Applying Various Learning Curves to Hyper-Geometric Distribution Software Reliability Growth Model

Rong-Huei Hou & Sy-Yen Kuo

Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, R.O.C.
Email: sykuo@cc.ee.ntu.edu.tw.

Yi-Ping Chang

Department of Business mathematics
Soochow University
Taipei, Taiwan, R.O.C.

Abstract

The Hyper-Geometric Distribution software reliability growth Model (HGDM) has been shown to be able to estimate the number of faults initially resident in a program at the beginning of the test-and-debug phase. A key factor of the HGDM is the "sensitivity factor", which represents the number of faults discovered and rediscovered at the application of a test instance. The learning curve incorporated in the sensitivity factor is generally assumed to be linear in the literature. However, this assumption is apparently not realistic in many applications. We propose two new sensitivity factors based on the exponential learning curve and the S-shaped learning curve, respectively. Furthermore, the growth curves of the cumulative number of discovered faults for the HGDM with the proposed learning curves are investigated. Extensive experiments have been performed based on two real test/debug data sets, and the results show that the HGDM with the proposed learning curves estimates the number of initial faults better than previous approaches.

1. Introduction

In the software development process, a program is developed through these four steps: specification, design, coding, and test–and–debug. In general, the test–and–debug procedure is performed by test specialists, independent of the programmers of the software. Although the programmers usually have the confidence that the program is completely debugged, many faults are detected by the test team.

However, it is difficult to know definitely how many faults exist in a program at the beginning of the test–and–debug process. Thus, the test specialists are unable to determine when the program under test can be released to service. Obviously, it is important and helpful to be able to estimate the number of initial resident faults. If the accuracy of this estimate is high, the test specialists can determine more precisely when to release the program to service.

In the literature, several Software Reliability Growth Models (SRGMs) based on the Non–Homogeneous Poisson Process (NHPP) to estimate the number of residual software faults have been proposed (e.g., Goel and Okumoto [1], Yamada *et. al.* [2], Ohba [3], and Musa and Okumoto [4]). The Hyper–Geometric Distribution software reliability growth Model (HGDM), first proposed by Tohma *et. al.* [5], was shown to be attractive. Tohma *et. al.* [6–8] and Jacoby *et. al.* [9–10] have made a series of studies on the HGDM recently.

A key factor of the HGDM is the "sensitivity factor" (denoted by w_i), which represents the number of faults discovered and rediscovered at the application of the i^{th} test instance. This model assumes that the learning curve of testers' skill involved in w_i is linear. However, the assumption of linear learning curve is apparently not realistic in many applications. Traditionally, the exponential or the S–shaped curve is used to represent the human learning process. In this paper, we thus propose two new w_i functions based on the exponential and the S–shaped learning curves, respectively.

Experiments have been performed based on two real test/debug data sets, and the results show that the HGDM with the proposed learning curves can estimate the number of initial faults better than previous approaches. Furthermore, the growth curves of the cumulative number of discovered faults for the HGDM with these two new sensitivity factors are also investigated.

The remaining presentation is organized as follows.

1071-9458/94 $4.00 © 1994 IEEE

Existing software reliability growth models based on the Non-Homogeneous Poisson Process (NHPP) are summarized in Section 2. The basic concept and the precise formulation of the HGDM are briefly reviewed in Section 3. Two new sensitivity factors are proposed and discussed in Section 4. We estimate the parameters of the HGDM with different sensitivity factors by using the Least Square Method in Section 5. Section 6 discusses the growth curves of the cumulative number of discovered faults for the HGDM with these two new sensitivity factors. The experiments and results are presented in Section 7 followed by the conclusions in Section 8.

2. Existing Software Reliability Growth Models

In general, software reliability growth model is defined by the mathematical relationship that exists between the time span of testing a program and the cumulative number of faults discovered [3]. In this section, we will summarize four interesting existing SRGMs based on the NHPP. In particular, these four models will be used for comparison with the HGDMs in Section 7.

• **Goel–Okumoto model**

Goel and Okumoto [1] first proposed a SRGM based on the NHPP. This model is also called the *exponential* SRGM, which assumes that all faults are independent and have the same probability to be detected. The mean value function showing an exponential growth curve is given by

$$E(t) = m(1 - e^{-at}),\ m > 0,\ a > 0\ , \qquad (1)$$

where m is the expected number of faults to be eventually detected and a represents the fault detection rate.

• **Delayed S–shaped model**

The *Delayed S–shaped* SRGM was proposed by Yamada *et. al.* [2]. They assume that a testing process contains not only a failure detection process, but also a fault isolation process. The mean value function of this model is

$$D(t) = m(1 - (1 + at)e^{-at}),\ m > 0,\ a > 0\ , \qquad (2)$$

which shows an S–shaped growth curve. Like the Goel–Okumoto model, the parameters m and a are the total expected number of faults and the fault detection rate, respectively.

• **Inflection S–shaped model**

The *Inflection S–shaped* SRGM, presented by Ohba [3], assumes a software failure detection phenomenon with a mutual dependence of detected faults. In the fault detection process, the more faults are detected, the more undetected faults become detectable. This NHPP model has a mean value function

$$I(t) = m\frac{(1 - e^{-at})}{(1 + be^{-at})},\ m > 0,\ a > 0,\ b > 0\ . \qquad (3)$$

The parameter m is again the total number of faults to be detected while a and b are the fault detection rate and the inflection factor, respectively.

• **Logarithmic Poisson Execution Time Model**

The logarithmic poisson execution time model was proposed by Musa and Okumoto [4]. The underlying software failure process is modeled as a logarithmic Poisson process with an intensity function that decreases exponentially with the expected number of faults experienced. This model belongs to the infinite-fault category [11], and then the concept of the number of initial faults does not exist. The mean value function is

$$\mu(t) = \frac{1}{\theta}\ln(\lambda_0\theta t + 1),\ \lambda_0 > 0,\ \theta > 0, \qquad (4)$$

where λ_0 and θ represent the initial failure intensity and the reduction rate of the software failure rate, respectively.

3. Review of HGDM

In this section, we will briefly review the hypergeometric distribution software reliability growth model for estimating the number of faults initially resident in a given program [5–10].

In the software development process, programmers first debug their products by themselves. After this preliminary debugging, the products will be passed to test workers for further test–and–debug. In general, a program is assumed to have m initial faults when the test–and–debug stage begins. Test operations performed in a day or a week may be called a *"test instance"*. Test instances are denoted by t_i, $i = 1, 2, \ldots, n$ in accordance with the order of applying them. Therefore, the test of a program can be viewed as a "set" of test instances. The *"sensitivity factor"*, w_i, represents how many faults are discovered by the application of the test instance t_i. Each fault can be classified into one of the two categories, **newly discovered** faults or **rediscovered** faults. Some of the faults detected by t_i may have been detected previously by the application of t_1 through t_{i-1} test instances. Therefore, the number of newly detected faults at the application of the i^{th} test instance is not necessarily equal to w_i. That is, when the first test instance t_1 is applied, the number of newly detected faults is, of course, w_1. However, when t_2 is applied, the number of newly detected faults is not necessarily equal to w_2 because some of these w_2 faults may have been detected by t_1. This is true for all t_i, $i = 3, 4, \ldots, n$.

Considering the application of t_i, let C_{i-1} be the number of faults already detected so far by $t_1, t_2, \ldots, t_{i-1}$ and N_i be the number of faults newly detected by t_i. Then, some of the w_i faults may be those that already counted in C_{i-1}, and the remaining faults account for the newly detected faults. With the assumption that new faults will not be inserted into the program while correction is being performed, the conditional probability $Prob(N_i = x_i \mid m, w_i, C_{i-1})$ can be formulated as

$$Prob(N_i = x_i \mid m, w_i, C_{i-1}) = \frac{\binom{m - C_{i-1}}{x_i}\binom{C_{i-1}}{w_i - x_i}}{\binom{m}{w_i}}$$

where $C_{i-1} = \sum_{k=1}^{i-1} x_k$, $C_0 = 0$, and x_k is an observed instance of N_k. The expected value of C_i denoted by EC_i is [10]

$$EC_i = m\left[1 - \prod_{j=1}^{i}(1 - p_j)\right], \qquad (5)$$

where $p_i = w_i/m$ and m as well as the coefficients of p_i are unknown parameters to be estimated (see Section 5).

4. Sensitivity Factors

In this section, we will introduce the sensitivity factor, w_i, proposed in [5-10]. However, there are some weaknesses on the factor. To make the factor more realistic in applications, we thus propose two new w_i functions based on two typical human learning curves, the exponential and the S-shaped curves, respectively (plotted in Fig.1).

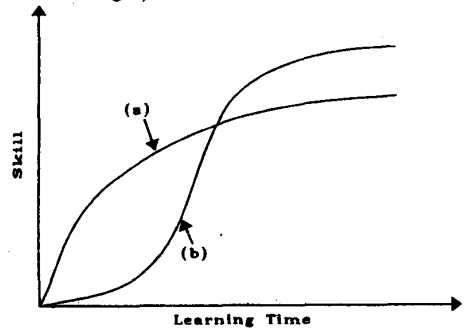

Fig.1 (a) Exponential learning curve; (b)S-shaped learning curve.

4.1. Interpretation of Sensitivity Factor

The sensitivity factor, w_i, is the number of faults discovered at the application of a test instance t_i. In general, w_i is usually defined as follows [5-10]:

$$\begin{aligned} w_i &= mu_i(ai + b), \text{ with} \qquad (6) \\ u_i &= \{\text{number of testers}(i) \text{ or computer time}(i) \text{ or number of test items}(i)\}, \end{aligned}$$

where $a > 0$ and i represents the i^{th} test instance. Eq.(6) means that w_i depends on the conditions in executing test operations of t_i such as the number of test specialists, the number of test items, the computer execution time, etc. For example, the larger the number of test specialists involved, the more faults will possibly be detected by t_i. Similarly, if more test items are checked in t_i, more faults are possibly detected. Furthermore, the term $(ai + b)$ represents the linear improvement of testers' skill along with the progress of the test instance t_i. That is to say, the skill of test workers will improve linearly when the test proceeds.

In order to explain the HGDM with new sensitivity factors more clearly, we will consider the following two cases for u_i, respectively.

1. u_i is a constant for all i. This case means that the number of test items (or the number of test workers, computer execution time, etc.) involved in the test process is unavailable or all the same for all t_i.
2. u_i is not a constant.

4.2. Learning Factor with Linear Curve

Considering the case that u_i is a constant, Eq.(6) can be rewritten as

$$w_i = m(ai + b), \forall\, i = 1, \ldots, n. \qquad (7)$$

If we profoundly explore the concept of sensitivity factor discussed above, we will find that the term, $(ai+b)$, can be regarded as a *"learning factor"* involved in the test-and-debug process. In other words, the test-and-debug process will be improved with experience. The term $(ai+b)$ also indicates that the learning factor is a linearly increasing function of i (the total number of test instances carried out so far). However, this assumption of linear curve is not realistic in many applications. Besides, this assumption will cause $w_i \longrightarrow \infty$ as $i \longrightarrow \infty$. That is, if the test-and-debug process is executed for an infinite number of test instances, an infinite number of faults will be discovered. This is not appropriate to the assumptions of the HGDM.

4.3. Learning Factor with Exponential Curve or S–shaped Curve

As described in the previous subsection, the assumption of linear learning factor is not so realistic. Traditionally, the exponential or the S–shaped curve is commonly used to interpret the human learning process. The exponential learning curve assumes that the skill of a tester grows faster at the initial test instances than at the later stage. On the other hand, the S–shaped learning curve assumes the test–team members will become more familiar with the test environment (e.g., test tools), and their test skill will gradually improve. In other words, the skill to discover faults grows slowly at the beginning stage of testing because the skill is not experienced initially; at the subsequent stage, the skill will increase quickly because the technique becomes versed; at the later stage, the skill again increases slowly. This is quite reasonable since the faults at the later stage are more difficult to be newly discovered than those at the beginning stage.

In this section, we thus propose two new w_i functions based on these two learning curves. The first w_i function is defined as

$$w_i = m(1 - e^{-ai}),\ a > 0. \tag{8}$$

The learning factor $(1 - e^{-ai})$ is an exponential curve, and will approach 1 as $i \longrightarrow \infty$. Therefore, Eq.(8) will cause $w_i \longrightarrow m$ as $i \longrightarrow \infty$, and it looks somewhat not realistic since sometimes we cannot detect all m faults in one instance when the testing resource (e.g., test specialists, test items, etc.) is not enough. In other words, it looks reasonable that the learning factor should approach a limit value p_{LT} instead of 1 as $i \longrightarrow \infty$. Therefore, Eq.(8) can be extended to be

$$w_i = mp_{LT}(1 - e^{-ai}),\ a > 0,\ 0 < p_{LT} \leq 1. \tag{9}$$

Considering the case that u_i is not a constant, Eq.(9) can be directly generalized as

$$w_i = mp_{LT}(1 - e^{-u_i ai}),\ a > 0,\ 0 < p_{LT} \leq 1. \tag{10}$$

Furthermore, w_i in Eq.(10) is ensured to be less than or equal to m as $i \longrightarrow \infty$.

The second new w_i function is defined as

$$w_i = m\frac{1}{1 + be^{-ai}},\ a > 0,\ b > 0. \tag{11}$$

The learning factor $1/(1 + be^{-ai})$ is a *"logistic function"* of i [12], and then a learning factor with a logistic function is called a *"logistic learning factor"*. For $i > 0$, this logistic function can be depicted as an exponential or an S–shaped curve. As $0 < b \leq 1$, the curve of $1/(1 + be^{-ai})$ is exponential; as $b > 1$, it is S–shaped. Therefore, we can utilize the logistic learning factor to describe either the exponential or the S–shaped learning curve. Similarly, in order to make the logistic learning factor approach p_{LT} as $i \longrightarrow \infty$, Eq.(11) can also be extended to be

$$w_i = mp_{LT}\frac{1}{1 + be^{-ai}},\ a > 0, b > 0, 0 < p_{LT} \leq 1. \tag{12}$$

If u_i is not a constant, Eq.(12) can be generalized as

$$w_i = mp_{LT}\frac{1}{1 + be^{-u_i ai}},\ a>0, b>0, 0 < p_{LT} \leq 1. \tag{13}$$

In Section 6, we will prove that the HGDM with these two new sensitivity factors can construct the exponential or the S–shaped growth curve of the cumulative number of discovered faults if the parameters (a, b, and p_{LT}) satisfy certain conditions.

5. Estimation of Parameters by Least Square Method

The parameters of the HGDMs can be estimated by the least square method. From the given sample observations, the sum of squares of errors is

$$\begin{aligned} S(p_i, m) &= \sum_{i=1}^{n}(C_i - EC_i)^2 \\ &= \sum_{i=1}^{n}\left\{C_i - m\left[1 - \prod_{j=1}^{i}(1 - p_j)\right]\right\}^2. \end{aligned} \tag{14}$$

For example, consider the logistic learning factor $p_{LT}/(1 + be^{-ai})$, and take the partial derivatives of $S(p_i, m)$ with respect to m, p_{LT}, a, and b. The following four equations can be obtained and the estimate of these four parameters can be solved by numerical methods.

$$(A)\quad \frac{\partial S(p_i, m)}{\partial m} = 2\sum_{i=1}^{n}\left\{C_i - m\left[1 - \prod_{j=1}^{i}(1 - p_j)\right]\right\} \left\{-\left(1 - \prod_{j=1}^{i}(1 - p_j)\right)\right\} = 0.$$

$$(B)\quad \frac{\partial S(p_i, m)}{\partial p_{LT}} = 2\sum_{i=1}^{n}\left\{C_i - m\left[1 - \prod_{j=1}^{i}(1 - p_j)\right]\right\} \left\{me^{\sum_{j=1}^{i} ln(1-p_j)}\sum_{j=1}^{i}\frac{-1}{(1 - p_j)(1 + be^{-aj})}\right\} = 0.$$

$$(C)\ \frac{\partial S(p_i,m)}{\partial a} = 2\sum_{i=1}^{n}\left\{C_i - m\left[1-\prod_{j=1}^{i}(1-p_j)\right]\right\}$$
$$\left\{me^{\sum_{j=1}^{i} ln(1-p_j)}\sum_{j=1}^{i}\frac{-p_{LT}bje^{-aj}}{(1-p_j)(1+be^{-aj})^2}\right\} = 0.$$

$$(D)\ \frac{\partial S(p_i,m)}{\partial b} = 2\sum_{i=1}^{n}\left\{C_i - m\left[1-\prod_{j=1}^{i}(1-p_j)\right]\right\}$$
$$\left\{me^{\sum_{j=1}^{i} ln(1-p_j)}\sum_{j=1}^{i}\frac{p_{LT}e^{-aj}}{(1-p_j)(1+be^{-aj})^2}\right\} = 0.$$

To solve this kind of nonlinear least square problems, we make use of a modified Levenberg–Marquardt algorithm and a finite–difference Jocobian method [13]. Besides, the least square estimators of the parameters of the HGDM with different w_i functions can be obtained similarly.

6. Growth Curves of EC_i for HGDM with Different Sensitivity Factors

In this section, we will discuss the growth curve of the mean value function EC_i for the HGDM with these two new sensitivity factors. In general, a continuous function $f(s)$ is called an exponential curve if $f'(s) > 0$ and $f''(s) < 0$ for $-\infty < s < \infty$. That is, $f'(s)$ is a decreasing function of s, and then

$$\lim_{\Delta s\to 0}\frac{f(s)-f(s-\Delta s)}{\Delta s} > \lim_{\Delta s\to 0}\frac{f(s+\Delta s)-f(s)}{\Delta s}, \quad (15)$$

where $-\infty < s < \infty$ and $\Delta s > 0$. Consider the case that a discrete function $f(s)$ is an exponential curve. Eq.(15) can be modified to be

$$\frac{f(s_i)-f(s_{i-1})}{s_i - s_{i-1}} > \frac{f(s_{i+1})-f(s_i)}{s_{i+1}-s_i}, \quad (16)$$

where $s_{i-1} < s_i < s_{i+1}$. Furthermore, assume $s_i - s_{i-1} = s_{i+1} - s_i$ for all i, and then Eq.(16) can be reduced to

$$f(s_{i+1}) + f(s_{i-1}) - 2f(s_i) < 0, \text{ for all } i.$$

A continuous function $f(s)$ is an S–shaped curve if $f'(s) > 0$ for all s and there is a S such that $f''(s) > 0$ for all $s < S$ and $f''(s) < 0$ for all $s > S$. Assuming $s_i - s_{i-1} = s_{i+1} - s_i$ for all i and by the same argument above, a discrete function $f(s)$ is an S–shaped curve if there is an integer I such that

$$f(s_{i+1}) + f(s_{i-1}) - 2f(s_i) > 0, \text{ for all } i < I \text{ and}$$
$$f(s_{i+1}) + f(s_{i-1}) - 2f(s_i) < 0, \text{ for all } i > I.$$

Similarly, assuming $s_i - s_{i-1} = s_{i+1} - s_i$ for all i, a discrete function $f(s)$ is called a Linear-exponential curve if there is an integer I such that

$$f(s_{i+1}) + f(s_{i-1}) - 2f(s_i) = 0, \text{ for all } i < I \text{ and}$$
$$f(s_{i+1}) + f(s_{i-1}) - 2f(s_i) < 0, \text{ for all } i > I.$$

In the following, we will investigate the growth curve of the mean value function EC_i for the HGDMs. Since EC_i is an increasing function of i, we thus have the following definition according to the above discussions.

Definition 1.

(i) EC_i is an exponential curve if $EC_{i+1} + EC_{i-1} - 2EC_i < 0$ for all $i > 0$;

(ii) EC_i is an S–shaped curve if there is an integer $I > 0$ such that $EC_{i+1} + EC_{i-1} - 2EC_i > 0$ for all $i < I$ and $EC_{i+1} + EC_{i-1} - 2EC_i < 0$ for all $i > I$;

(iii) EC_i is a Linear-exponential curve if there is an integer $I > 0$ such that $EC_{i+1} + EC_{i-1} - 2EC_i = 0$ for all $i < I$ and $EC_{i+1} + EC_{i-1} - 2EC_i < 0$ for all $i > I$.

Let

$$\Delta(i) = p_{i+1} - p_i - p_i p_{i+1}, \quad (17)$$

and then we have the following lemma.

Lemma 1.

(i) EC_i is an exponential curve if $\Delta(i) < 0$ for all i;

(ii) EC_i is an S–shaped curve if there is an integer I such that $\Delta(i) > 0$ for all $i < I$ and $\Delta(i) < 0$ for all $i > I$;

(iii) EC_i is a Linear–exponential curve if there is an integer I such that $\Delta(i) = 0$ for all $i < I$ and $\Delta(i) < 0$ for all $i > I$.

Proof: By Eq.(5), the value of $EC_{i+1} + EC_{i-1} - 2EC_i$ is equal to

$$m\{\prod_{j=1}^{i-1}(1-p_j)\}\{2(1-p_i) - (1-p_i)(1-p_{i+1}) - 1\}$$
$$= m\{\prod_{j=1}^{i-1}(1-p_j)\}\{p_{i+1} - p_i - p_i p_{i+1}\}.$$

Hence, by Definition 1 and $m\prod_{j=1}^{i-1}(1-p_j) > 0$, Lemma 1 can be proved. **Q.E.D.**

Lemma 2. Suppose $\Delta(i)$ is a decreasing function of i, we have:

(i) EC_i is an exponential curve if $\Delta(1) < 0$;

(ii) EC_i is an S–shaped curve if $\Delta(1) > 0$ and $\lim_{i\to\infty}\Delta(i) < 0$;

(iii) EC_i is a Linear–exponential curve if $\Delta(1) = 0$ and $\lim_{i\to\infty}\Delta(i) < 0$.

Proof: According to Lemma 1, it is obvious that Lemma 2 is true. **Q.E.D.**

In the following theorems, we investigate the properties of the mean value function EC_i of the HGDM with new sensitivity factors.

Theorem 1. Consider the case $p_i = p_{LT}(1-e^{-ai})$, we have:

(i) if $a > \ln\dfrac{1+\sqrt{1+4p_{LT}^2}}{2p_{LT}}$, $E(C_i)$ is an exponential curve;

(ii) if $0 < a < \ln\dfrac{1+\sqrt{1+4p_{LT}^2}}{2p_{LT}}$, $E(C_i)$ is an S–shaped curve;

(iii) if $a = \ln\dfrac{1+\sqrt{1+4p_{LT}^2}}{2p_{LT}}$, $E(C_i)$ is a Linear–exponential curve.

Proof: Since p_i is an exponential curve of i, we have $p_{i+1}-2p_i+p_{i-1} < 0$ and $p_ip_{i-1}-p_{i+1}p_i < 0$. Therefore,

$$\begin{aligned}&\Delta(i)-\Delta(i-1)\\&= p_{i+1}-2p_i+p_{i-1}+p_ip_{i-1}-p_{i+1}p_i < 0\end{aligned}$$

That is, $\Delta(i)$ is a decreasing function of i. Now, we want to evaluate the values of $\Delta(1)$ and $\lim_{i\to\infty}\Delta(i)$.

$$\Delta(1) = -p_{LT}e^{-3a}(e^a-1)(p_{LT}e^{2a}-e^a-p_{LT}).$$

Therefore,

$$\Delta(1)\lesseqgtr \text{ if and only if } p_{LT}e^{2a}-e^a-p_{LT}\gtreqless 0.$$

Besides, since $a > 0$ and $0 < p_{LT} \le 1$, we have

$$\lim_{i\to\infty}\Delta(i) = -p_{LT}^2 < 0.$$

By Lemma 2, Theorem 1 can be proved. **Q.E.D.**

To prove Theorem 2, we need the following conditions A1–A4.

A1: $\{0 < b \le 1,\ a > 0\}$ or $\{b > 1,\ a > \ln b\}$.

A2: $\left(\{p_{LT} \le \frac{b}{4}\}\text{ and }\{0 < a < \log\dfrac{b-\sqrt{b^2-4p_{LT}b}}{2p_{LT}}\right.$ or $\left. a > \log\dfrac{b+\sqrt{b^2-4p_{LT}b}}{2p_{LT}}\}\right)$ or $\{p_{LT} > \frac{b}{4}\}$.

A3: $\{p_{LT} < \frac{b}{4},\ 0 < \ln\dfrac{b-\sqrt{b^2-4p_{LT}b}}{2p_{LT}} < a < \ln\dfrac{b+\sqrt{b^2-4p_{LT}b}}{2p_{LT}}\}$.

A4: $\{p_{LT} \le \frac{b}{4},\ a = \ln\dfrac{b\pm\sqrt{b^2-4p_{LT}b}}{2p_{LT}}\}$.

Theorem 2. Consider the logistic learning factor $p_{LT}/(1+be^{-ai})$, we have:

(i) under Conditions A1 and A2, $E(C_i)$ is an exponential curve;

(ii) under Conditions A1 and A3, $E(C_i)$ is an S–shaped curve;

(iii) under Conditions A1 and A4, $E(C_i)$ is a Linear–exponential curve.

Proof: From Eq.(17), we have

$$\begin{aligned}&\Delta(i)-\Delta(i-1)\\&= p_{LT}\frac{D(i)}{(1+be^{-a(i+1)})(1+be^{-ai})(1+be^{-a(i-1)})},\end{aligned}$$

where $D(i)$ is equal to

$$(e^{-a}-1)be^{-2ia}(b-be^a+(p_{LT}-1)e^{ai}+(p_{LT}+1)(e^a)^{i+1}).$$

Since $a > 0$, we have

$$\begin{aligned}&\Delta(i)-\Delta(i-1) < 0 \text{ if and only if}\\&\quad b-be^a+(p_{LT}-1)e^{ai}+(p_{LT}+1)(e^a)^{i+1} > 0.\end{aligned}$$

Let $y = e^a > 1$. Since $b > 0$ and $0 < p_{LT} \le 1$, we have

$$\begin{aligned}&b-by+(p_{LT}-1)y^i+(p_{LT}+1)y^{i+1}\\&> b-by+y(y-1) \equiv g(y).\end{aligned}$$

Consider the following two cases for $g(y)$.

(i) If $0 < b \le 1$, then $g(y) > 0$ for all $y > 1$. Therefore, $\Delta(i)$ is a decreasing function of i.

(ii) If $b > 1$, then $g(y) > 0$ for $y > b$. Therefore, $\Delta(i)$ is a decreasing function of i when $b > 1$ and $a \ge \ln b$.

The above (i) and (ii) indicate that $\Delta(i)$ is a decreasing function of i under Condition A1. In the following, we will evaluate the values of $\Delta(1)$ and $\lim_{i\to\infty}\Delta(i)$. From Eq.(17), we have

$$\Delta(1) = \frac{-p_{LT}e^{-2a}}{(1+be^{-2a})(1+be^{-a})}(p_{LT}e^{2a} - be^{a} + b).$$

Hence,

$$\Delta(1)\lesseqqgtr 0 \text{ if and only if } p_{LT}e^{2a} - be^{a} + b \gtreqqless 0.$$

Let $y = e^a > 1$, $h(y) = p_{LT}y^2 - by + b$, and $\delta = b^2 - 4p_{LT}b$. In the following, we will consider three cases for δ.

Case 1. $\delta < 0$.

It is obvious $h(y) > 0$.

Case 2. $\delta = 0$.

Since there is a unique root of $h(y)$ which is equal to $b/(2p_{LT}) = 2$, we have:

$$\begin{cases} h(y) > 0, & \text{if } y \neq b/2p_{LT}; \\ h(y) = 0, & \text{if } y = b/2p_{LT}. \end{cases}$$

Case 3. $\delta > 0$.

There are two roots r_1 and r_2 of $h(y)$, where $r_1 = (b - \sqrt{b^2 - 4p_{LT}b})/2p_{LT}$ and $r_2 = (b + \sqrt{b^2 - 4p_{LT}b})/2p_{LT}$. Since $\delta > 0$ (i.e., $b > 4p_{LT}$), we have $r_2 > r_1 > 0$ and $r_2 > 2$. Moreover, since $h(1) = p_{LT} > 0$, we have:

$$\begin{cases} h(y) > 0, & \text{if } y > r_2 \text{ or } 1 < y < r_1; \\ h(y) = 0, & \text{if } y = r_1 \text{ or } r_2; \\ h(y) < 0, & \text{if } r_1 < y < r_2. \end{cases}$$

Since $a > 0$, $b > 0$, and $0 < p_{LT} \le 1$, we have

$$\lim_{i\to\infty}\Delta(i) = -p_{LT}^2 < 0.$$

By Lemma 2 and the simple arrangement, Theorem 2 can be readily proved. **Q.E.D.**

7. Application to Two Sets of Real Data

In this section, we will apply previously published data sets to compare the HGDM with various learning factors listed in Table 1 with the four SRGMs mentioned in Section 2 in terms of the goodness-of-fit. The sum of squares of errors $SSE = \sum_{i=1}^{n}(C_i - \widehat{C}_i)^2$ is adopted as the evaluation criterion [14] in the comparison of goodness-of-fit of the models. Besides, if the actual cumulative number of detected faults during the test and after the test is known (denoted by M_a), the accuracy of estimation E defined in the following is also chosen as a criterion [14]:

$$E = \left|\frac{M_a - \widehat{m}}{M_a}\right|. \tag{18}$$

The parameters of the HGDM with various learning factors are estimated by Least Square Method (LSM) as described in Section 5, and the parameters of the four SRGMs are estimated by both Maximum Likelihood Estimation (MLE) and LSM .

Table 1: Various w_i functions of HGDM.

$w_i = mp_i$ for the test instance t_i	Eq.
$w_i = m(ai + b)$	(a)
$w_i = mp_{LT}(1 - e^{-ai})$	(b)
$w_i = mp_{LT}\frac{1}{1+be^{-ai}}$	(c)
$w_i = mu_i(ai + b)$	(d)
$w_i = mp_{LT}(1 - e^{-u_iai})$	(e)
$w_i = mp_{LT}\frac{1}{1+be^{-u_iai}}$	(f)

7.1. First Data Set

The first real data set is the test-and-debug data of a PL/I database application program [3] listed in Table 2. The data is reported once per week, and consequently a test instance is "a week of observation". Table 2 shows the execution time u_i, the number of newly detected faults x_i, and the cumulative numbers of faults C_i for the 19 test instances. In this data set, M_a equals 358; that is, 30 faults are still detected after the test-and-debug process.

Table 3 shows the estimated parameters of the HGDM with Eq.(d), Eq.(e), and Eq.(f), respectively. Table 4 shows the estimated initial faults $\widehat{m}$, SSE, and E for each model. From Table 4, it shows that the HGDM with Eq.(f) fits the data best because $\widehat{m}$ (356.3) is very close to the real observed number of initial faults M_a (358) and SSE (1709.7) is the smallest. Besides, SSE and E of the HGDM with Eq.(e) are less than those of the HGDM with Eq.(d). The estimated growth curve $\widehat{C}_i$ for the HGDM with Eq.(f) is plotted in Fig.2. Applying the Kolmogorov-Smirnov test [15] to the HGDM with Eq.(f) shows that the model adequately fits the data at the 5% level of significance (the value of the Kolmogorov-Smirnov test statistics is 0.106). Hence, we can conclude that the HGDM with the logistic learning factor (i.e., the HGDM with Eq.(f)) fits the data very satisfactorily and gives a better fit in this experiment.

7.2. Second Data Set

The second set of real data is the test-and-debug data of a software system [16]. The cumulative numbers of faults C_i for the 24 test instances and the newly detected faults x_i are gathered week by week. Since the information of u_i is unavailable, u_i is assumed to be a constant for all i. Besides, the actual initial faults are unknown (i.e., M_a is unknown), and then E cannot be used for comparison.

Table 5 shows the estimated parameters of the HGDM with Eq.(a), Eq.(b), and Eq.(c), respectively. Table 6 shows the estimated initial faults $\widehat{m}$ and SSE for each model. From Table 6, the HGDM with Eq.(c) has the smallest SSE value (34314.3). Besides, SSE of the HGDM with Eq.(b) is less than that of the HGDM with Eq.(a). The estimated growth curve $\widehat{C}_i$ for the HGDM with Eq.(c) is plotted in Fig.3. The value of the Kolmogorov–Smirnov test statistics for the HGDM with Eq.(c) is 0.11 which shows that the model fits the data at the 5% level of significance. From the above discussion, we know that the HGDM with the logistic learning factor (i.e., the HGDM with Eq.(c)) fits the data better than others, and also gives a preferable fit in this experiment.

8. Conclusions

In this paper, we have proposed two new w_i functions based on the exponential and the S–shaped learning curves, respectively. The HGDM with different w_i functions, the Goel-Okumoto model, the Delayed S–shaped model, the Inflection S–shaped model, and the logarithmic poisson execution time model are compared by using two sets of real data. Experimental results show that the HGDM with the logistic learning factor fits these data sets satisfactorily in terms of the sum of squares of errors criterion. Besides, we also show that both of these two new sensitivity factors can derive the exponential and the S–shaped growth curves of the cumulative number of discovered faults under certain conditions.

There currently exists no general software reliability growth model applicable to all applications. A suitable model has to be selected among many candidates for each application. The HGDM with the logistic learning factor is certainly one of the candidates.

Table 2: First data set.

Week	x_i	C_i	u_i	Week	x_i	C_i	u_i
1	15	15	2.450	11	27	233	2.880
2	29	44	2.450	12	22	255	1.440
3	22	66	1.960	13	21	276	3.260
4	37	103	0.980	14	22	298	3.840
5	2	105	1.680	15	6	304	3.840
6	5	110	3.370	16	7	311	2.300
7	36	146	4.210	17	9	320	1.760
8	29	175	3.370	18	5	325	1.990
9	4	179	0.960	19	3	328	2.990
10	27	206	1.920				

x_i: Newly detected faults.
C_i: Cumulative number of real observed detected faults.
u_i: Execution time.

Table 3: Estimated parameters of HGDMs for the first data set.

w_i	$\widehat{m}$	$\widehat{a}$	$\widehat{b}$	$\widehat{p}_{LT}$
Eq.(d)	387.7094	0.0017	0.0224	
Eq.(e)	385.1323	0.2797		0.0634
Eq.(f)	356.2711	0.0340	14.661	0.8475
Initial	358			

Table 4: Comparison results for the first data set.

Model	$\widehat{m}$	$SSE^{(1)}$	$E^{(2)}$
HGDM with Eq.(d)	387.7	2624.2	8.30%
HGDM with Eq.(e)	385.1	2113.6	7.56%
HGDM with Eq.(f)	356.3	1709.7	0.47%
Goel-Okumoto, MLE	455.4	3932.3	27.21%
Goel-Okumoto, LSM	562.8	2997.2	57.21%
Delayed S–$s^{(3)}$, MLE	351.4	5673.2	1.84%
Delayed S–s, LSM	353.5	5523.2	1.26%
Inflec. S–$s^{(4)}$, MLE	347.2	3406.2	3.02%
Inflec. S–s, LSM	389.1	2537.1	8.69%
LOGM$^{(5)}$, MLE	***$^{(6)}$	27541.4	***
LOGM, LSM	***	3253.5	***

(1) The sum of squares of errors.
(2) The accuracy of estimation and $M_a = 358$.
(3) Delayed S–s: Delayed S–shaped model.
(4) Inflec. S–s: Inflection S–shaped model.
(5) LOGM: Logarithmic poisson execution time model.
(6) Concept of the number of initial faults for LOGM does not exist.

Table 5: Estimated parameters of HGDMs for the second data set.

w_i	$\widehat{m}$	$\widehat{a}$	$\widehat{b}$	$\widehat{p}_{LT}$
Eq.(a)	2183.3	0.0104	0.0197	
Eq.(b)	2300.8	0.1206		0.1602
Eq.(c)	2313.4	0.3362	5.5395	0.1363
Initial	unknown			

Table 6: Comparison results for the second data set.

Model	$\widehat{m}$	*SSE*
HGDM with Eq.(a)	2183.3	44259.1
HGDM with Eq.(b)	2300.8	42890.9
HGDM with Eq.(c)	2313.4	34314.4
Goel-Okumoto, MLE	3306.3	311935.1
Goel-Okumoto, LSM	4108.9	252654.7
Delayed S-s, MLE	2386.8	49919.4
Delayed S-s, LSM	2381.5	49865.2
Inflec. S-s, MLE	2304.4	52011.7
Inflec. S-s, LSM	2205.9	40946.7
LOGM, MLE	***	1490567.0
LOGM, LSM	***	279053.2

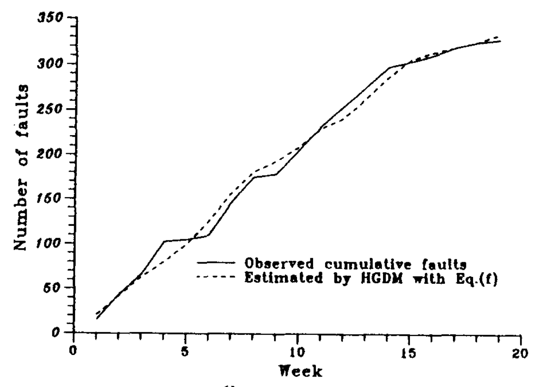

Fig.2 Fitness of $\widehat{C}_i's$ to $C_i's$ for the first data set.

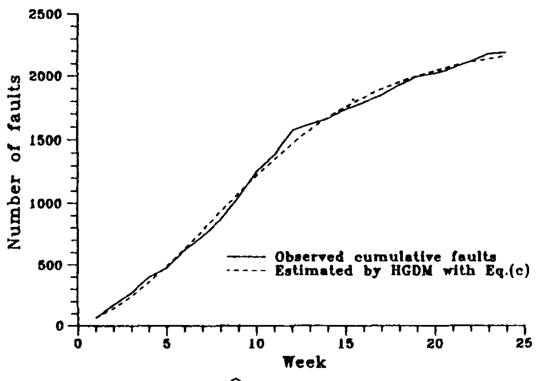

Fig.3 Fitness of $\widehat{C}_i's$ to $C_i's$ for the second data set.

Acknowledgment. We would like to express our gratitude for the support of the National Science Council, Taiwan, R.O.C., under Grant NSC 82-0408-E002-021. Reviewers' comments are also highly appreciated.

References

[1] A. L. Goel and K. Okumoto, "Time-Dependent Error Detection Rate Model for Software Reliability and Other Performance Measures," *IEEE Trans. Reliability*, Vol. R-28, No. 3, pp. 206-211, August 1979.

[2] S. Yamada, M. Ohba, and S. Osaki, "S-Shaped Software Reliability Growth Models and their Applications," *IEEE Trans. Reliability*, Vol. R-33, No. 4, pp. 289-292, October 1984.

[3] M. Ohba, "Software Reliability Analysis Models," *IBM J. Res. Develop.*, Vol. 28, No. 4, pp. 428-443, July 1984.

[4] J. D. Musa and K. Okumoto, "A Logarithmic Poisson Execution Time Model for Software Reliability Measurement," *Proc. 7th Int. Conf. Software Engineering*, pp. 230-238, 1984.

[5] Y. Tohma, K. Tokunaga, S. Nagase, and Y. Murata, "Structural Approach to the Estimation of the Number of Residual Software Faults Based on the Hyper-Geometric Distribution," *IEEE Trans. Software Engineering*, Vol. 15, No. 3, pp. 345-355, March 1989.

[6] Y. Tohma, R. Jacoby, Y. Murata, and M. Yamamoto, "Hyper-Geometric Distribution Model to Estimate the Number of Residual Software Faults," *Proc. COMPSAC-89*, Orlando, pp. 610-617, September 1989.

[7] Y. Tohma, H. Yamano, M. Ohba, and R. Jacoby, "Parameter Estimation of the Hyper-Geometric Distribution Model for Real Test/Debug Data," *Proc. Int. Symposium on Software Reliability Engineering*, Austin, Texas, May 17-18, pp. 28-34, 1991.

[8] Y. Tohma, H. Yamano, M. Ohba, and R. Jacoby, "The Estimation of Parameters of the Hypergeometric Distribution and Its Application to the Software Reliability Growth Model," *IEEE Trans. Software Engineering*, Vol. SE-17, No. 5, pp. 483-489, May 1991.

[9] R. Jacoby and Y. Tohma, "The Hyper–Geometric Distribution Software Reliability Growth Model (HGDM): Precise Formulation and Applicability," *Proc. COMPSAC90*, Chicago, pp. 13–19, October 1990.

[10] R. Jacoby and Y. Tohma, "Parameter Value Computation by Least Square Method and Evaluation of Software Availability and Reliability at Service–Operation by the Hyper–Geometric Distribution Software Reliability Growth Model (HGDM)," *Proc. 13th Int. Conf. Software Engineering*, pp. 226–237, 1991.

[11] J. D. Musa, A. Iannino, and K. Okumoto, "Software Reliability — Measurement, Prediction, Application," McGraw–Hill, INc., 1987.

[12] W. R. Dillion and M. Goldstein, "Multivariate Analysis Methods and Applications," New York: Wiley, 1984.

[13] J. E. Dennis and Robert B. Schnabel, "Numerical Methods for Unconstrained Optimization and Nonlinear Equations," Prentice–Hall, Englewood Cliffs, New Jersey, 1983.

[14] S. Yamada and S. Osaki, "Software Reliability Growth Modeling: Models and Applications," *IEEE Trans. Reliability*, Vol. R–11, No. 12, pp. 1431–1437, December 1985.

[15] W. J. Conover, "Practical Nonparametric Statistics," New York: Wiley, 1980.

[16] A. L. Goal, "Software Reliability Modeling and Estimation Technique," Final Technical Report RADC–TR–82–263, 1983.