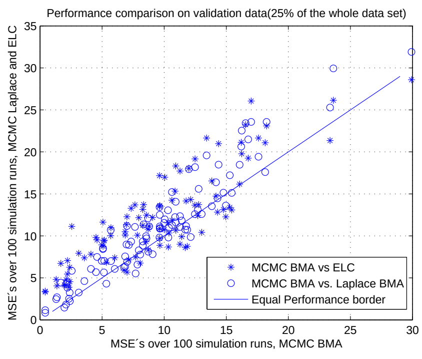

Fig. 4. Predictive performance comparison BMA MCMC vs combining models, 25% of the data were used for the validation

the BMA MCMC is clearly better suited for small data sets. If only 50% of the data was used as training data set BMA MCMC had a much better prediction performance than BMA Laplace. With more and more data the BMA Laplace is improving. This trend can be observed when for instance 75% of the data was used as training data set. In this case already 23% of the predictions had similar or better performance than BMA MCMC.

5 Conclusion

A review of the relevant literature reveals great agreement that there is no single model that can be used for all cases. This paper addressed this issue and studied two ways of implementing Bayesian Model Averaging for Non-Homogeneous Poisson Process models for grouped failure data. It could be shown that BMA had better prediction performance than the single models. Also it could be shown that BMA has better prediction performance than other simpler combination of approaches like ELC.

Considering the two ways of implementing BMA it was shown that the MCMC approach is by far better than the Laplace method if the data set used for parameter estimation is small. Laplace should not be used when the number of detected failures is small and therefore the ratio between the error term and the other terms on the right hand side of (9) is high. On the other hand, the Laplace approximation does not require complicated computational procedures like Metropolis within Gibb's sampler. However since the multiplicative normal distribution does not account for skewness, the accuracy of the approximation is low in many cases.

The MCMC exploration of the support of a-posteriori probability density function was pretty fast. The problem was the Metropolis within Gibbs variant of the algorithm, which was time consuming. One possibility to avoid these difficulties is the introduction of latent random variables for augmentation of Gibbs conditional densities. The number of iterations in the Gibbs sampler was determined by monitoring of convergence of averages [31]. We showed the high predictive performance of the MCMC BMA in comparison with Laplace BMA and ELC combination methods in the case if one wants to make long range predictions in the software testing processes for moderate sized software projects.

In this paper we concentrated on four NHPP based SRG Models. An extension of the presented implementation techniques of BMA to more models is possible. For larger model spaces techniques for optimal model space reduction like "Occam's window" or optimal model space exploration like MC^3 [36] or reversible jump MCMC could be of interest.

References

1. Abdel-Ghaly, A.A., Chan, P., Littlewood, B.: Evaluation of competing software reliability predictions. IEEE Transactions on Software Engineering **12(9)** (1986) 950–967

2. Nikora, A.P., Lyu, M.R.: Software Reliability Measurement Experience. In: Handbook of Software Reliability Engineering. McGraw–Hill (1995) 255–302
3. Brocklehurst, S., Littlewood, B.: Techniques for Prediction Analysis and Recalibration. In: Handbook of Software Reliability Engineering. McGraw–Hill (1995) 119–166
4. Almering, V., van Genuchten, M., Cloudt, G., Sonnemans, P.J.M.: Using software reliability growth models in practice. Volume 11-12. (2007) 82–88
5. Singpurwalla, N.D., Wilson, S.P.: Statistical Methods in Software Engineering: Reliability and Risk. Springer (1999)
6. Ravishanker, N., Liu, Z., Ray, B.K.: Nhpp models with markov switsching for software reliability. Computational statistics and data analysis **52** (2008) 3988–3999
7. Dharmasena, L.S., Zeephongsekul, P.: Fitting software reliability growth curves using nonparametric regression methods. Statistical Methodology **7** (2010) 109–120
8. Lyu, M.R., Nikora, A.: Applying reliability models more effectively. IEEE Software **9(4)** (1992) 43–52
9. Raftery, A.E., Madigan, D., Volinsky, C.T.: Accounting for model uncertainty in survival analysis improves predictive performance. Technical report, Department of Statistics, GN-22, University of Washington (1994)
10. Hoeting, J.A., Madigan, D., Raftery, A.E., Volinsky, C.T.: Bayesian model averaging: A tutorial. Statistical Science **14** (1999) 382–417
11. George, E.I., McCulloch, R.E.: Variable selection via gibbs sampling. Journal of the American Statistical Association **14(Series B)** (1993) 107–114
12. Clyde, M.A.: Bayesian model averaging and model search strategies. Bayesian Statistics **6** (1999) 157–185
13. Viallefont, V., Raftery, A.E., Richardson, S.: Variable selection and Bayesian model averaging in case-control studies. Statistics in Medicine **20** (2001) 3215–3230
14. Clyde, M.A., George, E.I.: Flexible empirical Bayes estimation for wavelets. Journal of Royal Statistical Society **62(Series B)** (2000) 681–698
15. Raftery, A.E., Zheng, Y.: Long run performance of Bayesian model averaging. Technical report, Department of Statistics University of Washington (2003)
16. Huang, C.Y., Lyu, M.R., Kuo, S.Y.: A unified scheme of some nonhomogeneous Poisson process models for software reliability estimation. IEEE Transactions on Software Engineering **29(3)** (2003) 261–269
17. Cai, K.Y., Hu, D.B., Bai, C.G., Hu, H., Jing, T.: Does software reliability growth behavior follow a non-homogeneous Poisson process. Information and Software Technology **50** (2008) 1232–1247
18. Yamada, S., Ohba, M., Osaki, S.: S-shaped reliability growth modeling for software error detection. IEEE Transactions on Reliabiity **R-32** (1983) 475–478
19. Goel, A.L., Okumoto, K.: Time-dependent error-detection rate model for software reliability and other performance measures. IEEE Transactions on Reliability **28(3)** (1979) 206–211
20. Goel, A.L.: Software reliability models: Assumptions, limitations, and applicability. IEEE Transactions on Software Engineering **11(12)** (1985) 1411–1423
21. Ohba, M.: Inflection S-shaped softwrae reliability growth models. In: Stochastic Models in Reliability Theory. Springer (1984) 144–162
22. ANSI/IEEE: Standard Glossary of Software Engineering Terminology. Std-729-1991 edn. (1991)
23. IEEE: IEEE Standard Glossary of Software Engineering Terminology. Institute of Electrical & Electronics Enginee (2005)

24. Lyu, M.R.: Software Reliabiliy Theory. In: Encyclopedia of Software Engineering. John Wiley & Sons (2002)
25. Xie, M.: Software Reliability Modeling. World Scientific Publishing (1991)
26. Lyu, M.R.: Handbook of Software Reliability Engineering. McGraw–Hill (1995)
27. Musa, J.D., Iannino, A., Okumoto, K.: Software Reliability Measurement Prediction Application. McGraw–Hill (1987)
28. Lakey, P.B., Neufelder, A.M.: System and Software Reliability Assurance Notebook. Produced for Rome Laboratory by SoftRel (1997)
29. Pham, H.: Software Reliability. Springer (2000)
30. Kass, R.E., Wasserman, L.: Improving the laplace approximation using posterior simulation. Technical report, Carnegie Mellon University, Dept. of Statistics (1992)
31. Casella, G. In: Monte Carlo Statistical Methods, Springer texts in statistics. Springer (1999)
32. Kuo, L., Yang, T.Y.: Bayesian computation for nonhomogeneous Poisson processes in software reliability. Journal of the American Statistical Association **91/434** (1996) 763–773
33. Gokhale, S., Lyu, M., Trivedi, K.: Software reliability analysis incorporating fault detection and debugging activities. (1998) 202
34. Grandell, J.: Aspects of risk theory. Springer (1991)
35. Burnecki, K., Härdle, W., Weron, R.: An introduction to simulation of risk processes. Technical report, Hugo Steinhaus Center for Stochastic Methods (2003)
36. Madigan, D., York, J.: Bayesian graphical models for diskrete data. International Statistical Review **63** (1995) 215–232