

# Performance Analysis of Software Reliability Models using Matrix Method

RajPal Garg<sup>1</sup>, Kapil Sharma<sup>2</sup>, *Member IEEE*, Rajive Kumar<sup>3</sup>, R. K. Garg<sup>4</sup>

**Abstract**—This paper presents a computational methodology based on matrix operations for a computer based solution to the problem of performance analysis of software reliability models (SRMs). A set of seven comparison criteria have been formulated to rank various non-homogenous Poisson process software reliability models proposed during the past 30 years to estimate software reliability measures such as the number of remaining faults, software failure rate, and software reliability. Selection of optimal SRM for use in a particular case has been an area of interest for researchers in the field of software reliability. Tools and techniques for software reliability model selection found in the literature cannot be used with high level of confidence as they use a limited number of model selection criteria. A real data set of middle size software project from published papers has been used for demonstration of matrix method. The result of this study will be a ranking of SRMs based on the Permanent value of the criteria matrix formed for each model based on the comparison criteria. The software reliability model with highest value of the Permanent is ranked at number – 1 and so on.

**Keywords**—Matrix method, Model ranking, Model selection, Model selection criteria, Software reliability models.

## ACRONYM

NHPP	Non-homogeneous Poisson process
SRM	Software reliability model
MLE	Maximum likelihood estimation

## NOTATION

$m(t)$	Mean value function
$\lambda(t)$	Intensity function
$m_i$	Total number of failures observed at time $t_i$ according to the actual data
$\hat{m}(t_i)$	Expected number of failures at time $t_i$ estimated by a Model

RajPal Garg<sup>2</sup> is a research scholar in Computer Science and Engineering with Singhania University, Rajasthan, India. rpg\_ashu@rediffmail.com.  
 Kapil Sharma<sup>2</sup> is working as senior lecturer in G.P.M. College of Engineering, 245 Budhpur, Delhi-36, India. (e-mail:kapil@ieee.org)  
 Rajive Kumar<sup>3</sup> and R. K. Garg<sup>4</sup> are now professors with the Department of Maths and Mechanical engineering, respectively at Deenbandhu Chhotu Ram University of Science and Technology, Murthal, Haryana, India (e-mail: romeshkgarg@gmail.com, Phone: 919416105066, Fax: 911302484147).

## I. INTRODUCTION

THE software development process becomes increasingly time-consuming and expensive due to the complexity of software systems. In the mean time, the need for the highly reliable software system is ever increasing. How to enhance the reliability of the software systems and reduce the cost to an acceptable level becomes the main focus of the software industry. Methods of applying reliability and cost models to the software development practice are highly desired [1].

The effects of this process, by which it is hoped software is made more reliable, can be modeled through the use of Software Reliability Models, hereafter referred to as SRMs. Ideally, these models provide a means of characterizing the development process and enable software reliability practitioners to make predictions about the expected future reliability of software under development. Such techniques allow managers to accurately allocate time, money, and human resources to a project, and assess when a piece of software has reached a point where it can be released with some level of confidence in its reliability. Various models have been proposed to characterize software reliability and its dependence on a number of factors related to the product or the software process, some of these are presented in Xie [2], Goel and Okumoto [3], Lyu [4], and Musa and Okumoto [5].

With the rapid development of computer technology, wide use of computers to control all military and civil systems and increasing demand of high quality software products, software reliability has become the primary concern and it is must to evaluate software reliability accurately and carefully to determine the system reliability. The software reliability models also called as counting models, those represent cumulative number of failures and have time dependent failure intensity function are considered the best ways to measure software reliability. Many mathematical models called SRMs (software reliability models) have been developed. The techniques for achieving and demonstrating high reliability are available, through the various reliability models. But how do we use them and do we need this diversity of models? To answer this question we have to compare models and study how they relate to each other. First of all we have to study the models and select those are best suited for our environment, techniques and applications according to the assumptions made by the models. Having done this we are probably left with either a number of models or none at all. If we are left with none, we have to take an approach similar to the one

presented in Wohlin [6], i.e. to develop a model which is tailored to the environment and the techniques used. Let us suppose that we still have at least two possible models.

In this paper we proposed a matrix method for performance analysis of sixteen different NHPP software reliability models based on a set of seven contributing model selection criteria. The remainder of the paper is organized as follows: Section II reveals the existing literature for different types of software reliability models, models' selection criteria and selection methodologies etc. In section III, parameter estimation technique and various comparison criteria are identified. The evaluation of the SRM based on individual criteria for a data set comprising three releases is given in section IV. The matrix method and model demonstration with the help of case study to develop a procedure mingling various comparison criteria for comprehensive ranking of the alternative NHPP software reliability models are described in Section V. Finally, the conclusions are given in Section VI.

## II. LITERATURE REVIEW

Early work in the field of software reliability focused around proposing new models. Over the past 30 years, many SRGMs have been proposed for estimation of reliability growth of products during software development process [2, 4, 7-16]. Each model could be shown to work well with a unique data set, but no model appeared to do well on all data sets. Many researchers like Musa et al. [7] have shown that some families of models have, in general, certain characteristics that are considered better than others; for example, that the geometric family of models tends to have better predictive quality than other models. These and other attempts, Schick and Wolverton [17] and Sukert [18], to compare different models have led to an evolution from proposing a new model to proposing techniques for finding the best model for each individual application from among the existing models. Ideally we would like to be able to select, before starting, which model we should use.

This has proven to be a very difficult, almost impossible task. Brocklehurst et al. [19] suggest that it is the very nature of software failures that has made the model selection process in general a difficult task. Software failures are caused by hidden design flaws and not by the psychological sciences that will someday show us how to select the model beforehand. Today we must evaluate different models, compare them, and choose the best.

Goel and Okumoto [3] published a paper describing a non-homogeneous Poisson process model from the finite exponential class of models. This was one of the first non-homogeneous Poisson process models proposed. Goel and Okumoto validated this model by showing that it predicted well on a unique data set.

Goel [20] and others started describing processes for which each model would be tested to see how well the model fits the data and predicts the future events. The assertion was that different models predict well only on certain data sets and that by comparing the predictive quality of different models, it is possible to select the best one for a given application.

Abdel-Ghaly et al. [21] compared the predictive quality of 10 models using five different methods of comparison. They showed that different methods of model selection result in different models being chosen. Also some of their methods were rather subjective as to which model was better than others. Clearly a simple and objective method to select models is needed.

Khoshgoftaar and Woodcock [22] proposed a method to select a reliability model among various alternatives using the log-likelihood function. They apply the method to the failure logs of a project. The method selected an S-shaped model as the most appropriate one.

## III. PARAMETER ESTIMATION AND COMPARISON CRITERIA

Since computers are being used increasingly to monitor and control both safety critical and civilian systems, there is a great demand for high-quality software products. Reliability is a primary concern for both software developers and software users. Research activities in software reliability engineering have been conducted and a number of NHPP software reliability models have been proposed to assess the reliability of software. In fact, software reliability models based on the NHPP have been quite successful tools in practical software reliability engineering. These models consider the debugging process as a counting process characterized by its mean value function. Software reliability can be estimated once the mean value function is determined. Model parameters are usually estimated using either the maximum likelihood method or regression. Different models have been built upon different assumptions. The sixteen NHPP software reliability models, as mentioned in Table 1, are considered for comparison and ranking in this research paper.

### A. Parameter Estimation

Once the analytic expression for the mean value function is derived, it is required to estimate the parameters in the mean value function, which is usually carried out by using Maximum likelihood Estimation technique.

### B. Comparison Criteria

A model can be judged according to its ability to reproduce the observed behavior of the software, and to predict the future behavior of the software from the observed failure data. A detailed study of the available literature reveals that the following twelve quantitative criteria are being used for comparison of software reliability models for different data sets.

1. The Bias is defined as [23], [24]:

$$Bias = \frac{\sum_{i=1}^k (\hat{m}(t_i) - m_i)}{k} \quad (1)$$

It is the sum of the difference between the estimated curve and the actual data.

TABLE 1  
SUMMARY OF THE SOFTWARE RELIABILITY MODELS

Model Name	Mean Value Function $m(t)$	Intensity Function $\lambda(t)$
Generalized Goel [8]	$m(t) = a(1 - e^{-bt^c})$	$\lambda(t) = abct^{c-1} e^{-bt^c}$
Goel-Okumoto [8]	$m(t) = a(1 - e^{-bt})$	$\lambda(t) = abe^{-bt}$
Gompert [8]	$m(t) = ake^{-bt}$	$\lambda(t) = ab \ln(k) k^{e^{-bt}} e^{-bt}$
Inflection S-Shaped [8]	$m(t) = \frac{a(1 - e^{-bt})}{1 + \beta e^{-bt}}$	$\lambda(t) = \frac{abe^{-bt}(1 + \beta t)}{(1 + \beta e^{-bt})^2}$
Logistic Growth[8]	$m(t) = \frac{a}{1 + ke^{-bt}}$	$\lambda(t) = \frac{abke^{-bt}}{(1 + ke^{-bt})^2}$
Modified Duane [8]	$m(t) = a \left[ 1 - (b/(b+t))^c \right]$	$\lambda(t) = acb^c (b+t)^{1-c}$
Musa-Okumoto [5]	$m(t) = a \ln(1 + bt)$	$\lambda(t) = ab/(1 + bt)$
Yamada imperfect debugging model 1 [25]	$m(t) = \frac{ab(e^{\alpha t} - e^{-bt})}{a + b}$	$\lambda(t) = \frac{ab(\alpha e^{\alpha t} + e^{-bt})}{\alpha + b}$
Yamada Rayleigh [26]	$m(t) = a(1 - e^{-r\alpha(1 - e^{(-\beta t^2/2)})})$	$\lambda(t) = ar\alpha\beta t e^{-r\alpha(1 - e^{(-\beta t^2/2)}) - \beta t^2/2}$
Delayed S-Shaped[8]	$m(t) = a(1 - (1 + bt)e^{-bt})$	$\lambda(t) = ab^2 t e^{-bt}$
Yamada imperfect debugging model 2 [25]	$m(t) = a(1 - e^{-bt})(1 - \frac{\alpha}{b}) + \alpha t$	$\lambda(t) = abe^{-bt}(1 - \frac{\alpha}{b}) + \alpha a$
Yamada exponential [26]	$m(t) = a(1 - e^{-r\alpha(1 - e^{-\beta t})})$	$\lambda(t) = ar\alpha\beta e^{-r\alpha(1 - e^{-\beta t}) - \beta t}$
P-N-Z Model [27]	$m(t) = \frac{a(1 - e^{-bt})(1 - \frac{\alpha}{b}) + \alpha t}{1 + \beta e^{-bt}}$	$\lambda(t) = \frac{abe^{-bt}(1 - \frac{\alpha}{b}) + \alpha}{1 + \beta e^{-bt}} + \frac{ab\beta e^{-bt}(1 - \frac{\alpha}{b})(1 - e^{-bt}) + \alpha t}{(1 + \beta e^{-bt})^2}$
P-Z Model [28]	$m(t) = \frac{1}{(1 + \beta e^{-bt})} \left( (c + a)(1 - e^{-bt}) - \frac{ab}{b - \alpha} (e^{-\alpha t} - e^{-bt}) \right)$	$\lambda(t) = \frac{b(c + a)(1 + \beta)e^{-bt} - [be^{-bt}(1 + \beta e^{-\alpha t}) - \alpha e^{-\alpha t}(1 + \beta e^{-bt})]}{(1 + \beta e^{-bt})}$
Pham Zhang IFD [29]	$m(t) = a - ae^{-bt}(1 + (b + d)t + bdt^2)$	$\lambda(t) = ae^{-bt} [bt(b - d) + d(b^2 t^2 - 1)]$
Zhang-Teng-Pham [30]	$m(t) = \frac{a}{p - \beta} \left[ \left( 1 - \frac{(1 + \alpha)e^{-bt}}{1 + \alpha e^{-bt}} \right)^{\frac{c}{b}(p - \beta)} \right]$	$\lambda(t) = \frac{ac}{1 + \alpha e^{-bt}} \left[ \left( \frac{(1 + \alpha)e^{-bt}}{1 + \alpha e^{-bt}} \right)^{\frac{c}{b}(p - \beta)} \right]$

2. The mean square error (MSE) measures the deviation between the predicted values with the actual observations and is defined as [31]:

$$MSE = \frac{\sum_{i=1}^k (m_i - \hat{m}(t_i))^2}{k - p} \tag{2}$$

3. The mean absolute error (MAE) is similar to MSE, but the way of measuring the deviation is by the use of absolute values. It is defined as [32]:

$$MAE = \frac{\sum_{i=1}^k |m_i - \hat{m}(t_i)|}{k - p} \tag{3}$$

4. The mean error of prediction (MEOP) sums the absolute value of the deviation between the actual data and the estimated curve and is defined as [33]:

$$MEOP = \frac{\sum_{i=1}^k |\hat{m}(t_i) - m_i|}{(k - p + 1)} \tag{4}$$

- The accuracy of estimation (AE) can reflect the difference between the estimated numbers of all errors with the actual number of all detected errors. It is defined as [32]:

$$AE = \left| \frac{M_a - a}{M_a} \right| \quad (5)$$

where  $M_a$  and  $a$  are the actual and estimated cumulative number of detected errors after the test, respectively.

- The noise is defined as [34]:

$$Noise = \sum_{i=1}^k \left| \frac{\lambda(t_i) - \lambda(t_{i-1})}{\lambda(t_{i-1})} \right| \quad (6)$$

- The predictive-ratio risk (PRR) is defined as [35]:

$$PRR = \sum_{i=1}^k \frac{\hat{m}(t_i) - m_i}{\hat{m}(t_i)} \quad (7)$$

which measures the distance of model estimates from the actual data against the model estimate.

- The variance is defined as [23], [24]:

$$Variance = \sqrt{\frac{1}{k-1} \sum_{i=1}^k (m_i - \hat{m}(t_i) - Bias)^2} \quad (8)$$

which is standard deviation of prediction bias.

- The Root Mean Square Prediction Error (RMSPE) is a measure of the closeness with which the model predicts the observation. It is defined as [23], [24]:

$$RMSPE = \sqrt{Variance^2 + Bias^2} \quad (9)$$

R square (Rsqr) can measure how successful the fit is in explaining the variation of the data.

- It is defined as [32]:

$$Rsqr = 1 - \frac{\sum_{i=1}^k (m_i - \hat{m}(t_i))^2}{\sum_{i=1}^k (m_i - \sum_{j=1}^k m_j / n)^2} \quad (10)$$

- The sum of squared errors (SSE) is defined as [30]:

$$SSE = \sum_{i=1}^k (m_i - \hat{m}(t_i))^2 \quad (11)$$

- The Theil statistic (TS) is the average deviation percentage over all periods with regard to the actual values. The closer, Theil's Statistic is to zero, the better the prediction capability of model. It is defined as [36]:

$$TS = \sqrt{\frac{\sum_{i=1}^k (\hat{m}(t_i) - m_i)^2}{\sum_{i=1}^k m_i^2}} \times 100\% \quad (12)$$

In equations 1 to 12, above,  $k$  represents the sample size of the data set and  $p$  is the number of parameters.

The comparison criteria, Root Mean Square Prediction Error (RMSPE) is a combination of the comparison criteria 'bias' and 'variance'. The criteria MSE, MAE and MEOP are used to measure the deviation whereas the criteria AE and SSE measure the errors. In order to avoid the replication of the

TABLE II  
MIDDLE SIZE SOFTWARE PROJECT FAILURE DATA

Week	Cumulative Faults	Week	Cumulative Faults	Week	Cumulative Faults
1	15	8	134	15	179
2	35	9	139	16	182
3	60	10	141	17	184
4	74	11	148	18	185
5	94	12	149	19	187
6	102	13	157	20	191
7	114	14	173	21	192

criteria and in order to investigate the effectiveness of software reliability models, a set of seven distinct comparison criteria namely mean absolute error (MAE), accuracy of estimation (AE), noise, predictive-ratio risk (PRR), Root Mean Square Prediction Error (RMSPE), R square (Rsqr) and Theil statistic (TS) are proposed to compare models quantitatively.

#### IV. MODEL EVALUATION AND COMPARISON

In order to evaluate and compare the models, failure data considered by [37] is used in this research paper. As reported by the researchers, this data set is from the testing process on a middle-size software project. Table 2 shows failure data. First column present failure time in weeks and second column presents cumulative number of failures. The values of the parameters for these sixteen NHPP SRMs have been estimated using the MLE technique and confidence bounds of 95%. The estimated values of the parameters have been provided in Table 3.

TABLE III  
PARAMETER ESTIMATION OF THE SRMS

Model Name	Parameters
Generalized Goel	$a = 185.36, b = 8.0 \times 10^{-4}, c = 3.1005$
Goel-Okumoto	$a = 215.763, b = 0.108$
Gompert	$a = 191.787, b = 0.242, c = 5.972 \times 10^{-2}$
Inflection S-Shaped	$a = 203.307, b = 0.155, \beta = 0.524$
Logistic Growth	$a = 188.349, b = 0.332, k = 7.215$
Modified Duane	$a = 237.581, b = 40.437, k = 4.096$
Musa-Okumoto	$a = 113.003, b = 0.230$
Yamada imperfect debugging model 1	$a = 128, b = 0.189, c = 2.467 \times 10^{-2}$
Yamada Rayleigh	$a = 307.2, b = 4.8 \times 10^{-2}, c = 3.301 \times 10^{-2}$
Delayed S-Shaped	$a = 190.796, b = 0.296$
Yamada imperfect debugging model 2	$a = 128, b = 0.191, c = 3.255 \times 10^{-2}$
Yamada exponential	$a = 307.2, \alpha = 6.400 \times 10^{-2}, \beta = 0.154$
P-N-Z Model	$a = 128, b = 0.122,$
P-Z Model	$a = 128, b = 0.122, c = 81,$
Pham Zhang IFD	$a = 190.795, b = 0.296, d = 1.0 \times 10^{-5}$
Zhang-Teng-Pham	$a = 186.350, b = 5.223 \times 10^{-2}, c = 0.81$

The values of the seven comparison criteria considered in this research paper has been obtained. using relevant equations (Eqs. 3, 5, 7, 9, 10 and 12) The estimated and optimal values of the parameters are given in Table 4.

From the comparison of rankings of the sixteen SRMs based on the values of all these seven criteria as given in Table 4, it is observed that the ranking of the SRMs varies with respect to the release and criteria. No single model is best suitable for all comparison criteria. In order to avoid this problem it is proposed to apply matrix methodology to analyze the performance and rank the SRMs based on all these seven criteria taken collectively.

V. METHODOLOGY ADOPTED

To depart from complexity of the formulation of objective and constraint functions that occur when the mathematical programming model is used in a multi-attributes decision problem, a modest attempt is made in this paper to develop a deterministic quantitative model based on matrix operations for the purpose of ranking of software reliability models. Matrix methods have previously been used for power quality evaluation in deregulated power system [38] and optimizing selection of power plants [39]. The brief introduction to the basic concepts matrix operations is presented in this section.

A. Criteria Matrix

Each software reliability model at this stage is characterized by multiple criteria, which need to be converted into a single number index that will be used to rank the software reliability models. The matrices lend themselves easily to mechanical manipulations and are suitable for computer processing. The ratings and the relative aggregated weights of the comparison criteria for a software reliability model are stored in a matrix that is called 'Criteria Matrix'. The size of this matrix will be n x n corresponding to n criteria. The diagonal elements (a<sub>ii</sub>'s or a<sub>i</sub>'s) and the off-diagonal elements (a<sub>ij</sub>'s) of this matrix give the ratings and the relative aggregated weights of the comparison criteria, respectively. Thus, the criteria matrix is a combination of two matrices namely 'criteria rating matrix' and 'criteria relative weight matrix'.

Criteria Rating Matrix: This is a diagonal matrix whose elements (a<sub>ii</sub>'s or a<sub>i</sub>'s) represent the ratings of different comparison criteria for a software reliability model and is represented as follows:

$$\begin{bmatrix} a_{11} & 0 & 0 & \dots & 0 \\ 0 & a_{22} & 0 & \dots & 0 \\ \vdots & & & & \\ 0 & 0 & 0 & \dots & a_{nn} \end{bmatrix}$$

TABLE IV  
DATABASE FOR ESTIMATED AND OPTIMAL VALUES OF ATTRIBUTES FOR EACH ALTERNATE SRM

	AE	Rank	Noise	Rank	Rsqr	Rank	TS	Rank	PRR	Rank	RMSE	Rank	MAE	Rank
Generalized Goel	0.034614	13	15.0034	16	0.380572	13	28.4307	13	11135.07	16	74.47332	13	35.15119	13
Goel-Okumoto	0.007959	5	2.28572	6	0.992825	2	3.0598	2	0.147447	3	4.54664	2	3.843456	1
Gompert	0.01836	8	3.78193	11	0.982757	4	4.74347	4	0.191337	4	7.045852	3	5.853303	4
Inflection S-Shaped	0.001625	3	2.56465	7	0.993054	1	3.01068	1	0.098526	1	4.467552	1	3.942101	2
Logistic Growth	0.025522	12	4.75978	14	0.973298	6	5.90287	6	0.393583	8	8.851496	4	7.692465	8
Modified Duane	0.014315	7	1.54903	3	0.775166	12	17.1287	12	1.069524	9	53.44483	12	27.12543	12
Musa-Okumoto	0.03768	14	1.63260	4	0.981355	5	4.93260	5	0.217156	5	9.35941	8	6.397615	5
Yamada imperfect debugging model 1	0.020482	11	1.49818	2	0.939523	10	8.88356	10	0.318831	7	24.38592	10	13.29734	10
Yamada Rayleigh	0.925064	16	7.8719	15	0.879863	11	12.5208	11	5.096563	14	27.27232	11	18.22131	11
Delayed S-Shaped	0.020405	9	4.21039	12	0.972967	7	5.93937	7	1.892668	10	8.862182	6	7.120771	6
Yamada imperfect debugging model 2	0.001086	1	1.38165	1	0.96755	9	6.50725	9	0.236762	6	16.4287	9	9.250905	9
Yamada exponential	0.904552	15	3.40447	10	0.031786	15	35.545	14	2.535634	12	114.2261	15	58.0289	15
P-N-Z Model	0.009734	6	1.87603	5	0.987202	3	4.08667	3	0.122122	2	8.859432	5	5.680468	3
P-Z Model	0.001507	2	2.5994	8	0.108714	14	38.0367	16	10.31861	15	119.504	16	67.87029	16
Pham Zhang IFD	0.020407	10	4.21064	13	0.972964	8	5.93968	8	1.893453	11	8.862517	7	7.516719	7
Zhang-Teng-Pham	0.006976	4	2.86226	9	0.00392	16	36.0529	15	2.677499	13	104.2807	14	57.67601	14
optimal	0.001086		1.38165		0.993054		3.01068		0.098526		4.467552		3.843456	

As these criterion ratings are different for different software reliability models, hence, the criteria rating matrix differs from model to model. The criteria ratings are determined as under:

Case - I: When smaller value of the criterion represents fitting well to the actual data i.e. is the best value:

$$\text{Criteria rating} = \frac{\text{Criterion Maximum Value in the Database} - \text{Criterion Value}}{\text{Criterion Maximum Value in the Database} - \text{Criterion Minimum Value in the Database}}$$

Case - II: When bigger value of the criterion represents fitting well to the actual data i.e. is the best value:

$$\text{Criteria rating} = \frac{\text{Criterion Value} - \text{Criterion Minimum Value in the Database}}{\text{Criterion Maximum Value in the Database} - \text{Criterion Minimum Value in the Database}}$$

Criteria Relative Weight Matrix: The Criteria Relative Weight Matrix is formed on the basis of the aggregated weights of different criteria. The off diagonal elements of this matrix represent the relative weights of the criteria e.g. the element  $(a_{ij})$  of this matrix will give the relative weight of  $j^{\text{th}}$  criterion in respect of  $i^{\text{th}}$  criterion. All diagonal elements of this matrix are zero because there is no significance of comparing a criterion with respect to itself. Mathematically  $a_{ij}$  = weight of  $j^{\text{th}}$  criteria/ weight of  $i^{\text{th}}$  criteria. In this research paper, all the criteria are equally weighted however the alternative weighing schemes can be applied using expert opinion and are based on engineering judgments. The criteria relative weight matrix considered in the present study can be written as under:

$$\begin{bmatrix} 0 & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & 0 & a_{23} & \dots & a_{2n} \\ \vdots & & & & \\ a_{n1} & a_{n2} & a_{n3} & \dots & 0 \end{bmatrix}$$

Thus the 'Criteria Matrix' corresponding to 'n' criteria, in general, is written as:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ \vdots & & & & \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix}$$

**B. Permanent Function Representation**

Variable Permanent Function or simply known as Permanent is a standard matrix function that is used in combinatorial mathematics [40]. It is a powerful tool for multi-criteria based evaluation and ranking of the systems in ascending or descending order. The Permanent is similar to the determinant of a matrix with a difference that no negative term appears in the permanent. Computer software is developed to determine the value of the Permanent of the 'Criteria Matrix'. The algorithm is:

$$(A) P \leftarrow 0; X_i \leftarrow a_{in} - \frac{1}{2} \sum_{i,j=1}^n a_{ij}; \text{sgn} \leftarrow -1$$

$$(B) \text{sgn} \leftarrow -\text{sgn}; P \leftarrow \text{sgn},$$

Get next subset of  $(1, 2, \dots, n - 1)$  from NEXSUB;

if empty, go to (C) and if  $j$  was deleted, then :

$$z \leftarrow -1; \text{ otherwise, } z \leftarrow 1;$$

$$x_i \leftarrow x_i + z a_{ij}; (i = 1, 2, \dots, n)$$

$$(C) P \leftarrow P.x_i; (i = 1, 2, \dots, n); p \leftarrow p + p$$

if more subsets remain, to (B);

$$\text{Permanant} \leftarrow 2(-1)^{n-1} p; \text{ EXIT.}$$

**ALGORITHM NEXSUB**

(A)[First entry]  $m \leftarrow 1; j \leftarrow 1; z \leftarrow 1; \text{ exit.}$

(B)[Later entry]  $m \leftarrow m + 1; x \leftarrow m; j \leftarrow 0;$

(C)  $j \leftarrow j + 1; x \leftarrow \frac{x}{2}; \text{ if } x \text{ is an integer, to (C).}$

(D)  $z \leftarrow (-1)^{\frac{x+1}{2}}; \text{ if } = 2^n, \text{ final exit; EXIT.}$

**C. Case study**

The objective of this demonstration is to test the suitability of the developed matrix method so that a comprehensive ranking of the alternative SRMs could be made combining various criteria relevant to SRMs for the data set of three releases of large medical system provided in Table 2.

Table 5 shows the Permanent value and the ranking of the alternate SRMs based on the contributing criteria. The overall ranking is based on Permanent value of each of the alternate SRM that is determined considering all seven contributing criteria together using matrix method. The alternate SRM with highest permanent value is given rank no. - 1, that with second highest Permanent value is given rank no. - 2, and so on.

The results, so obtained, depict that the P-Z model is ranked

TABLE V  
SRMS RANKING BASED ON DBA

Model Name	Permanent	Rank	Model Name	Permanent	Rank
Generalized Goel	0.51102	4	Yamada Rayleigh	0.77544	6
Goel-Okumoto	0.99994	16	Delayed S-Shaped	0.94881	11
Gompert	0.96861	13	Yamada imperfect debugging model 2	0.91554	8
Inflection S-Shaped	0.99846	15	Yamada exponential	0.15371	2
Logistic Growth	0.93988	9	P-N-Z Model	0.97131	14
Modified Duane	0.63637	5	P-Z Model	0.00058	1
Musa-Okumoto	0.96011	12	Pham Zhang IFD	0.94263	10
Yamada imperfect debugging model1	0.85234	7	Zhang-Teng-Pham	0.15922	3

at number one, Yamada exponential is number two whereas inflection S-shaped and Goel-Okumoto models are ranked lowest at 15 & 16 numbers respectively. It is well established that the proposed method is suitable for distinct ranking of the models for any data sets based on a number of criteria taken collectively.

## VI. CONCLUSION

This paper addresses the issue of performance analysis of software reliability models. The decision has unrestricted choices in exploring the influences of various different set of model selection criteria to final decision. As soon as a complete set of criteria for SRMs selection, along with the set of alternative SRMs and their level of criteria are formulized, and efficient rationalization process around multi-attribute decision model 'matrix method' can be performed. This model allows a decision maker to perform, not just a general analysis, but also other various focused analyses regarding his or her personal preferences.

The proposed method is suitable for ranking of SRMs based on a number of conflicting criteria taken all together with equal or unequal weights. This method uses a simple mathematical formulation and straight forward matrix operation to be performed by computing machines and is capable of solving complex multi-attributes decision problems, incorporating both quantitative and qualitative factors.

## REFERENCES

- [1] H. Pham, and X. Zhang, "A software cost model with warranty and risk costs" *IEEE Trans. on Computers*, 48 (1), 1999, pp. 71-75.
- [2] M. Xie, *Software Reliability Modelling*, World Scientific Publishing Co. Ltd., 1991.
- [3] A. Goel, and K. Okumoto, "Time dependent error-detection rate model for software reliability and other performance measures," *IEEE Trans. on Reliability*, R-28(3), 1979, pp. 206-211.
- [4] M. R. Lyu, *Handbook of Software Reliability Eng.*, McGraw-Hill, 1996.
- [5] J. D. Musa, and K. Okumoto, "A logarithmic Poisson execution time model for software reliability measurement," *Conf. Proc. 7<sup>th</sup> International Conf. on Softw. Engineering*, 1983, pp. 230-237.
- [6] C. Wohlin, "Software testing and reliability for telecommunication systems," *Softw. Engineering '86*, ed. D. Barnes and P. Brown, Peter Peregrinus Ltd., Stevenage, United Kingdom, 1986, pp. 27-42.
- [7] J. D. Musa, A. Iannino, and K. Okumoto, *Software Reliability Measurement, Predication, Application*, McGraw Hill, 1987.
- [8] C. Y. Huang, M. R. Lyu, and S. Y. Kuo, "A unified scheme of some non-homogenous Poisson process models for software reliability estimation," *IEEE Trans. on Softw. Engineering*, vol. 29, no. 3, March 2003, pp. 261-269.
- [9] C. G. Bai, "Bayesian network based software reliability prediction with an operational profile," *J. Syst. Softw.*, vol. 77, no. 2, 2005, pp.103-112.
- [10] L. Tian, and A. Noore, "On-line prediction of software reliability using an evolutionary connectionist model," *J. Syst. Softw.*, vol. 77, no. 2, 2005, pp. 173-180.
- [11] W. L. Wang, D. Pan, and M. H. Chen, "Architecture-based software reliability modeling," *J. Syst. Softw.*, vol. 79, no. 1, 2006, pp. 132-146.
- [12] X. Zhang, and H. Pham, "Software field failure rate prediction before software deployment," *J. Syst. Softw.*, Vol.79, no. 3, 2006, pp. 291-300.
- [13] C. Y. Huang, "Performance analysis of software reliability growth models with testing-effort and change-point," *J. Syst. Softw.*, vol. 76, no. 2, 2005, pp. 181-194.
- [14] C. Y. Huang, and C. T. Lin, "Software reliability analysis by considering fault dependency and debugging time lag," *IEEE Trans. Reliability*, vol. 55, no. 3, 2006, pp. 436-450.
- [15] Q. P. Hu, M. Xie, S. H. Ng, and G. Levitin, "Robust recurrent neural network modeling for software fault detection and correction prediction," *Reliability Engineering and System Safety*, vol. 92 no. 3, 2007, pp. 332-340.
- [16] D. R. Jeske, and X. Zhang, "Some successful approaches to software reliability modeling in industry," *J. Syst. Softw.*, vol. 74, no. 1, 2005, pp. 85-99.
- [17] G.H. Schick, and R.W. Wolverton, "An analysis of competing software reliability models," *IEEE Trans. on Softw. Engineering*, March 1978, pp. 104-120.
- [18] A.N. Sukert, "Empirical validation of three software errors predictions models," *IEEE Trans. on Reliability*, August 1979, pp. 199-205.
- [19] S. Brocklehurst, P.Y. Chan, B. Littlewood, and J. Snell, "Recalibrating software reliability models," *IEEE Trans. on Softw. Engineering*, vol. SE-16 no. 4, April 1990, pp. 458-470.
- [20] A.L. Goel, "Software reliability models: assumption, limitations, and applicability," *IEEE Trans. on Softw. Engineering*, December 1985, pp. 1411-1423.
- [21] Abdel-Ghaly, P.Y. Chan, and B. Littlewood, "Evaluation of competing software reliability predictions," *IEEE Trans. on Softw. Engineering*, vol. SE-12 no. 12, September 1986, pp. 950-967.
- [22] T.M. Khoshgoftaar, and T. Woodcock, "Software reliability model selection: A case study," *Proc. of the 2<sup>nd</sup> International Symposium on Softw. Reliability Engineering*. IEEE Computer Society Press, Austin, TX: 1991, pp. 183-191.
- [23] C. Y. Huang, and S. Y. Kuo, "Analysis of incorporating logistic testing effort function into software reliability modeling," *IEEE Trans. on Reliability*, vol. 51, no. 3, 2002, pp. 261-270.
- [24] K. Pillai, and V. S. S. Nair, "A model for software development effort and cost estimation," *IEEE Trans. on Softw. Engineering*, vol. 23, no. 8, 1997, pp. 485-497.
- [25] S. Yamada, K. Tokuno, and S. Osaki, "Imperfect debugging models with fault introduction rate for software reliability assessment," *International J. Syst. Science*, vol. 23, no. 12, 1992.
- [26] H. Pham, "Software reliability and cost models: perspectives, comparison and practice", *European J. of Operational Research*, vol. 149, 2003, p. 475- 489.
- [27] H. Pham, L. Nordmann, and X. Zhang, "A general imperfect software debugging model with s-shaped fault detection rate," *IEEE Trans. Reliability*, vol. 48, June 1999, pp. 169-175.
- [28] H. Pham and X. Zhang, "An NHPP software reliability models and its comparison," *International J. of Reliability, Quality and Safety Engineering*, vol. 14, no. 3, 1997, pp. 269-282.
- [29] H. Pham, *System software reliability*, Springer London, 2006.
- [30] X. Zhang, X. Teng and H. Pham, "Considering Fault Removal Efficiency in Software Reliability Assessment", *IEEE Trans. on Systems, Man, & Cybernetics-Part A*, vol. 33, no.1, 2003, pp. 114-120.
- [31] S. Hwang and H. Pham, "Quasi-renewal time-delay fault-removal consideration in software reliability modelling," *IEEE Trans. on systems, man and cybernetics-Part A:Systems and humans*, vol. 39, no. 1, January 2009.
- [32] K. C. Chiu, Y. S. Huang, and T. Z. Lee, "A study of software reliability growth from the perspective of learning effects," *Reliability Engineering and System Safety*, 2008, pp. 1410-1421.
- [33] M. Zhao, and M. Xie, "On the log-power NHPP software reliability model," *Proceedings of the Third IEEE International Symposium on Softw. Reliability Engineering*, Research Triangle Park, North Carolina, 1992, pp. 14-22.
- [34] M. R. Lyu, and A. Nikora, "Applying software reliability models more effectively," *IEEE Softw.*, 1992, 43-52.
- [35] H. Pham and C. Deng, "Predictive-ratio risk criterion for selecting software reliability models," *Proc. Ninth International Conf. On Reliability and Quality in Design*, August 2003.
- [36] P. L. Li, J. Herbsleb, and M. Shaw, "Forecasting field defect rates using a combined time-based and metrics-based approach: a case study of OpenBSD," *Proceedings of the 16th IEEE International Symposium on Softw. Reliability Engineering*, Chicago, IL, 2005, pp. 193-202.
- [37] R. Peng, Q.P. Hu, and S.H. Ng, "Incorporating Fault Dependency and Debugging Delay in Software Reliability Analysis," *Proceedings of the IEEE ICMIT*, 2008, pp.641-645.
- [38] S. Dahiya et al., "Power quality evaluation in deregulated power system using matrix method," *International J. Global Energy Issues*, vol. 28, no. 1, 2007.
- [39] R. K. Garg, V. K. Gupta, and V. P. Agrawal, "Quality evaluation of thermal power plants by graph theoretical methodology," *International J. of Power and Energy Systems*, 27 (1), 2007, pp. 42-48.
- [40] M. Marcus, and H. Minc, "Permanents," *American Mathematics*, 72, 1965, pp. 571-91.

**RajPal Garg** was born in Haryana, India in 1963. He is pursuing a Doctors Degree in Computer Science and Engineering under the Faculty of Engineering and Technology at the Singhania University, Jhunjhunu (Rajasthan), India. He has obtained his Master of Technology and Master of Science Degrees in Information Technology and Maths from Punjabi

University, Patiala, and Kurukshetra University, Kurukshetra, India in 2003, and 1986 respectively.. His research interests include software engineering, and reliability.

**Kapil Sharma** was born in Haryana, India in 1977. He is pursuing a Doctors Degree in Computer Science and Engineering under the Faculty of Engineering and Technology at the M. D. University, Rohtak (Haryana), India. He has obtained his Bachelor of Engineering and Master of Technology Degrees in Computer Science & Engineering and Information Technology from M. D. University, Rohtak, and IASE, Rajasthan, India in 2000, and 2005 respectively. Presently, he is working as a Faculty of the Department of Information Technology at Guru Premsukh Memorial College of Engineering, Budhpur, Delhi, India. His research interests include software engineering, and reliability.

**Rajive Kumar** was born in Hardwar (UP), India in 1963. He has obtained degrees of Master of Science and Master of Philosophy from Roorkee University, Roorkee, India, and Doctor of Philosophy from Indian Institute of Technology, Delhi, India, all in Mathematics, in 1982, 1985, and 1990 respectively. Presently, he is working as Professor in the Department of Mathematics with Deenbandhu Chhotu Ram University of Science and Technology, Murthal, Haryana, India. His research interests include numerical methods and software engineering. Dr. Kumar has published a number of research papers in International journals and conferences. Presently, he is guiding seven research scholars.

**R. K. Garg** was born in Haryana, India in 1970. He has obtained degrees of Bachelor of Engineering with Honours, Master of Engineering, and Doctor of Philosophy in Mechanical Engineering from M. D. University, Rohtak, Panjab University, Chandigarh, and M. D. University, Rohtak, Haryana, India in 1991, 1996, and 2006 respectively. Presently, he is working as Professor in the Department of Mechanical Engineering with Deenbandhu Chhotu Ram University of Science and Technology, Murthal, Haryana, India. His research interests include system design, reliability modeling & analysis, and power & energy engineering. Dr. Garg has published a number of research papers in International journals and conference, and has guided a number of Masters degree dissertations. Presently, he is guiding five research scholars.