ゴンペルツ型ソフトウェア信頼性モデル

大石幸治, 土肥正 (01307065)
広島大学大学院工学研究科情報工学専攻

1. 概要

ゴンペルツ型ソフトウェア信頼性モデルを新たに考案する動機として以下の2点が挙げられる．まず，ソフトウェアのテスト工程において発見されるバグの累積総数を推定するための（決定論的）予測曲線として，従来からゴンペルツ曲線が経験的に用いられてきたが [1]，その理論的妥当性についてはこれまで十分に議論されることはなかった．また，ソフトウェア信頼性予測に極値分布を適用するという考え方が最近になって導入され [2]，ゴンペルツ分布はタイプI極値分布（グンベル分布）の切断分布であることから，従来研究とは異なる視点からソフトウェア信頼性モデルを構築することが可能である．山田 [3] はゴンペルツ曲線の類似関数を非定常ポアソン過程 (NHPP) の平均値関数として仮定することで，決定論的立場から NHPP タイプのソフトウェア信頼性モデルを提案している．これに対して本稿では，ソフトウェアバグのデバッギング理論に忠実な立場で，NHPP に基づいたゴンペルツ型ソフトウェア信頼性モデルを再構築する．提案モデルの実務的利点として，パラメータ推定に最尤法を適用することが可能であり，重回帰分析に基づいた従来までの方法 [1, 3] と比べて推定結果を確率論的に解釈することが出来る．

2. 従来研究

2.1 ゴンペルツ曲線

テスト工程において観測されるソフトウェアバグ発見の成長曲線を表現するために，ロジスティック曲線やゴンペルツ曲線がよく利用されている [1]．時刻 $t\ (\geq 0)$ におけるソフトウェアバグの累積発見数を $M(t)$ とすると，ゴンペルツ曲線は

$$M(t) = \omega a^{b^t} \tag{1}$$

によって与えられる．ここで，$\lim_{t\to\infty} M(t) = \omega\ (>0)$ はソフトウェアに潜在する初期バグ数として見なすことができ，$a, b \in (0,1)$ は任意のパラメータである．この曲線は $t = -\log[-\log a]/\log b$ で変曲点を持つS字曲線を形成することが知られており，遺伝学における伝染病の感染者数や人口減少数の予測，ハードウェア製品の信頼度成長現象などの記述に頻繁に利用されている．パラメータ ω, a, b は観測データに適合するよう重回帰分析によって決定されるのが通常である．$\alpha = -\log a$ 及び $\beta = -\log b$ の変数変換を行うと，ゴンペルツ曲線は

$$M(t) = \omega \exp\{-\alpha e^{-\beta t}\} \tag{2}$$

のように書き換えられる．また，$M(0) = \omega a = \omega \exp\{-\alpha\}$ であるので，$a = \exp\{-\alpha\}$ は既に発見されたバグの占有率を表す．

2.2 関連した NHPP モデル

山田 [3] は式 (1) を変形し，平均値関数

$$M(t) = \omega\{a^{b^t} - a\} \tag{3}$$

をもつ NHPP をソフトウェア信頼性モデルとして利用することを提案している．ここで，$M(0) = 0$ かつ $\lim_{t\to\infty} M(t) = \omega(1-a)$ となる．しかしながら，上述のモデルは NHPP の平均値関数にゴンペルツ曲線の類似関数を仮定し，$M(0) = 0$ となるよう人為的に調整したモデルであり，ソフトウェアに潜在する有限のフォールトからのデバッギング過程を忠実にモデル化したものではないことに注意しなければならない．さらに，式 (3) の平均値関数をもつ NHPP に対する対数尤度方程式を数値的に解くことが事実上不可能であることから [3]，異なる観点からゴンペルツ曲線に基づいたソフトウェア信頼性モデルを考察することが必要である．

3. ゴンペルツ型ソフトウェア信頼性モデル

3.1 ゴンペルツ分布

確率分布関数 $F(t)$ が以下のようなゴンペルツ分布であると仮定する．

$$F(t) = 1 - \exp\Bigl\{\frac{\lambda}{\alpha}(1 - e^{\alpha t})\Bigr\}, \quad t \geq 0. \tag{4}$$

ここで，$\lambda\ (>0)$ と $\alpha\ (>0)$ は非負の定数である．文献 [2] ではソフトウェアバグの発見時間間隔を表す確率分布としてタイプI極値分布を仮定し，ソフトウェア MTBF (Mean Time Between Failures) の予測を行っている．

3.2 ゴンペルツ型 NHPP モデル

ここでは **2.2** とは異なる視点からソフトウェア信頼性モデルを構築する．テスト開始前のソフトウェアには $N\ (\geq 0)$ 個のバグが潜在し，個々のバグを発見・除去するまでの時間は独立で同一に分布する確率変数であり，その確率分布関数を $F(t)$ とする．時刻 $t\ (\geq 0)$ までに発見・除去されるソフトウェアバグの累積総数を $\{N(t), t \geq 0\}$ とし，初期バグ数 N は平均 $\omega\ (>0)$ のポアソン分布に従うと仮定すれば，

$$\Pr\{N(t) = 0 \mid N(0) = 0\} = \frac{[\omega F(t)]^n e^{-\omega F(t)}}{n!} \tag{5}$$

を得る．これは，平均値関数 $\omega F(t)$ をもつ NHPP の確率関数である．確率分布関数 $F(t)$ が式 (4) のゴンペルツ分布に

従うならば，時刻 t までに発見される総期待バグ数を表す平均値関数は

$$M(t)=\omega\Big\{1-\exp\big[\frac{\lambda}{\alpha}(1-e^{\alpha t})\big]\Big\} \tag{6}$$

によって与えられ，$M(0)=0$ および $\lim_{t\to\infty}M(t)=\omega$ となる．さらに，単位テスト時間当たりに発見されるデバッグ率は $d(t)=(dM(t)/dt)/[\omega-M(t)]=\lambda\exp\{\alpha t\}$ となり，IFR の性質をもつ．

4. 数値例

4.1 最尤法

パラメータ $\theta=(\omega,\alpha,\lambda)$ を推定するために最尤法を用いる．テスト工程においてソフトウェアバグの発見時間データ $\mathbf{x}=(t_1,t_2,\cdots,t_n,T)$ が観測されているものと仮定する．ここで，$n\ (\geq 1)$ は時刻 $T\ (>0)$ までに観測された総バグ数である．このとき，ゴンペルツ型 NHPP モデルに対する対数尤度関数 $\log L(\theta\mid\mathbf{x})$ は

$$\begin{aligned}\log L(\theta\mid\mathbf{x}) &= \sum_{i=1}^{n}m(t_i)-M(T)\\ &= \frac{n\lambda}{\alpha}-\frac{\lambda}{\alpha}\sum_{i=1}^{n}e^{\alpha t_i}+\alpha\sum_{i=1}^{n}t_i+n\log\lambda\\ &\quad +n\log\omega-\omega\left\{1-\exp\left[\frac{\lambda}{\alpha}(1-e^{\alpha T})\right]\right\}\end{aligned} \tag{7}$$

となる．ここで，$m(t)=dM(t)/dt$ である．最終的に，同時尤度方程式 $\partial\log L(\theta\mid\mathbf{x})/\partial\theta=0$ を解くことにより，最尤推定値 $\hat{\theta}=(\hat{\omega},\hat{\alpha},\hat{\lambda})$ を求めることが出来る．

4.2 適合性評価

実際のテスト工程で観測された 136 個のソフトウェアバグ発見時間データ ($n=136$, $T=t_{136}$) を用いて，ゴンペルツ型 NHPP モデルの適合性評価を行う．最尤法に基づいてパラメータ推定を行った結果，それぞれ $(\hat{\omega},\hat{\alpha},\hat{\lambda})=(136.1962, 0.0000256763, 0.0000192039)$ となった．図 1 は累積発見バグ数と NHPP モデルの平均値関数をプロットしたものである．指数形 NHPP モデルや遅延 S 字形 NHPP モデルと比較して，ゴンペルツ型 NHPP モデルは累積発見バグ数を少なめに見積もる傾向がある．また，図 2 はソフトウェア信頼度 $R(x\mid T)=\exp\{M(x+T)-M(T)\}$ を調べた結果である．他のモデルと比較して，ゴンペルツ型 NHPP モデルはソフトウェア信頼度を高めに評価することがわかる．

5. 結論

本稿では，ゴンペルツ分布に基づいた新しいソフトウェア信頼性モデルを提案し，実データを用いた数値例を通じてモデルの適合性評価を行った．最近，式 (2) が満足する微分方程式を離散化することによって離散時間モデルを新たに導出する試みがなされているが [4, 5]，本稿で採用した方法では連続分布を離散分布に置き換えることにより（少なくとも確率論の意味において）妥当な離散時間モデルを求めることが可能である（例えば [6]）．任意の連続分布を離散分布に変換する方法としては Continuity Theory に基づいた方法がある．

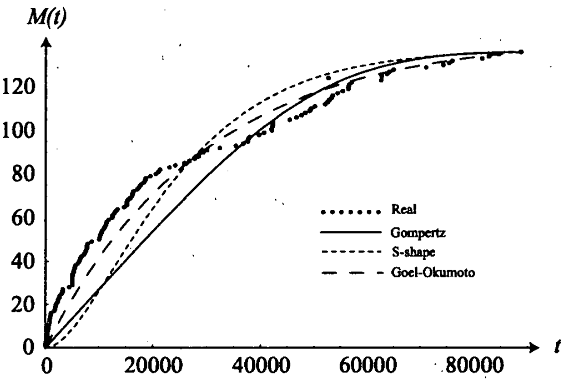

図 1: 累積発見バグ数の振舞い．

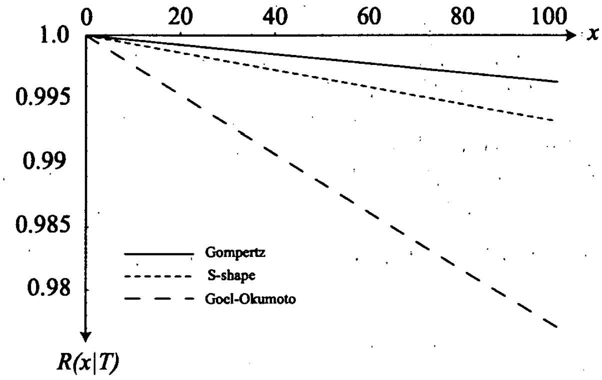

図 2: ソフトウェア信頼度の推定値の比較．

謝辞: 本研究の一部は文部科学省科学研究費萌芽研究 Grant No. 15651076 の助成の下で行われたものである．

参考文献

[1] 三觜 武,「ソフトウェアの品質評価法—統計的管理へのアプローチ」, 日科技連, 1981.

[2] L. M. Kaufman, J. B. Dugan and B. W. Johnsdon, Using statistics of the extremes for software reliability analysis, *IEEE Trans. Reliab.*, **48**, 292–299 (1999).

[3] 山田, 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, 情処学論, **33** (7), 964–969, 1992.

[4] D. Satoh, A discrete Gompertz equation and a software reliability growth model, *IEICE Trans. Inf. & Syst.*, **E83-D**, 1508–1513 (2000).

[5] 井上, 山田, 離散化統計的データ解析モデルに基づく確率的ソフトウェア信頼度成長モデル, 信学技法（信頼性）, **103** (77), 19–24 (2003).

[8] 岡村, 村山, 土肥, 累積ベルヌーイ過程による離散型ソフトウェア信頼性モデルの統一化とパラメータ推定手法に関する考察, 日本 OR 学会春季研究発表会アブストラクト集, 102–103, 富山 (2002).