ゴンペルツ曲線を用いた確率的ソフトウェア信頼度成長モデル†

山田 茂††

従来より日本では，ソフトウェア開発のテスト工程における代表的な品質評価法の一つとして，テスト実施時間と発見・除去されたソフトウェアエラー数の関係を，ゴンペルツ曲線に代表されるS字形成長曲線により記述し，ソフトウェア内の潜在エラー数の推定やテスト項目数の消化率をはじめとするテスト進捗度の確認が行われてきた．本論文では，この決定論的モデルとしてのゴンペルツ曲線モデルを，テストにより発見される総エラー数に対して確率則を導入することにより再構築し，ソフトウェア信頼度成長モデルとしての汎用性を高める．ここで，導入される確率則は，ソフトウェア信頼性モデルによく採用され，有望視されている非同次ポアソン過程（NHPP）である．従来のゴンペルツ曲線に修正を加えた上でNHPPの平均値関数，すなわち任意のテスト時刻までに発見される総エラー数の平均値に適用し，新たに期待残存エラー数，ソフトウェア信頼度，MTBF（平均ソフトウェア故障時間間隔）などの信頼性評価尺度を導出する．これらの定量的尺度は，従来のゴンペルツ曲線モデルでは導出できなかったものである．さらに，ゴンペルツ曲線に基づくNHPPモデルの信頼性評価例も示す．

1. まえがき

高品質ソフトウェアを効率良く開発（生産）することは，緊急に対処すべき問題であることは周知の事実である．このため，ソフトウェア工学の観点からソフトウェア開発プロセスにおける品質管理[1]が必要となり，ソフトウェアの代表的な品質特性である信頼性を計量化して，開発プロセスにおけるその達成状況の把握や出荷品質の確認，さらにユーザに対する出荷時期の見積りを行うことが重要視されている．したがって，ソフトウェア信頼性を定量的に評価するために，実用的なソフトウェア信頼性評価モデルが従来より研究されている[2)~8)]．そのうち，最もよく研究され，妥当性や有効性が高いといわれるものが，確率・統計論に基づくソフトウェア信頼度成長モデル（software reliability growth model）である．これは，ソフトウェア開発のテスト工程や運用段階におけるソフトウェアエラー発見事象あるいはソフトウェア故障発生現象を，ソフトウェアの信頼度成長過程として記述するものである．ここで，ソフトウェア故障とは，開発プロセスで作り込まれてソフトウェア内に潜在するエラー（開発プロセスで作り込まれたバグと呼ばれる欠陥や誤り）により，期待どおりにソフトウェアが動作しないことと定義する．

日本では従来より，コンピュータメーカやソフトウェアハウスにおいて，テスト実施時間とテストにより発見された総エラー数の関係を，経験的にゴンペルツ（Gompertz）曲線あるいはロジスティック（logistic）曲線などのS字形成長曲線により表現して，ソフトウェア品質を評価する方法がとられてきた（例えば菅野[9)]，三浦[10)]参照）．これは，需要予測や人口予測などによく用いられるこれらの傾向曲線を，テスト工程で観測されたソフトウェアエラーデータに直接あてはめ，各曲線の収束値を回帰分析により推定するという決定論的なソフトウェア信頼度成長モデルである．

本論文では，日本において実際のソフトウェア品質管理手法として使われている上記の傾向曲線のうち，適用例も多く実際データに対する妥当性が高いといわれるゴンペルツ曲線を取り上げる．ここで，テスト工程におけるエラー発見事象に対して，従来のゴンペルツ曲線モデルでは考慮しなかった確率則を導入し，テストにより発見される総エラー数の平均的な挙動を，修正されたゴンペルツ曲線を用いて記述する．特に，任意のテスト時刻までに発見される総エラー数を表す確率変数を導入し，その確率則として非同次ポアソン過程（nonhomogeneous Poisson process，以下NHPPと略す）[7),11)]を仮定する．まず，ゴンペルツ曲線モデルの概要と意味付けを述べる．つぎに，従来より使われてきたゴンペルツ曲線に修正を加え，これをNHPPの平均値関数とするソフトウェア信頼度成長モデルを議論する．さらに，実際のテスト工程で観測されたソフトウェアエラーデータに対する適用例も示す．

† A Stochastic Software Reliability Growth Model with Gompertz Curve by SHIGERU YAMADA (Department of Industrial and Systems Engineering, Faculty of Engineering, Hiroshima University).

†† 広島大学工学部第二類計数管理工学教室

2. ゴンペルツ曲線モデル

従来より，日本の多くのコンピュータメーカやソフトウェアハウスでは，ソフトウェア開発プロセスの最終工程であるテストにおいて，テスト時間の経過にともなって発見される総エラー数をS字形成長曲線によって表現できることから，ゴンペルツ曲線やロジスティック曲線を観測されたソフトウェアエラーデータに直接あてはめて，その解析結果に基づく品質評価法がとられてきた[12]．すなわち，テストにおけるエラーの発見数は，テスト初期では少なく，中期に移行すると急速に増加し，末期に近づくにつれて飽和状態になるというテスト期間全体の時間的傾向を記述して，ソフトウェア内に潜在する総エラー数の推定を中心とする品質評価尺度を導出するものである．ここで，ゴンペルツ曲線により上記のエラー発見事象を記述するにあたっては，その時間的変化を定式化するための確率則は仮定していない．また，テスト時間の計測単位としては，カレンダ時間（日，週，月など）や，工数，テスト項目数などのテスト労力がよく採用されている．本論文では，実際のソフトウェアエラーデータに対する適用例も多く妥当性の高さが指摘されている[12]ゴンペルツ曲線を取り上げる．

ゴンペルツ曲線モデルにおいては，テスト時刻 t までに発見される総エラー数は次式によって与えられる．

$$G_0(t)=ka^{(b^t)}\ (k>0, 0<a<1, 0<b<1). \quad (1)$$

ここで，a, b, および k は定数パラメータである．特に，

$$\lim_{t\to\infty} G_0(t)=k, \quad (2)$$

であるから，k はテスト開始前にソフトウェア内に潜在する総エラー数を表す．また，式(1)は

$$\frac{dG_0(t)}{dt}\Big/G_0(t)=(\ln a\cdot\ln b)b^t, \quad (3)$$

と変形され，さらに

$$\left.\begin{aligned}&\ln\left[\frac{dG_0(t)}{dt}\Big/G_0(t)\right]=A+Bt\\&A=\ln(\ln a\cdot\ln b),\ B=\ln b\end{aligned}\right\}, \quad (4)$$

の関係式を得る．式(3)より，ゴンペルツ曲線モデルは，任意のテスト時刻 t におけるエラーの発見数の増加率が幾何級数的に減少していく過程を記述するものであることがわかる．式(4)の関係式は，式(1)で表されるゴンペルツ曲線モデルのパラメータ a, b, および k を，線形回帰分析により推定するために用いられる．テスト工程において，一定のテスト時刻 t_k までに発見された総エラー数が y_k であるという n 組のソフトウェアエラーデータ (t_k, y_k) $(k=1, 2, \cdots, n;\ 0<t_1<t_2<\cdots<t_n)$ が観測されたものとする．このデータセットを用いて，式(4)に基づく回帰分析から，モデルパラメータ a, b, および k の推定値はそれぞれ次式により与えられる（例えば三浦[10]参照）．

$$\hat{a}=\exp\left[\frac{\exp(\hat{A})}{\hat{B}}\right],\quad \hat{b}=\exp(\hat{B}),\quad \hat{k}=\frac{\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n}\hat{a}^{(\hat{b}^{t_i})}} \quad (5)$$

3. ゴンペルツ曲線に基づく NHPP モデル

一般に，ソフトウェア信頼度成長モデルは，テスト工程におけるエラー発見事象を，実施されたテスト時間とテストにより発見される総エラー数あるいはソフトウェア故障発生時間間隔の関係によりとらえ，この関係をソフトウェアの信頼度成長過程として記述するものである[8),13)]．そこで，任意のテスト時刻 t までに発見された総エラー数を表す確率変数を $N(t)$ とする．ソフトウェア信頼度成長モデルとして代表的な NHPP モデルは，計数過程 $\{N(t), t\geq 0\}$ により記述されるエラー発見事象に対して，次のような性質をもつ NHPP を仮定するものである（文献 7), 11) 参照）．

(1) $N(0)=0$.

(2) 計数過程 $\{N(t), t\geq 0\}$ は独立増分をもつ．

(3) $\Pr\{N(t+\Delta t)-N(t)=1\}=h(t)\Delta t+o(\Delta t)$.

(4) $\Pr\{N(t+\Delta t)-N(t)\geq 2\}=o(\Delta t)$.

ここで，$\Pr\{A\}$ は事象 A の起こる確率を表し，関数 $o(\Delta t)$ は微小時間 Δt の2次以上の高次の影響は無視できることを示し，

$$\lim_{\Delta t\to 0}\frac{o(\Delta t)}{\Delta t}=0,$$

である．性質(3)における $h(t)$ は，エラー発見率を表し，NHPP の強度関数 (intensity function) と呼ばれる．以上の性質(1)～(4)を使って，次の NHPP の確率関数を得る．

$$\left.\begin{aligned}&\Pr\{N(t)=n\}=\frac{\{H(t)\}^n}{n!}\exp[-H(t)]\\&\qquad(n=0, 1, 2, \cdots)\\&H(t)=\int_0^t h(x)dx\end{aligned}\right\}. \quad (6)$$

式(6)の $H(t)$ は，$N(t)$ の平均値すなわちテスト時間 $(0, t]$ において発見される総期待エラー数を表し，

NHPP の平均値関数 (mean value function) と呼ばれる.

2章で議論したゴンペルツ曲線モデルは，テスト工程におけるエラー発見事象に対して確率則を仮定しない決定論的モデルであった．このゴンペルツ曲線モデルに，確率則として上記の NHPP を導入して，汎用性の高いソフトウェア信頼度成長モデルとして再構築する．通常の NHPP に基づくソフトウェア信頼度成長モデルと同様にして，式(1)の $G_0(t)$ を平均値関数とする NHPP を考えると，上記の NHPP の性質(1)が満足されない．これは同時に，従来のゴンペルツ曲線モデルの欠陥でもある．すなわち，$G_0(0)=ka\neq 0$ であり，テスト開始前に発見されるエラー数が存在するという不合理な性質を持っている．これを解消するためには，

$$G(t)=G_0(t)-ka=k(a^{b^t}-a), \qquad (7)$$

とすればよい．また，式(7)の $G(t)$ を NHPP の平均値関数とすれば，上述の NHPP の性質(1)は確率1で満足され，式(6)の $H(t)$ に式(7)を代入して

$$\Pr\{N(t)=n\}=\frac{\{G(t)\}^n}{n!}\exp[-G(t)] \quad (n=0,1,2,\cdots), \qquad (8)$$

を得る．ここで，

$$G(0)=0,\ G(\infty)=k(1-a), \qquad (9)$$

である．したがって，式(7)はテスト実施期間中に発見されるエラー（すなわちプログラムの動的解析により発見可能なエラー）を対象とし，テストにより最終的に発見されるエラー数の期待値は $k(1-a)$ であることを意味している．式(7)より，NHPP の強度関数は

$$g(t)\equiv\frac{dG(t)}{dt}=ka^{b^t}(\ln a\cdot\ln b)b^t, \qquad (10)$$

により与えられる．

式(7)および式(8)により定式化される NHPP モデルから，種々のソフトウェアの定量的な信頼性評価尺度を導出することができる．テスト時刻 t におけるソフトウェア内の期待残存エラー数 $n_r(t)$ は，確率変数 $\{N(\infty)-N(t)\}$ の期待値（平均値）を考えて次式により与えられ，これはその分散値にも一致する．

$$n_r(t)\equiv\mathrm{E}[N(\infty)-N(t)]=\mathrm{Var}[N(\infty)-N(t)]=k(1-a^{b^t}). \qquad (11)$$

ここで，$\mathrm{E}[A]$ および $\mathrm{Var}[A]$ は，それぞれ確率変数 A の期待値および分散値を表す．式(11)から，残存エラー数 $\{N(\infty)-N(t)\}$ も NHPP に従うことがわかる．また，S_k を k 番目のソフトウェア故障発生時刻を表す確率変数とする ($k=1,2,\cdots$)．このとき，テスト時刻 t で n 番目のソフトウェア故障が起こったという条件の下で，時間区間 $(t,t+x]$ で次の $(n+1)$ 番目のソフトウェア故障が発生しないという条件付き確率は，

$$R(x|t)\equiv\Pr\{S_{n+1}-S_n>x|S_n=t\}=\exp[-\{G(t+x)-G(t)\}]$$

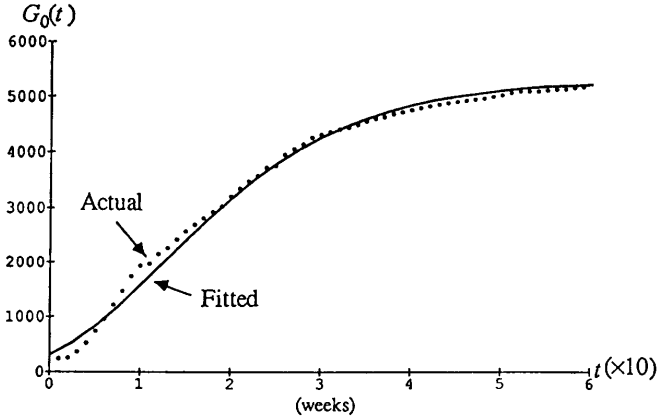

図 1 ゴンペルツ曲線 $G_0(t)$ の推定結果
Fig. 1 Estimated Gompertz curve $\hat{G}_0(t)$ along with actual data.

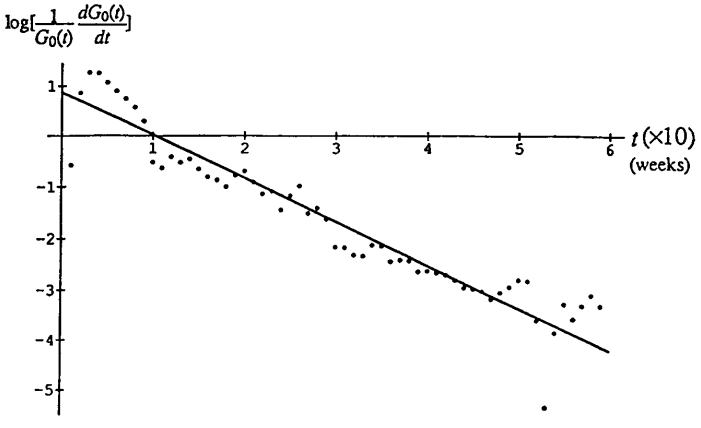

図 2 $\log\left[\frac{1}{G_0(t)}\frac{dG_0(t)}{dt}\right]$ の実測値と推定値
Fig. 2 Actual and fitted values for $\log\left[\frac{dG_0(t)}{dt}/G_0(t)\right]$.

$$=\exp[k\{a^{b^t}-a^{b^{(t+x)}}\}] \quad (t\geq 0, x\geq 0), \tag{12}$$

により与えられ，発生するソフトウェア故障数 n には無関係な確率量である．式(12)の $R(x|t)$ は，一般にソフトウェア信頼度 (software reliability)[6),7),14)] と呼ばれ，t を実施された総テスト時間，x を所定のテスト時間長と考えれば，テスト時刻 t における信頼性の達成度合を評価できる．さらに，n 番目と $(n+1)$ 番目の間の平均ソフトウェア故障時間間隔 $E[S_{n+1}-S_n]$ (mean time between software failures, 通常 MTBF により表される) は，理論的には存在せず求められないが (例えば文献 8), 15) 参照)，代替的に次式の累積 (cumulative) MTBF および瞬間 (instantaneous) MTBF をそれぞれ求めることができる[7),16)]．

$$Mc(t)\equiv\frac{t}{G(t)}=\frac{t}{k(a^{b^t}-a)}, \tag{13}$$

$$M_I(t)\equiv\frac{1}{g(t)}=\frac{a^{-b^t}\cdot b^{-t}}{k(\ln a\cdot\ln b)}. \tag{14}$$

なお，上述したゴンペルツ曲線に基づく NHPP モデルでは，2章で示したようなソフトウェアエラーデータ (t_k, y_k) $(k=1,2,\cdots,n)$ が観測されたときに，パラメータ a, b, および k の推定値 $\hat{a}$, $\hat{b}$, および $\hat{k}$ は回帰分析により求めた式(5)を用いることにする．もちろん，一般の NHPP モデルと同様にして最尤法(method of maximum likelihood)[6),7)] によりパラメータ a, b, および k を推定したいときには，数値計算上は非常に複雑になるが，尤度関数を

$$L=\prod_{k=1}^{n}\frac{\{G(t_k)-G(t_{k-1})\}^{(y_k-y_{k-1})}}{(y_k-y_{k-1})!}\exp[-\{G(t_k)-G(k_{k-1})\}], \tag{15}$$

として，

$$\partial\ln L/\partial a=\partial\ln L/\partial b=\partial\ln L/\partial k=0, \tag{16}$$

を数値的に解けばよい．

4. 適 用 例

3章において議論したゴンペルツ曲線に基づく NHPP モデルを使って，具体的なソフトウェアの信頼性評価例を示す．ここで用いたソフトウェアエラーデータは，三浦[10)]が引用した59組のデータセット (t_k, y_k) $(k=1,2,\cdots,59$; テスト時間 t_k の単位は週で 1/10 に縮尺してある) である．2章

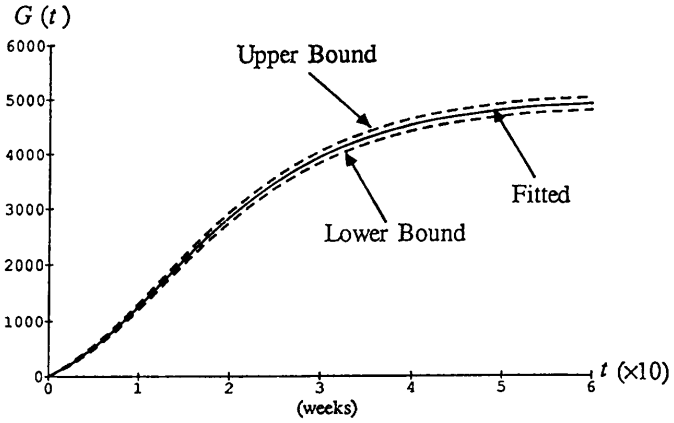

図 3 平均値関数 $G(t)$ の推定結果
Fig. 3 Estimated mean value function $\hat{G}(t)$ along with the 90% confidence bounds.

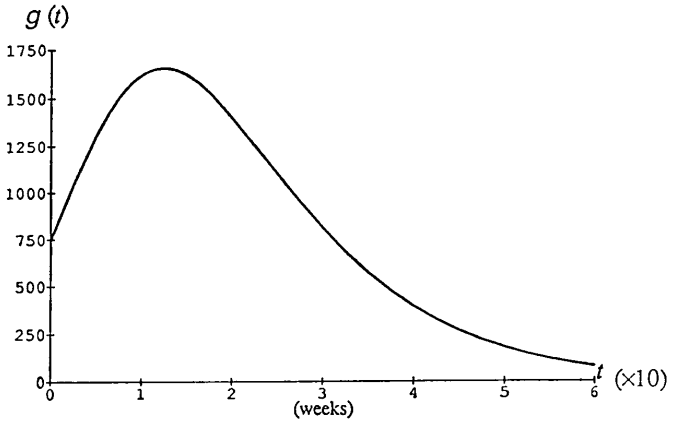

図 4 エラー発見率 $g(t)$ の推定結果
Fig. 4 Estimated error detection rate $\hat{g}(t)$.

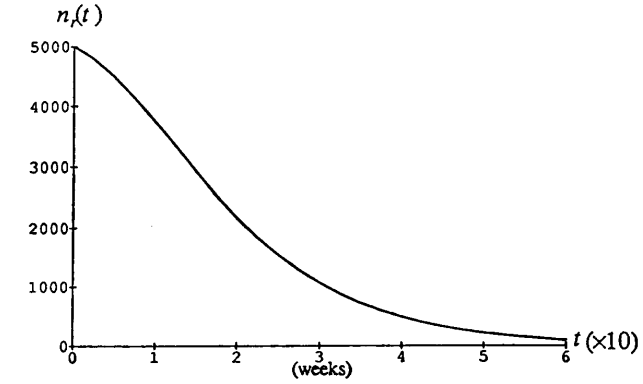

図 5 期待残存エラー数 $n_r(t)$ の推定結果
Fig. 5 Estimated expected number of detected errors $\hat{n}_r(t)$.

で述べた回帰分析により，式(1)のゴンペルツ曲線のパラメータの推定値 $\hat{a}=0.5884\times10^{-1}$, $\hat{b}=0.4293$, および $\hat{k}=0.5322\times10^{4}$ が得られる．これらの推定結果を使って，式(1)の推定値 $\hat{G}_0(t)$ を実測値とともに示したのが**図1**である．ここで，$\hat{k}\hat{a}=313.1$ であるから，テストにより発見される総期待エラー数 $\hat{k}(1-\hat{a})$ は約5,008個と推定される．また，式(4)のエラー発見数の増加率の対数値 $\ln\left[\frac{dG_0(t)}{dt}\Big/G_0(t)\right]$ とテスト時間の関係について，実測値と推定値を示したのが**図2**である．図2から，本ソフトウェアのテスト工程におけるエラー発見数の増加率の減少量は，テスト時間の増分だけ線形的に減少していくことがわかり，この関係は上述の回帰分析における適合度検定[7]により有意水準1%で確認された．

以上の結果を使って，式(7)の推定値

$$\hat{G}(t)=5322\{(0.5884\times10^{-1})^{(0.4293)^t}-0.5884\times10^{-1}\}, \quad (17)$$

を平均値関数とするNHPPモデルの推定結果を**図3**に示した．図3には，実測データ (t_k, y_k) $(k=1,2,\cdots,59)$ とともに，式(17)の90%信頼限界[7),17)] (90%の確率で $\hat{G}(t)$ の存在する範囲) も示した．**図4**には，式(10)のエラー発見率の推定値 $\hat{g}(t)$ を示した．S字形成長曲線の特徴である，単位時間当りに発見されるエラー数に最大値が存在する様子がよくわかる．その最大値に達するテスト時間は，

$$t_0=\frac{-\ln(-\ln\hat{a})}{\ln(\hat{b})}=1.23, \quad (18)$$

のとき，すなわち12.3(週)のときである．

さらに，NHPPモデルの特徴的な信頼性評価尺度の推定結果を示す．**図5**および**図6**は，それぞれ式(11)のソフトウェア内の期待残存エラー数および式(12)のソフトウェア信頼度の挙動を示したものである．図6において，テスト終了時刻は5.9(59週間)であるので，それ以降のソフトウェア信頼度の推移状況を示している．また**図7**は，式(14)の瞬間MTBFの挙動を示している．テスト初期の段階では瞬間MTBFの低下現象が見られるが，図4のエラー発見率 $\hat{g}(t)$ の挙動からもわかるように，これは単位テスト時間当りに発見されるエラー数が増加するためであり，発見された総エラー数の実測値がS字形成長曲線を示す場合の典型的な信頼性特性でもある．テスト中期以降より，急速に瞬間MTBFは増加し，信頼度成長が実現されていく様子がわかる．

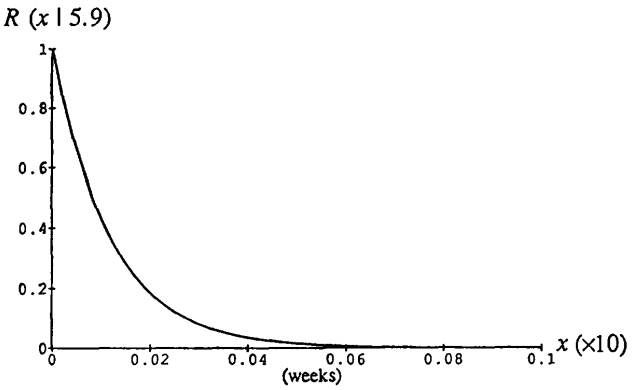

図6 ソフトウェア信頼度 $R(x|t)$ の推定結果
Fig. 6 Estimated software reliability function $\hat{R}(x|t)$ $(t=5.9)$.

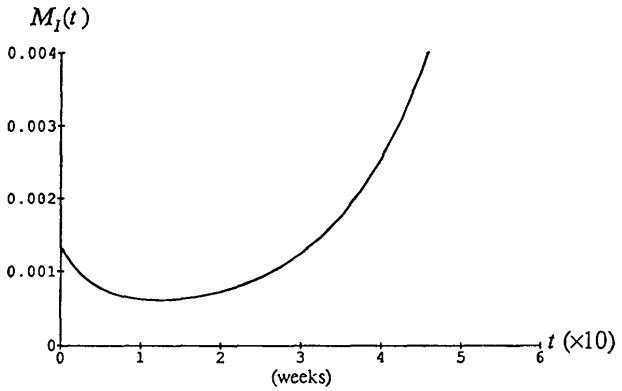

図7 瞬間MTBF $M_I(t)$ の推定結果
Fig. 7 Estimated instantaneous MTBF $\hat{M}_I(t)$.

5. むすび

本論文では，従来よりソフトウェアの品質評価のために広範に用いられてきたゴンペルツ曲線モデルをNHPPモデルとして再構築し，テストにより発見されるエラー数の変動を考慮でき，また種々の信頼性評価尺度を導出できることを示した．NHPPの平均値関数に含まれるモデルパラメータは，実際の観測データから回帰分析により推定するので，従来と同様にして容易に求められる．これは，通常のNHPPモデルによるソフトウェア信頼性評価に見られるように，最尤法に基づくモデルパラメータ推定の数値計算にともなう問題点がないので，大きな利点となっている．したがって，本論文で議論した信頼性評価法は，現在のゴンペルツ曲線モデルによる品質評価法にすぐに組み

込むことができる.

本論文では取り扱わなかったが，ゴンペルツ曲線モデルとともに品質評価のためによく適用され，テスト時刻 t までに発見される総エラー数が

$$L_0(t)=\frac{k}{1+m\cdot e^{-\alpha t}}, \qquad (19)$$

で与えられるロジスティック曲線モデルも同様にしてNHPP モデルとして再構築できる．ここで，k はテスト開始前にソフトウェア内に潜在する総エラー数であり，m および α は定数パラメータである．すなわち，NHPP における平均値関数を

$$L(t)=L_0(t)-k/(1+m)$$
$$=k\{1/(1+m\cdot e^{-\alpha t})-1/(1+m)\}, \qquad (20)$$

とすればよい．したがって，$L_0(0)=k/(1+m)$ を除外して，テストにおけるエラー発見事象を NHPP により記述するのである．

今後は，ゴンペルツ曲線あるいはロジスティック曲線に基づく NHPP モデルによるソフトウェア信頼性評価の多くの実例を示し，ソフトウェアの最適な出荷時期の見積り問題（すなわちソフトウェアのリリース問題）などのプロジェクト管理上の方策への応用を検討する予定である．

謝辞 日頃，暖かいご指導をいただく広島大学工学部の尾崎俊治教授に深謝いたします．また，本研究を進めるにあたりご協力いただいた，広島大学大学院生の藤井達也君に感謝いたします．なお，本研究の一部は文部省科学研究費（一般研究(C) 課題番号 04650316）の補助を受けたことを付記する．

参考文献

1) 菅野文友（編著）：ソフトウェアの品質管理，日科技連出版社（1986）.
2) Bittanti, S. (ed.): *Software Reliability Modeling and Identification*, Springer-Verlag (Lecture Notes in Computer Science No. 341), Berlin (1988).
3) Littlewood, B. (ed.): *Software Reliability: Achievement and Assessment*, Blackwell Scientific Publications, Oxford (1987).
4) Littlewood, B. and Miller, D. (eds.): *Software Reliability and Safety*, Elsevier Applied Science, London (1991).
5) Malaiya, Y. K. and Srimani, P. K. (eds.): *Software Reliability Models: Theoretical Development, Evaluation and Application*, IEEE Computer Society Press, Los Alamitos (1990).
6) Musa, J. D., Iannino, A. and Okumoto, K.: *Software Reliability: Measurement, Prediction, Application*, McGraw-Hill, New York (1987).
7) 山田 茂：ソフトウェア信頼性評価技術，HBJ 出版局（1989）.
8) 山田 茂，大寺浩志：ソフトウェアの信頼性～理論と実践的応用～，ソフト・リサーチ・センター（1990）.
9) 菅野文友：ソフトウェア・エンジニアリング，日科技連出版社（1979）.
10) 三浦 武：ソフトウェアの品質評価法，日科技連出版社（1981）.
11) Ascher, H. E. and Feingold, H.: *Repairable Systems Reliability: Modeling, Inference, Misconceptions and Their Causes*, Marcel Dekker, New York (1984).
12) 菅野文友（監修）：ソフトウェア品質管理事例集，日科技連出版社（1990）.
13) Ramamoorthy, C. V. and Bastani, F. B.: Software Reliability-Status and Perspectives, *IEEE Trans. Softw. Eng.*, Vol. SE-8, No. 4, pp. 354-371 (1982).
14) Goel, A. L. and Okumoto, K.: Time-Dependent Error-Detection Rate Model for Software Reliabilty and Other Performance Measures, *IEEE Trans. Reliability*, Vol. R-28, No. 3, pp. 206-211 (1979).
15) Hishitani, J., Yamada, S. and Osaki, S.: Reliability Assessment Measures Based on Software Reliability Growth Model with Normalized Method, *J. Inf. Process.*, Vol. 14, No. 2, pp. 178-183 (1991).
16) 市田 嵩，鈴木和幸：信頼性の分布と統計，日科技連出版社（1984）.
17) Yamada, S.: Software Quality/Reliability Measurement and Assessment: Software Reliability Growth Models and Data Analysis, *J. Inf. Process.*, Vol. 14, No. 3, pp.254-266(1991).

(平成4年1月16日受付)
(平成4年5月14日採録)

山田 茂（正会員）

昭和27年生．昭和50年広島大学工学部経営工学科卒業．昭和52年同大学院修士課程修了．昭和58～63年岡山理科大学勤務．昭和63年広島大学工学部第二類（電気系）助教授，現在に至る．工学博士．ソフトウェア信頼性モデル，ソフトウェアマネジメントモデル，信頼性工学，品質管理などの研究に従事．平成3年度情報処理学会 Best Author 賞受賞．著書「ソフトウェア信頼性評価技術」(HBJ 出版局)，「ソフトウェアの信頼性～理論と実践的応用～」（ソフト・リサーチ・センター）など．電子情報通信学会，日本 OR 学会，日本経営工学会，IEEE 各会員．