

ソフトウェアエラー発見過程の確率モデルとその応用

山田 茂 尾崎俊治
(広島大学大学院工学研究科) (広島大学工学部)

1. はじめに

ハードウェアコストに対するソフトウェアコストの比率が高まるにつれ、高信頼性ソフトウェアを作り出す重要性が高まってきている。ソフトウェアの信頼性評価は、ソフトウェアシステムの性能を測定し予測するために非常に重要である。その主要な活動の1つは、開発中のソフトウェアシステムのテストをいつ打ち切るべきかを予測し、システム内に残存するエラー数を推定し、あるいはエラー発見過程における信頼度成長を記述するのに有用なソフトウェア信頼性モデルを開発することである。ソフトウェアの信頼性は、ソフトウェア故障の性質と発生頻度により定義される。ここで、ソフトウェアが正確に所期の動作をしないときソフトウェアのエラーがあるといい、そのエラーが表面に出ることをソフトウェア故障と定義する(Myers [11] 参照)。

近年、ソフトウェアシステムの信頼度およびシステム内のエラー数を予測するためのモデルを開発することに関心が寄せられ、多くのソフトウェア故障過程を記述する信頼性モデルが提案されてきた([3, 4, 6-12, 14-17] 参照)。我々が興味のあるのは、ソフトウェア開発段階におけるソフトウェアエラー発見過程のモデル化である。特に、Goel and Okumoto [5] および Goel [6] は、ハードウェアの信頼度成長を表わすのに用いられてきた非同次ポアソン過程(Nonhomogeneous Poisson process (NHPP))を適用してソフトウェアエラー発見モデルを議論した。このモデルは、時間に依存するエラー発見率を考慮したモデルであり、NHPPの平均値関数により特徴づけられる発見エラー数の成長曲線は指数分布関数形をしている。しかし一般には、観測データの発見エラー数の成長曲線は指数分布関数形よりもむしろS字形を示すことも多い。実際、Goel and Okumoto [5] において解析されたデータの発見エラー数の成長曲線はS字形を示している。

本稿では、NHPPに基づき観測データに対する発見エラー数の成長曲線がS字形を示す場合に適切なソフトウェアエラー発見モデルを提案する。まず、ソフトウェアシステムの性能を評価するのに有用な定量的尺度を導き、我々のモデルにおける未知パラメータの推定方法について議論する。つぎに、実際のソフトウェアエラーデータを解析して我々のモデルの適用性を考察する。ここではさらに、Goel and Okumoto [5] および Goel [6] のモデルと比較することにより、我々のモデルの適合性の良さを示す。

2. モデル

システム内のエラーによりソフトウェア故障を引き起こすソフトウェアシステムを考える。 $\{N(t), t \geq 0\}$ を時刻 t までに発見される総エラー数を表わす計数過程とする。Goel and Okumoto [5] および Goel [6] は、

$$Pr\{N(t)=n\} = \frac{[m(t)]^n}{n!} e^{-m(t)} \quad (n=0, 1, 2, \dots), \quad (1)$$

$$m(t) = a(1 - e^{-bt}) \quad a, b > 0, \quad (2)$$

というNHPPにより記述されるソフトウェアエラー発見モデルを提案した。こ

で、 a は究極的に発見される総期待ソフトウェアエラー数、 b はエラー1個当りのエラー発見率である。このモデルにおける発見エラー数の成長曲線は、図1の(A)で示されるような指数分布関数形をしている。しかし、一般には図1の(B)で示されるようなS字形を示す場合もよくある。発見エラー数のS字形成長曲線は、初期には時間が経過する割には発見エラー数は少ないが、その後急速に増加し、終期には飽和状態に近づいていくという性質をもつ。このような場合、(2)式の $m(t)$ をもつNHPPモデルは、観測データへの適合性が悪い。そこで、時間 t に対してS字形を示す平均値関数

$$M(t) = a(1 - (1 + bt)e^{-bt}) \quad a, b > 0, \quad (3)$$

をもつNHPPモデルを提案する。ここで、 a は究極的に発見される総期待エラー数を表す。(3)式から、1個当りのエラー発見率は、

$$v(t) \equiv \lim_{\Delta t \rightarrow 0} \frac{M(t+\Delta t) - M(t)}{(a - M(t))\Delta t} = b(1 - \frac{1}{1 + bt}), \quad (4)$$

となる。 $v(\infty) = b$ であるから、 b は定常状態における1個当りのエラー発見率を表す。時刻 t におけるエラー発見率を表すNHPPの強度関数は

$$\mu(t) \equiv dM(t)/dt = ab^2te^{-bt}, \quad (5)$$

で与えられる。 $M(t)$ の一般的な形を図2に示す。 $M(t)$ の変曲点は $(1/b, 0.2642a)$ である。 $M(t)$ および $m(t)$ の間には、つぎのような興味ある関係式が成立する。

$$M(0) = m(0) = 0, \quad (6)$$

$$M(\infty) = m(\infty) = a, \quad (7)$$

$$dM(t)/dt = b(m(t) - M(t)), \quad (8)$$

$$dM(t)/dt = b(a - m(t)). \quad (9)$$

さらに(3)式から、 $M(t)/a$ は形状パラメータおよび尺度パラメータが与えられ2お

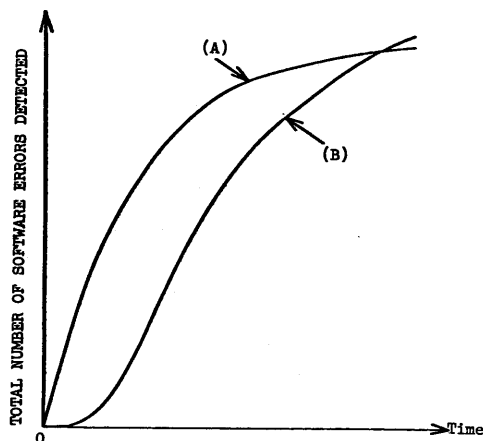


Fig. 1. The growth curves of software errors detected for the observed data.

図1. 観測データに対する発見ソフトウェアエラー数の成長曲線。

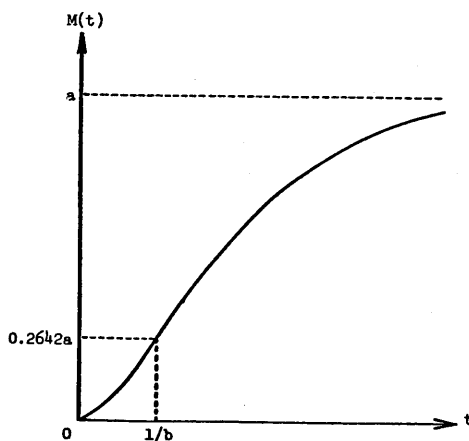


Fig. 2. The mean value function $M(t)$ with an S-shaped growth curve.

図2. S字形成長曲線をもつ平均値関数 $M(t)$ 。

よび b のガンマ分布を表わす。そこで、(3)式の平均値関数をもつ NHPP モデルをガンマ型成長モデル (Gamma Type Growth Model (GTGM)) と定義し、(2)式の平均値関数をもつ NHPP モデルを指数型成長モデル (Exponential Type Growth Model (ETGM)) と定義する。

3. ソフトウェアの性能評価の定量的尺度

GTGM に対して、ソフトウェアシステムの性能評価のための各定量的尺度を導出する。

3.1. 総発見ソフトウェアエラー数の分布

$N(t)$ の分布は平均値が $M(t)$ のポアソン分布である。その極限分布は、

$$\lim_{t \rightarrow \infty} \Pr\{N(t)=n\} = \frac{a^n}{n!} e^{-a} \quad (n=0, 1, 2, \dots), \quad (10)$$

となり、ETGM の結果と一致する。(10)式はシステムが長時間テストされにときの総発見エラー数も平均値 a のポアソン分布に従うことを意味している。

3.2. 残存ソフトウェアエラー数の分布

時刻 t においてシステム内に残存するエラー数 $\bar{N}(t) \equiv N(\infty) - N(t)$ の期待値および分散を求めると、それぞれ

$$E\{\bar{N}(t)\} = a(1+bt)e^{-bt}, \quad (11)$$

$$\text{Var}\{\bar{N}(t)\} \equiv E\{[\bar{N}(t) - E\{\bar{N}(t)\}]^2\} = a(1+bt)e^{-bt}, \quad (12)$$

を得る。また、時刻 t までは n_d 個のエラーが発見されたとする。このとき、 $N(t) = n_d$ が与えられたときの $\bar{N}(t)$ の条件付き確率および期待値は、それぞれ

$$\Pr\{\bar{N}(t)=x | N(t)=n_d\} = \frac{a^{x+n_d}}{(x+n_d)!} e^{-a}, \quad (13)$$

$$E\{\bar{N}(t) | N(t)=n_d\} = (a-n_d)\left\{1 - \sum_{u=0}^{n_d-1} \frac{a^u}{u!} e^{-a}\right\} + \frac{a^{n_d}}{(n_d-1)!} e^{-a}, \quad (14)$$

となる。条件付き分布(13)は、製品としてのソフトウェアの出荷基準として有用である。

3.3. ソフトウェア信頼度

ソフトウェア故障間、すなわちエラー発見間の時間間隔を表わす確率変数列を $\{X_k, k=1, 2, \dots\}$ と定義する。したがって、 $S_k = \sum_{i=1}^k X_i$ ($k=1, 2, \dots$) は k 番目のエラー発見時刻を表わす。このとき、 $S_{k-1} = A$ が与えられたときの X_k の条件付き信頼度関数、すなわち $(A, A+t]$ でソフトウェア故障の起こらない確率は

$$R_{X_k | S_{k-1}}(t | A) = \exp[-a\{(1+bt)e^{-bt} - (1+b(t+A))e^{-b(t+A)}\}], \quad (15)$$

により与えられる。

4. パラメータの推定

$M(t)$ における未知パラメータ a および b を推定するために、利用可能なデータに依りつぎの2つの方法について述べる。

4.1. ソフトウェア故障時刻のデータが与えられている場合

ソフトウェア故障時刻、すなわちエラー発見時刻のデータ $A = (A_1, A_2, \dots, A_n)$ が観測されたとする。 $\{S_1, S_2, \dots, S_n\}$ の同時確率密度関数を求めると

$$f_{S_1, S_2, \dots, S_n}(A_1, A_2, \dots, A_n) = \exp[-M(A_n)] \prod_{k=1}^n ab^2 A_k e^{-bA_k}, \quad (16)$$

となる。△が与えられたときの未知パラメータ a および b に対する対数尤度関数は(16)式から

$$L(\Delta; a, b) = n \cdot \ln a + 2n \cdot \ln b + \sum_{k=1}^n (\ln \Delta_k - b \Delta_k) - a(1 - (1 + b \Delta_n) e^{-b \Delta_n}), \quad (17)$$

により与えられる。したがって、2つの未知パラメータ a および b の最尤推定値 $\hat{a}$ および $\hat{b}$ は、 $\partial L / \partial a = \partial L / \partial b = 0$ より

$$\frac{n}{a} = 1 - (1 + b \Delta_n) e^{-b \Delta_n}, \quad \frac{2n}{b} = \sum_{k=1}^n \Delta_k + a b \Delta_n^2 e^{-b \Delta_n}, \quad (18)$$

の同時方程式を数値的に解いて得られる。

4.2. 一定時刻までの累積ソフトウェア故障数のデータが与えられている場合

ある一定時刻までの累積ソフトウェア故障数、すなわち時刻 t_k までに発見されたソフトウェアエラー数が y_k であるようなデータ $(t_k, y_k) (k=1, 2, \dots, n)$ が観測されたとする。 $\{N(t_1)=y_1, N(t_2)=y_2, \dots, N(t_n)=y_n\}$ の同時確率を求めると

$$Pr\{N(t_1)=y_1, N(t_2)=y_2, \dots, N(t_n)=y_n\} = \prod_{k=1}^n \frac{[M(t_k) - M(t_{k-1})]^{y_k - y_{k-1}}}{(y_k - y_{k-1})!} e^{-M(t_n)}, \quad (19)$$

となる。ここで、 $t_0=0$ および $y_0=0$ とする。 $\underline{t}=(t_1, t_2, \dots, t_n)$ および $\underline{y}=(y_1, y_2, \dots, y_n)$ とすると、 $(\underline{t}, \underline{y})$ が与えられたときの未知パラメータ a および b に対する対数尤度関数は(19)式より

$$L(\underline{t}, \underline{y}; a, b) = \sum_{k=1}^n (y_k - y_{k-1}) \ln a + \sum_{k=1}^n (y_k - y_{k-1}) \ln [(1 + b t_k) e^{-b t_k}] - (1 + b t_n) e^{-b t_n} - \sum_{k=1}^n \ln (y_k - y_{k-1})! - a(1 - (1 + b t_n) e^{-b t_n}), \quad (20)$$

により与えられる。したがって、2つの未知パラメータ a および b の最尤推定値 $\hat{a}$ および $\hat{b}$ は、 $\partial L / \partial a = \partial L / \partial b = 0$ から

$$y_n = a(1 - (1 + b t_n) e^{-b t_n}), \quad a t_n^2 e^{-b t_n} = \frac{n}{k=1} \frac{(y_k - y_{k-1})(t_k^2 e^{-b t_k} - t_{k-1}^2 e^{-b t_{k-1}})}{[(1 + b t_{k-1}) e^{-b t_{k-1}} - (1 + b t_k) e^{-b t_k}]}, \quad (21)$$

の同時方程式を数値的に解いて得られる。

大なる n に対して最尤推定値 $\hat{a}$ および $\hat{b}$ は漸近的に2変数正規分布

$$N\left(\begin{pmatrix} \hat{a} \\ \hat{b} \end{pmatrix}, \underline{V}\right), \quad (22)$$

に従うことが知られている (Cox and Hinkley [1] 参照)。ここで、分散-共分散行列 $\underline{V}$ は Fisher の情報行列 $\underline{I}$ の逆行列として与えられ、(20)式から

$$\underline{I} = \begin{bmatrix} i_{aa} & i_{ab} \\ i_{ba} & i_{bb} \end{bmatrix}, \quad (23)$$

$$i_{aa} = -E\left[\frac{\partial^2 L}{\partial a^2}\right], \quad i_{ab} = i_{ba} = -E\left[\frac{\partial^2 L}{\partial a \partial b}\right], \quad i_{bb} = -E\left[\frac{\partial^2 L}{\partial b^2}\right], \quad (24)$$

である。

5. ソフトウェアエラーデータの解析

観測データが $\Delta=(\Delta_1, \Delta_2, \dots, \Delta_n)$ である場合および $(\underline{t}, \underline{y})=(t_k, y_k) (k=1, 2, \dots, n)$ である場合について、実際のソフトウェアエラーデータを解析する。また、解析結果をもとに GTGM および ETGM の観測データに対する適合性の比較を行なう。

5.1. 観測データが Δ である場合のデータ解析

Goel and Okumoto [5] は、Naval Tactical Data System (NTDS) の中核を成すフン

コンピュータシステムの大規模ソフトウェアシステムの開発期間中に得られたソフトウェアエラーデータを解析した。このエラーデータ $\Delta = (\Delta_1, \Delta_2, \dots, \Delta_{26})$ はテスト段階に先立つ生産 (production) 段階で観測された。彼らは、ETGMに基づき $m(t)$ を

$$\hat{m}(t) = 33.99(1 - e^{-0.00579t}), \quad (25)$$

と推定した。実際の観測データおよび $\hat{m}(t)$ を図3に示す。図3から、観測データの発見エラー数の成長曲線はS字形を呈していることがわかる。そこで、ETGMよりもむしろGTGMの方が観測データに適合性のあることが期待される。これを確認するために、GTGMを図3に示された実際の観測データに適用し解析を行う。

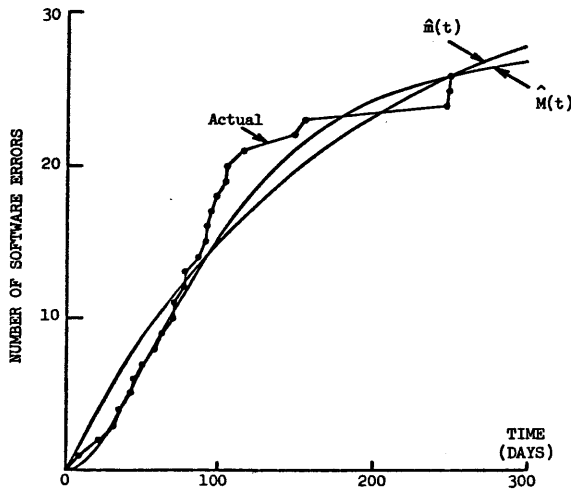


Fig. 3. The NTDS data and the estimated mean value functions $M(t)$ and $\hat{m}(t)$.

図3. NTDSデータおよび推定された平均値関数 $M(t)$ と $\hat{m}(t)$.

26個のデータを用いて(18)式を数値的に解いて最尤推定値 $\hat{\alpha}$ および $\hat{\beta}$ を求めると、推定された平均値関数は

$$\hat{M}(t) = 27.49(1 - (1 + 0.0186t)e^{-0.0186t}), \quad (26)$$

により与えられ、90%信頼限界とともに図4に示した。さらに、(15)式から $\Delta_{26} = 250$ (日)より測定される条件付き信頼度関数を求めると

$$R_{X_{27}|S_{26}}(t|250) = \exp[-27.49\{(1 + 0.0186(250))e^{-0.0186(250)} - (1 + 0.0186(t+250))e^{-0.0186(t+250)}\}], \quad (27)$$

となり、比較のためETGMの結果とともに図5に示した。

ここで、実際の観測データに対するGTGMおよびETGMの適合性の比較を行なう。比較基準として、ソフトウェア故障発生起時刻すなわちエラー発見時刻 Δ_k

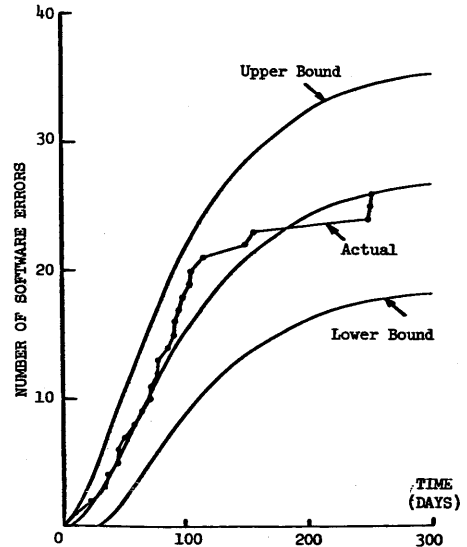


Fig. 4. The estimated mean value function $M(t)$ and the 90% s-confidence bounds for the NTDS data.

図4. 推定された平均値関数 $M(t)$ および90%信頼限界(NTDSデータ).

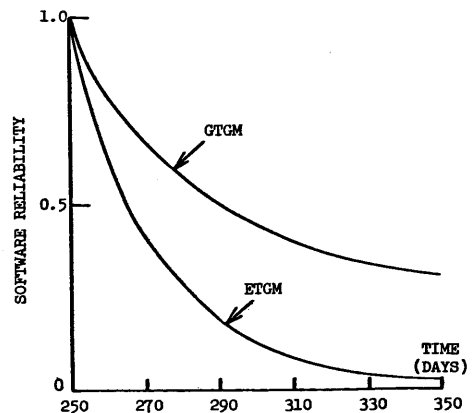


Fig. 5. The software reliability functions for the GTGM and the ETGM (NTDS data).

図5. GTGMおよびETGMに基づくソフトウェア信頼度の比較(NTDSデータ).

までの実際の総発見エラー数 k と推定された総発見エラー数 $\hat{k}$ の偏差二乗和

$$F = \sum_{k=1}^{26} (k - \hat{k})^2, \quad (28)$$

を採用する。ここで、GTGMに対しては $\hat{k} = \hat{M}(A_k)$ 、ETGMに対しては $\hat{k} = \hat{m}(A_k)$ である。したがって、GTGMに対して $F = 71.18$ 、ETGMに対して $F = 129.72$ を得る。この基準によれば、GTGMの方がETGMより実際の観測データに対して適合性は良いと言える。さらに、GTGMの実際の観測データに対するKolmogorov-Smirnov適合度検定を行なうと、有意水準5%でNHPP

$$P\{N(t) = n\} = \frac{[27.49\{1 - (1 + 0.0186t)e^{-0.0186t}\}]^n}{n!} \exp[-27.49\{1 - (1 + 0.0186t)e^{-0.0186t}\}], \quad (29)$$

に従うことが確認できた。

5.2. 観測データが(5, 4)である場合のデータ解析

日本IBM社の製品保証グループにより、図6に示した発見エラー数の成長曲線がS字形を呈する10組のソフトウェアエラーデータ(t_k, y_k) ($k=1, 2, \dots, 10$)が得られた(藤田他[16]参照)。この実験は、構造化プログラミングのためのマクロを用いてコード化された約3,000 ステップのアセンブラによるオンライン端末制御プログラムの開発中に行なわれたものである。このデータをOTCPデータと呼ぶことにする。ここで、GTGMに基づく解析を行なう。

上記10組のOTCPデータを用いて(21)式を数値的に解いて最尤推定値 $\hat{\alpha}$ および $\hat{\beta}$ を求めると、推定された平均値関数は

$$\hat{M}(t) = 37.91(1 - (1 + 0.3118t)e^{-0.3118t}), \quad (30)$$

により与えられ、90%信頼限界とともに図6に示した。前項と同様に実際の観測データに対する(30)式のKolmogorov-Smirnov適合度検定を行なうと、有意水準5%で十分な適合性があることを確認できた。また、実際の観測データに対するGTGMおよびETGMの適合性を比較するために、ETGMに基づく推定された平均値関数を求めると

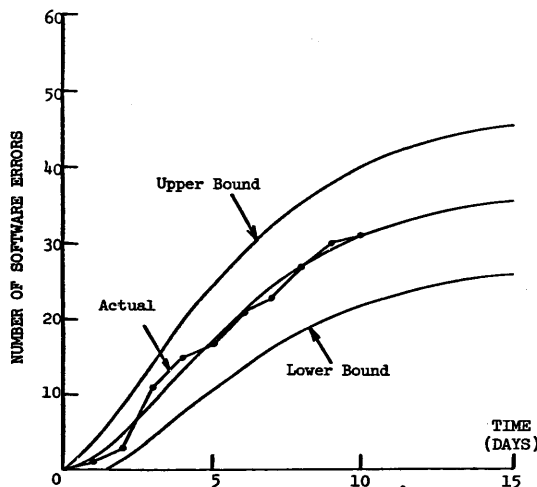


Fig. 6. The estimated mean value function $\hat{M}(t)$ and the 90% s-confidence bounds for the OTCP data.

図6. 推定された平均値関数 $\hat{M}(t)$ および90%信頼限界(OTCPデータ)。

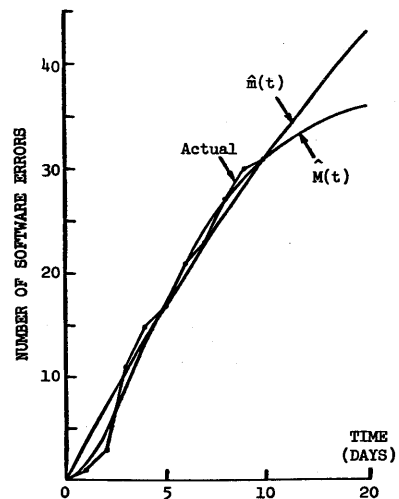


Fig. 7. The OTCP data and the estimated mean value functions $\hat{M}(t)$ and $\hat{m}(t)$.

図7. OTCPデータおよび推定された平均値関数 $\hat{M}(t)$ と $\hat{m}(t)$ 。

$$\hat{m}(t) = 109.46(1 - e^{-0.0333t}), \quad (31)$$

を得る。実際の観測データ、 $\hat{M}(t)$ 、および $\hat{m}(t)$ を図7に示した。適合性の比較基準として、一定時刻 t_k までの総見された実際の総発見エラー数 N_k と推定された総発見エラー数 $\hat{N}_k$ の偏差二乗和

$$S = \sum_{k=1}^{10} (N_k - \hat{N}_k)^2, \quad (32)$$

を採用する。ここで、GTGMに対しては $\hat{N}_k = \hat{M}(t_k)$ 、ETGMに対しては $\hat{N}_k = \hat{m}(t_k)$ である。したがって、GTGMに対して $S=12.57$ 、ETGMに対して $S=31.46$ を得る。この基準によれば、GTGMの方がETGMよりも実際の観測データに対して適合性は良いと言える。

さて、実際の観測データ $(t_k, y_k) (k=1, 2, \dots, 10)$ に対する最尤推定値 $\hat{a}$ および $\hat{b}$ の漸近的性質について議論する。(22), (23), および(24)式から最尤推定値 $\hat{a}=37.91$ および $\hat{b}=0.3118$ を適用して推定された分散-共分散行列を求めると

$$\hat{\Sigma} = \begin{bmatrix} 67.6835 & -0.3335 \\ -0.3335 & 5.2147 \times 10^{-3} \end{bmatrix}, \quad (33)$$

となる。したがって、最尤推定値 $\hat{a}$ および $\hat{b}$ の相関係数は

$$\hat{\rho}(\hat{a}, \hat{b}) = -0.5613, \quad (34)$$

により与えられる。これは、 $\hat{a}$ および $\hat{b}$ の間には負の相関があり、最初にシステム内に存在する推定される総エラー数が多ければ多いほど定常状態における1個当りの推定されるエラー発見率は小さくなることを意味する。この漸近的性質を用いて期待残存ソフトウェアエラー数およびソフトウェア信頼度の区間推定を行なうことができる。(11)式から推定された期待残存ソフトウェアエラー数は、

$$E\{\hat{N}(t)\} = 37.91(1 + 0.3118t)e^{-0.3118t}, \quad (35)$$

であり、90%信頼限界および実際の残存エラー数とともに図8に示した。一方、 $A=10$ (日)からの推定された条件付き信頼度関数は、(15)式から

$$R(t|10) = \exp[-37.91\{(1+0.3118(10))e^{0.3118(10)} - (1+0.3118(t+10))e^{-0.3118(t+10)}\}], \quad (36)$$

となる。例えば、 $t=11$ (日)でのソフトウェア信頼度は0.2305である。 $E\{\hat{N}(t)\}$ と同様に、推定された $R(t|10)$ およびその90%信頼限界を図9に示した。

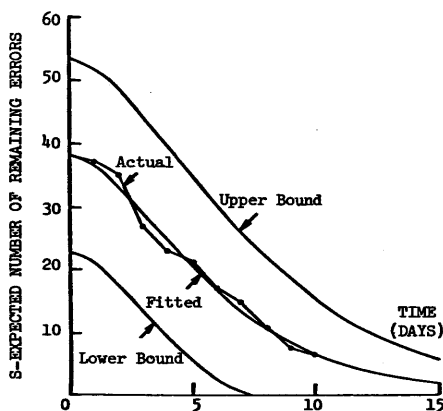


Fig. 8. The s-expected number of remaining errors at time t and the 90% s-confidence bounds for the OTCP data.

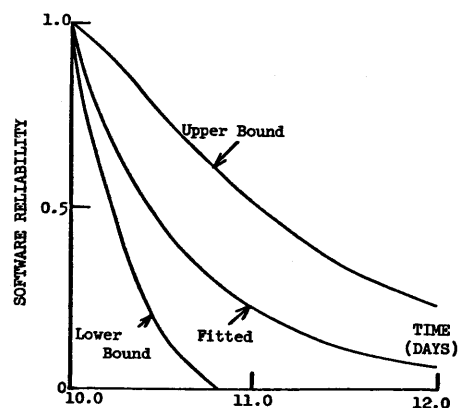


Fig. 9. The software reliability function and the 90% s-confidence bounds for the OTCP data.

図8. 期待残存ソフトウェアエラー数および90%信頼限界(OTCPデータ)。

図9. ソフトウェア信頼度関数および90%信頼限界(OTCPデータ)。

6. おわりに

本稿では、平均値関数 $M(t) = a(1 - (1 + bt)e^{-bt})$ をもつ NHPP として記述されるソフトウェアエラー発見過程モデルについて議論した。第5節の例に見られるように、ソフトウェアの開発期間中、観測データの発見エラー数の成長曲線はS字形を呈することがよくある。このような場合、GTGMの方がETGMよりも観測データに対する適合性が良いということが示された。

謝 辞

最後に、貴重なソフトウェアエラーデータを提供して頂きました日本アイ・ビー・エム(株)製品保証技術担当 篠田東吾氏ならびに同所属 大場 亮氏に深く感謝の意を表します。

参考文献

- [1] D.R. Cox, D.V. Hinkley, Theoretical Statistics, Chapman and Hall, London 1974.
- [2] D.R. Cox, P.A.W. Lewis, The Statistical Analysis of Series of Events, Methuen, London 1966.
- [3] J.W. Duran, J.J. Wiorkowski, "Capture-recapture sampling for estimating software error content," IEEE Trans. Software Engineering, vol SE-7, 1981 Jan, pp 147-148.
- [4] W. Freiberger (ed.), Statistical Computer Performance Evaluation, Academic Press, New York 1972.
- [5] A.L. Goel, K. Okumoto, "Time-dependent error-detection rate model for software reliability and other performance measures," in [8], pp 206-211.
- [6] A.L. Goel, "Software error detection model with applications," J. Systems and Software, vol 1, 1980, pp 243-249.
- [7] 伊土誠一他, "「Capture-recapture法」による潜在バグの推定法ともの応用," 情報処理学会研究会 ソフトウェア工学 19-1, 1981年7月, pp 1-10.
- [8] M. Lipow (ed.), "Special issue on software reliability," IEEE Trans. Reliability, vol R-28, 1979 Aug, pp 178-253.
- [9] B. Littlewood, J.L. Verrall, "A Bayesian reliability growth model for computer software," J. Roy. Stat. Soc.-Series C, vol 22, 1973, pp 332-346.
- [10] J.D. Musa, "The measurement and management of software reliability," Proc. IEEE, vol 68, 1980 Sep, pp 1131-1143.
- [11] G.J. Myers, Software Reliability: Principles and Practice, John Wiley & Sons, New York 1976.
- [12] A.J. Perlis, F.G. Sayward, M. Shaw (eds), Software Metrics, The MIT Press, Cambridge 1981.
- [13] G.G. Roussas, A First Course in Mathematical Statistics, Addison-Wesley, Massachusetts 1973.
- [14] 酒巻恒一他, "ソフトウェアの信頼性: 品質管理のためのソフトウェアの信頼性予測," 信学技報, vol. 81, 1981年5月, pp 17-24.
- [15] G.J. Schich, R.W. Wolverton, "An analysis of competing software reliability models," IEEE Trans. Software Engineering, vol SE-4, 1978 March, pp 104-120.
- [16] 篠田東吾他, "制御欠陥と機能試験の妥当性評価," 情報処理学会第22回全国大会 5C-7, 1981年, pp 377-378.
- [17] M. Shooman, "Operational testing and software reliability estimation during program development," Record 1973 IEEE Symposium on Computer Software Reliability, 1973 April, pp 51-57.