ポアソン2項分布とその周辺
—信頼性問題を中心に—

土肥　正，岡村　寛之

直感的な理解が容易でかつ最も基本的な確率モデルであるポアソン試行において，その結果の累積を表わす確率分布はポアソン 2 項分布として知られている．一方，ポアソン 2 項分布の計算アルゴリズムは最近になってようやく整備され，今後，多くの確率モデルの評価に対して適用されることが期待される．本稿では，これまでに知られているポアソン 2 項分布の計算アルゴリズム（近似解法と厳密解法）を概観し，さらに応用例として，取替え問題やソフトウェアバグ予測などの信頼性問題に適用した例について紹介する．

キーワード：ポアソン 2 項分布，確率モデル，計算アルゴリズム，信頼性問題

1. はじめに

コイン投げの例において，同一のコインを投げて確率 p $(0<p<1)$ で表（報酬 1）がでて，確率 $1-p$ で裏（報酬 0）がでる現象を記述するベルヌーイ試行は最も初等的な確率モデルであり，確率変数を学ぶ際の例題として教科書の最初の部分に必ずでてくる基本モデルである．一方，ベルヌーイ試行における「同一なコインを投げる」という仮定を一般化し，i $(=1,2,\ldots)$ 番目の試行において確率 p_i $(0<p_i<1)$ で表がでて，確率 $1-p_i$ で裏がでるような独立試行はポアソン試行と呼ばれる．ベルヌーイ試行やポアソン試行のような基本モデルは，信頼性理論や待ち行列理論など，OR で扱うさまざまな確率モデルの中に構造的に含まれていることが多い．

i $(=1,2,\ldots,n)$ 番目のポアソン試行の結果を

$$X_i=\begin{cases}1 & :\text{確率 } p_i\\ 0 & :\text{確率 } 1-p_i\end{cases} \tag{1}$$

と定義すると，その平均値，分散，特性関数は，それぞれ，$\mathrm{E}[X_i]=p_i$，$\mathrm{Var}[X_i]=p_i(1-p_i)$，$\mathrm{E}[\exp(iuX_i)]=p_i\exp(iu)+1-p_i$ となる．ここで，$i=\sqrt{-1}$ は虚数単位である．

次に $S_n=\sum_{i=1}^n X_i$ を n 回のポアソン試行の累積の結果とすれば，独立性の仮定から，ただちに $\mathrm{E}[S_n]=\sum_{i=1}^n p_i$，$\mathrm{Var}[S_n]=\sum_{i=1}^n p_i(1-p_i)$，$\mathrm{E}[\exp(iuS_n)]=\Pi_{i=1}^n\{p_i\exp(iu)+1-p_i\}$ を得る．これより，確率変数 S_n の累積分布関数は一意に存在し，S_n の確率分布は，Simeon D. Poisson が最初に考察したことから ポアソン（の）2 項分布，もしくは一般化 2 項分布と呼ばれる．2 項分布の名称は，$p_i=p$ $(i=1,2,\ldots)$ のベルヌーイ試行において，

$$\Pr\{S_n=k\}=\binom{n}{k}p^k(1-p)^{n-k} \tag{2}$$

となる事実による．

しかしながら，ポアソン 2 項分布の確率関数は簡単な形で表現できないため，その計算アルゴリズムについては，確率論や計算統計学の分野において長い間議論されてきた．近年，ポアソン 2 項分布の効率的な計算アルゴリズムが開発され，ようやく手軽に計算を行える環境が整ってきたといえる．

ポアソン 2 項分布の実際問題への応用例として，高電圧電源トランスにおけるコンポネント故障数の予測，保険契約における死亡保険金支払総額の推定，企業デフォルト予測，マルチセンサフュージョンシステムの評価，データベースモデルの解析，ゲノム異常の再発判定，風力発電システムの評価など枚挙にいとまがない[1]．本稿では，これまでに知られているポアソン 2 項分布の計算アルゴリズム（近似解法と厳密解法）を概観した後に，取替え問題やソフトウェアバグ予測などの信頼性問題に適用した例について紹介する．

どひ　ただし
広島大学大学院先進理工系科学研究科
〒 739–8527 東広島市鏡山 1–4–1
dohi@hiroshima-u.ac.jp
おかむら　ひろゆき
広島大学大学院先進理工系科学研究科
〒 739–8527 東広島市鏡山 1–4–1
okamu@hiroshima-u.ac.jp

[1] ポアソン 2 項分布の応用事例は Hong [1] の参考文献を参照されたい．

2. ポアソン 2 項分布

いま，X_i $(i=1,2,\ldots,n)$ を独立で同一ではない非負の連続形確率変数とし，その累積分布関数を $F_i(t)=\Pr\{X_i\le t\}$，確率密度関数を $f_i(t)=dF_i(t)/dt$，平均を μ_i (>0) とする．一般性を失うことなく，$F_i(0)=0$，$F_i(\infty)=1$ および $F_i(t)\neq F_j(t)$ $(i\neq j; i,j=1,2,\ldots,n)$ を仮定しておく．$X_{1:n}\le X_{2:n}\le\ldots\le X_{n:n}$ を $X_1, X_2,\ldots,X_n$ の順序統計量とすれば，任意の k $(1\le k\le n)$ に対する $X_{k:n}$ の累積分布関数は

$$\begin{aligned}F_{k:n}(t)&=\Pr\{X_{k:n}\le t\}\\&=\sum_{i=k}^{n}\sum_{\mathcal{S}_i}\Pi_{l=1}^{i}F_{j_l}(t)\Pi_{l=i+1}^{n}\{1-F_{j_l}(t)\}\end{aligned}\tag{3}$$

のように表わすことができる [2]．ここで，$\mathcal{S}_i$ は $j_1<\cdots<j_i$ と $j_{i+1}<\cdots<j_n$ を満たす 1 から n までのすべての順列 $(j_1,\ldots,j_n)$ に対する和演算を示す．

式 (1) と同様に，固定された t に対して $I_1(t)$，$I_2(t),\ldots,I_n(t)$ を n 番目のポアソン試行の結果とすれば，$S_n(t)=\sum_{i=1}^{n}I_i(t)$ はパラメータ $(n,F_i(t))$ $(i=1,2,\ldots,n)$ をもつポアソン 2 項分布に従い，その累積分布関数と確率関数は，$j=0,1,2,\ldots,n$ に対して，それぞれ

$$\Pr(S_n(t)\ge j)=\Pr\{X_{j:n}\le t\},\tag{4}$$

$$\Pr(S_n(t)=j)=\sum_{\mathcal{S}_j}\Pi_{l=1}^{j}F_{j_l}(t)\Pi_{l=j+1}^{n}\{1-F_{j_l}(t)\}\tag{5}$$

となる．$p_i=F_i(t)$ とおけば，式 (5) は $S_n=\sum_{i=1}^{n}X_i$ の確率関数となる[2]．式 (5) の計算は $S_n(t)=k$ に至るすべてのポアソン試行の結果の組合せを全列挙することを意味しており，たとえば $(n,k)=(30,15)$ のとき，場合の数は ${}_{30}C_{15}=155,117,520$ となる．よって，十分大きい n に対してポアソン 2 項分布を計算することは必ずしも容易ではないことがわかる．ポアソン 2 項分布の確率関数の形状，上下限値，平均，分散，中央値，最頻値の特徴づけは，Hoeffding [4]，Darroch [5]，Samuels [6]，Gleser [7]，Wang [8] に詳しい．

[2] 統計学ではここで定義したものとは異なるポアソン 2 項分布が存在する．Sprott [3] は パラメータ μ (>0) のポアソン分布とパラメータ $0<p<1$ をもつ 2 項分布の混合分布

$$\Pr\{S=k\}=\sum_{j=0}^{\infty}\Big(\frac{\mu^j e^{-\mu}}{j!}\Big)\begin{pmatrix}nj\\k\end{pmatrix}p^k(1-p)^{nj-k}$$

$(k=0,1,2,\ldots)$ をポアソン 2 項分布と呼んでいる．

3. 計算アルゴリズム

ポアソン 2 項分布の計算では組合せ爆発の問題が生じるため，これまで近似解法と厳密解法に関する研究が並行して行われてきた．以下では，ポアソン 2 項分布をパラメトリック分布で近似する方法と，数値的に確率関数を求める厳密解法のいくつかを紹介する．

3.1 近似解法

ポアソン 2 項分布の近似理論において最も代表的なものはポアソン近似である．Le Cam [9]，Hodges and Le Cam [10] はポアソン 2 項分布 $\Pr\{S_n=k\}=\xi_{k,n}$ に対して，

$$\xi_{k,n}\approx\frac{(\sum_{i=1}^{n}p_i)^k}{k!}\exp(-\sum_{i=1}^{n}p_i)\tag{6}$$

のようなポアソン近似を与え，確率分布間の距離をはかる尺度である全変動 (total variation)

$$d_{TV}(n)=\sum_{k=0}^{n}||\xi_{k,n}-\frac{(n\overline{p})^k}{k!}\exp(-n\overline{p})||\tag{7}$$

の上下限値を導出した．ここで $\overline{p}=\sum_{i=1}^{n}p_i/n$，$\overline{q}=1-\overline{p}$ とおく．以降，多くの研究者らによってポアソン近似の全変動に対するよりシャープな上下限値を導出することが試みられてきた．たとえば，Barbour and Hall [11] によって与えられたポアソン近似の全変動の上下限値は

$$\underline{d_{TV}}(n)\le d_{TV}(n)\le\overline{d_{TV}}(n),\tag{8}$$

$$\underline{d_{TV}}(n)=\frac{1}{32}\min\Big\{\frac{1}{n\overline{p}},1\Big\}\sum_{i=0}^{n}p_i^2,\tag{9}$$

$$\overline{d_{TV}}(n)=\frac{\{1-\exp(-n\overline{p})\}}{n\overline{p}}\sum_{i=0}^{n}p_i^2\tag{10}$$

となる．

また，ベルヌーイ試行とポアソン試行の類似性から，2 項分布による近似 [10, 12]

$$\xi_{k,n}\approx\begin{pmatrix}n\\k\end{pmatrix}\overline{p}^k\overline{q}^{n-k}\tag{11}$$

の全変動の上下限値は

$$\underline{d_{TV}}(n)=\alpha\min\{(n\overline{p}\ \overline{q})^{-1},1\}\sum_{i=1}^{n}(p_i-\overline{p})^2,\tag{12}$$

$$\begin{aligned}\overline{d_{TV}}(n)&=(1-\overline{p}^{n+1}-\overline{q}^{n+1})\{(n+1)\overline{pq}\}^{-1}\\&\quad\times\sum_{i=1}^{n}(p_i-\overline{p})^2\end{aligned}\tag{13}$$

表 1　厳密解法の計算時間の比較（単位は CPU second）

n	R1	R2	DFFT	DC	DC-FFT
10	0.001	0.001	0.005	0.001	0.001
100	0.001	0.001	0.006	0.001	0.001
1000	0.002	0.009	0.014	0.178	0.004
10000	0.057	2.163	1.433	0.455	0.063
100000	5.040	NG	128.864	39.823	0.859
1000000	662.393	NG	12722.476	3221.286	7.156

となる．ここで α は $\alpha \geq 124^{-1}$ を満たすある定数である．ポアソン近似と 2 項近似の比較やその他の 2 項近似に関する結果については文献 [13, 14] を参照して頂きたい．

3.2　厳密解法

ポアソン 2 項分布の厳密解法は，漸化式による方法と逆変換による方法に大別される．最もよく知られている漸化式による方法では，X_i $(i = 1, 2, \ldots, n)$ が 2 値確率変数であることから，シャノン分解 (Shannon's decomposition) を用いて

$$\xi_{i,n} = \{1 - p_n\}\xi_{i,n-1} + p_n\xi_{i-1,n-1} \qquad (0 \leq i \leq n) \tag{14}$$

を逐次的に解けばよい [15]．ここに，$\xi_{-1,j} = \xi_{j+1,j} = 0$ $(j = 0, 1, \ldots, n-1)$ および $\xi_{0,0} = 1$ である．漸化式による表現にはいくつかのバリエーションが存在し，たとえば Chen and Liu [16] では

$$\xi_{i,n} = \frac{1}{i}\prod_{l=1}^{i}(-1)^{l-1}\sum_{j=1}^{n}\Big(\frac{p_j}{1-p_j}\Big)^l\xi_{i-l,n}, \tag{15}$$

$$\xi_{0,n} = \Pi_{i=1}^{n}\{1 - p_i\} \tag{16}$$

が紹介されている．順序統計量 $X_{1:n} \leq X_{2:n} \leq \ldots \leq X_{n:n}$ の累積分布関数 $F_{k:n}(t)$ を計算するアルゴリズムは Boncelet のアルゴリズム [17] として知られている．

一方，ポアソン 2 項分布の特性関数は簡単な形式で与えられるので，数値的な逆変換を適用することで確率関数を求めることが考えられる．Barlow and Heidtmann [18] はポアソン 2 項分布の確率母関数を展開することで k-out-of-n システム（たとえば文献 [19] を参照）の信頼度を計算する BASIC プログラムを開発している．Fernandez and Williams [20] はポアソン 2 項分布の特性関数に離散高速フーリエ変換を適用している．Hong [1] は離散高速フーリエ変換アルゴリズムにおいて，詳細な代数的手続きを明確に記述し，ポアソン 2 項分布の計算アルゴリズムを統計解析向けプログラミング言語 R によるパッケージツールとして無償公開している．最近，Biscarri et al. [21] は，ポアソン試行の畳込み演算と離散高速フーリエ変換を組み合わせた分割統治高速フーリエ変換木アルゴリズム (direct convolution and divided and conquer FFT tree algorithm) を開発し，漸化式による方法や単純に特性関数の逆変換を行う方法と比べて，高速にポアソン 2 項分布の計算を実行できることを示した．

Hong [1] や Biscarri et al. [21] によって示されているように，一般的に漸化式による方法では n が十分大きい問題を高速に解くことができないと考えられているが，それは必ずしも真実ではない．アルゴリズムの性能はプログラムの実装技術に大きく依存する点に着目し，Sakata et al. [22] は式 (14) で与えられる単純な漸化式による方法において，(k, n) の 2 次元配列に基づいた従来のアルゴリズム (R2) を 1 次元配列を使って書き換える (R1) ことでメモリ消費を大幅に削減でき，むしろ 離散高速フーリエ変換アルゴリズム [1] よりも効率的であることを示している．表 1 は文献 [22] と同じ計算環境で，漸化式アルゴリズム (R1, R2)，離散高速フーリエ変換アルゴリズム (DFFT) [1]，畳込みアルゴリズム (DC) [21]，分割統治高速フーリエ変換木アルゴリズム (DC-FFT) [21] による実行時間の比較を行ったものである．表中の NG はメモリ不足による計算打切りを表わしている．漸化式アルゴリズム (R1) と畳込みアルゴリズムは計算複雑度が $O(n^2)$ であるが，分割統治高速フーリエ変換木アルゴリズムの計算複雑度は $O(n \log n)$ となる．これより n が 100 万規模の大規模なポアソン 2 項分布を評価する場合には，分割統治高速フーリエ変換木アルゴリズムのような高速アルゴリズムを適用する必要があるが，n が 1 万程度の中規模問題であれば古典的な漸化式による方法を工夫する方がむしろ効率的に計算を行うことが可能であることがわかる．

4.　取替え問題

信頼性理論において，コンポーネントシステムの信頼度評価は基本的な問題であり，特に n 個のコンポーネン

トから構成されるシステムにおいて k $(1 \le k \le n)$ 個のコンポーネントが故障すればシステム故障に至るシステムは k-out-of-$n:F$ システムと呼ばれる．もしコンポーネント i $(=1,2,\ldots,n)$ は独立であるが同一ではない故障時間分布 $F_i(t)$ をもつとすれば，システムの信頼度関数，確率密度関数，MTTF（平均故障時間）は，それぞれ，$R_{k:n}(t)=1-F_{k:n}(t)$，$f_{k:n}(t)=dF_{k:n}(t)/dt$，$\mu_{k:n}=\mathrm{E}[X_{k:n}]=\int_0^\infty R_{k:n}(t)dt$ によって定義される．特別な場合として，直列システム（1-out-of-n: F システム）の故障率は

$$r_{1:n}(t) = \frac{f_{1:n}(t)}{R_{1:n}(t)} = -\sum_{i=1}^{n}\ln\{1-F_i(t)\} \quad (17)$$

となり，並列システム（n-out-of-$n:F$ システム）の故障率は

$$r_{n:n}(t) = \frac{f_{n:n}(t)}{R_{n:n}(t)} = \frac{\left[\Pi_{i=1}^{n}F_i(t)\right]\sum_{i=1}^{n}(f_i(t)/F_i(t))}{1-\Pi_{i=1}^{n}F_i(t)} \quad (18)$$

となる．これにより，直列システムの故障率はコンポーネントの故障率 $r_i(t)=f_i(t)/\{1-F_i(t)\}$ の性質に拘わらず常に IFR (Increasing Failure Rate) であることが示されるが，並列システムに対しては故障率の単調性は必ずしも成立しない．また，$1<k<n$ の場合の k-out-of-$n:F$ システムに対して信頼性評価を行うためにはポアソン 2 項分布の計算が必要であることは明らかである．

独立であるが同一ではない n 個のコンポーネントをもつ並列システムの取替え問題は，特殊な場合 [23, 24] を除いてほとんど考察されていない．並列システムにおいて，故障したコンポーネントの数が k_0 $(=1,2,\ldots,n)$ に達した時点ですべてのコンポーネントを取り替える一斉取替え (group replacement) を考える．ここで，一斉取替えに要する固定費用を c_0 (>0)，故障した各コンポネントを新品に取り替える費用を c_r (>0)，故障していないコンポーネントを新品に取り替える費用を c_s $(<c_r)$ とする．並列システムの動作開始から一斉取替えが終了するまでの期間は $X_{k_0:n}$ となるので，その平均値は

$$\mathrm{E}[X_{k_0:n}] = \mu_{k_0:n} = \int_0^\infty \Pr\{X_{k_0:n}>t\}dt = \int_0^\infty \Pr\{S_n(t)<k_0\}dt = \int_0^\infty R_{k_0:n}(t)dt \quad (19)$$

となる．よって，定常状態における単位時間当たりの期待費用は，再生報酬定理より，

$$\begin{aligned} C(k_0) =& \Big\{c_0 + c_s(n-k_0) + c_r k_0 \\ &+ c_d\sum_{i=1}^{k_0}\mathrm{E}\big[X_{k_0:n}-X_{i:n}\big]\Big\}/\mu_{k_0:n} \\ =& \Big\{c_0 + c_s n + (c_r-c_s)k_0 \\ &+ c_d\Big[(k_0-1)\int_0^\infty R_{k_0:n}(t)dt \\ &- \sum_{i=1}^{k_0-1}\int_0^\infty R_{i:n}(t)dt\Big]\Big\} \\ &\div \int_0^\infty R_{k_0:n}(t)dt \end{aligned} \quad (20)$$

となる．Gertsbakh [25]，Assaf and Shanthikumar [26] は X_i $(i=1,2,\ldots,n)$ が独立で同一な指数分布に従うとき，上述のような閾値取替え方策が年齢に基づいた一斉取替え方策 [27] よりも常に優れていることを示している．一方，X_i $(i=1,2,\ldots,n)$ が独立で同一ではない一般分布に従うならば，$R_{k_0:n}(t)$ が k_0 $(=1,2,\ldots,n)$ の狭義凸関数のとき，唯一の最適なコンポーネント数の取替え閾値 k_0^* $(1<k_0^*<n)$ が存在することが示される．この問題のより一般化された結果は文献 [28] で紹介されている．

5. ソフトウェアバグ予測への応用

ソフトウェア開発におけるテスト工程および保守工程において，ソフトウェアに含まれるバグ含有モジュール (bug-prone module) の識別は，ソフトウェアバグの潜在箇所を特定しテストを効率化するために重要である．各モジュールの規模や複雑性を表わすプログラム行数，サイクロマティック数，分岐数などのメトリクス情報から，モジュールごとにバグ含有確率を求め，バグ含有確率が大きい順番に単体（モジュール）テストを行う．このように，バグ含有確率により各モジュールにバグが含まれているか否かを予測する問題はソフトウェアバグ予測と呼ばれる．あるモジュールにバグが含まれていたかどうかを表わす 0-1 データとソフトウェアメトリクスデータを訓練データとして用いて，バグ含有確率に含まれるモデルパラメータを推定し，まだテストされていないモジュールのバグ含有確率を予測する．従来までに，線形ロジスティック回帰モデルや半正定値ロジスティック回帰，カーネルロジスティッ

ク回帰，サポートベクターマシンなどの非線形ロジスティック回帰モデル，決定木，ランダムフォレスト，バギング，ブースティングなどのアンサンブル学習モデル，多層パーセプトロン型ニューラルネットワークや畳込みニューラルネットワークなどの深層学習モデル，ナイーブベイズやベイジアンネットワークなどのベイズ学習モデルといった，ほとんどすべての機械学習モデルがバグ含有確率の予測に用いられてきた．単体テストではすべてのモジュールをレビュー/テストするとテスト費用が大幅に嵩む一方で，モジュールのバグ含有率はスパース性が高い，すなわち単体テストにおいて多くのモジュールでバグを検出できることはかなり稀であることが知られている．この事実は，バグ含有確率の予測精度は訓練データの質に大きく依存することを示唆している．

いま，分析の対象となるソフトウェアのモジュールが n 個存在するものとする．モジュール $i\,(=1,2,\ldots,n)$ に含まれる m 種類の特性データ（ソフトウェアメトリクス）を $\boldsymbol{x}_i=(\boldsymbol{x}_{i1},\boldsymbol{x}_{i2},\ldots,\boldsymbol{x}_{im})$，モジュール i にバグが含まれれば 1，そうでなければ 0 を出力する 2 値確率変数を

$$X_i=\begin{cases}1, & \text{モジュール } i \text{ にバグが含まれる場合}\\ 0, & \text{モジュール } i \text{ にバグが含まれない場合}\end{cases} \tag{21}$$

とする．ここでモジュール i のバグ含有確率

$$p_i=\mathrm{E}[X_i]=\Pr\{X_i=1\mid \boldsymbol{x}_i\}\quad i=1,2,\ldots,n \tag{22}$$

を定義し，オッズ $\ln(p_i/(1-p_i))$ が以下のような非線形回帰式

$$\ln\Big(\frac{p_i}{1-p_i}\Big)=f(\boldsymbol{x}_i) \tag{23}$$

によって表現されるものとする．ここで $f(\cdot)$ は任意のスカラー関数である．これより，ただちに

$$p_i=\frac{\exp(f(\boldsymbol{x}_i))}{1+\exp(f(\boldsymbol{x}_i))} \tag{24}$$

を得る．D. R. Cox によって提案された線形ロジスティック回帰は定式化の理解が容易であり，計算コストもほかのモデルに比べると少なくて済む利点をもつ．すなわち，線形ロジスティック回帰において，回帰式 $f(\boldsymbol{x}_i)$ は

$$f(\boldsymbol{x}_i)=\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}_i \tag{25}$$

によって与えられる．ここで，$\boldsymbol{\beta}=(\beta_1,\beta_2,\ldots,\beta_m)$ は m 次元回帰係数ベクトルであり，T は転置を意味する．線形ロジスティック回帰では，フォールト存在性向を表わすオッズ $\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0$ が $\boldsymbol{x}_i$ の 1 次式で与えられているため，バグ含有確率はオッズの各成分に関して単調関数となる．そのため，線形ロジスティック回帰ではソフトウェアバグ予測においていくつかの問題点が存在することが知られており [29]，これらの問題点を克服するために非線形ロジスティック回帰モデル，アンサンブル学習モデル，深層学習モデル，ベイズ学習モデルなどの機械学習モデルが適用されている [30]．

各機械学習モデルのソフトウェアバグ予測能力を評価するためには，訓練データに基づいてバグ含有確率を推定し，検証データによってソフトウェアバグ予測の精度を比較する．すなわち，テストされていないモジュールに対して予測されたバグ含有確率がある閾値の値（通常は 0.5）を超えた場合，検証対象であるモジュールに少なくとも一つのフォールトが含まれると判断する．対象プログラムにおいて，結果的に正しくバグが存在すると予測できたモジュール数を TP (True Positive)，正しくバグが存在すると予測できなかったモジュール数を TN (True Negative)，誤ってバグが存在すると予測したモジュール数を FP (False Positive)，誤ってバグが存在しないと予測したモジュール数を FN (False Negative) とすれば，予測における適合率，再現率，正解率は以下のように与えられる．

$$\text{適合率}=\mathrm{TP}/(\mathrm{TP}+\mathrm{FP}), \tag{26}$$
$$\text{再現率}=\mathrm{TP}/(\mathrm{TP}+\mathrm{FN}), \tag{27}$$
$$\text{正解率}=(\mathrm{TP}+\mathrm{TN})/(\mathrm{TP}+\mathrm{FP}+\mathrm{FN}). \tag{28}$$

すなわち，適合率は予測結果の中にどの程度正解が含まれているかの割合を示し，再現率は正解のうちどの程度の割合で予測できたかを示す指標である．これら二つの指標は明らかにトレードオフ関係にあるため，適合率と再現率の調和平均である F 値を用いることで，予測性能を評価することがなされる．F 値は

$$\text{F 値}=\frac{2\times\text{適合率}\times\text{再現率}}{\text{適合率}+\text{再現率}} \tag{29}$$

によって定義され，0 と 1 の間の値をとり，F 値の値が大きくなるほど機械学習モデルの予測性能が高いといえる．よって F 値が大きくなる機械学習モデルを選ぶことで，ソフトウェアバグ予測の精度は向上することになる．

しかしながら先にも述べたように，ソフトウェアバグ予測の予測精度は訓練データの品質に大きく依存す

表 2　XG ブーストによるバグ残存モジュール数の予測（訓練データ 50 セットの平均値）

評価規範	10%	20%	30%	40%	50%	60%	70%	80%	90%
正解率	87.89	92.54	96.24	98.64	99.57	99.84	99.96	99.99	100.00
F 値	0.41	0.32	0.11	0.09	0.03	0.01	0.00	0.00	0.00
実バグ残存モジュール数	160.76	90.10	37.62	12.26	3.18	0.96	0.16	0.04	0.00
期待バグ残存モジュール数	134.60	56.59	16.85	4.21	0.67	0.19	0.06	0.02	0.00
バグ残存モジュール数の分散	55.53	27.90	11.65	3.24	0.61	0.18	0.06	0.02	0.07

るため，予測精度の高い機械学習モデルもデータによって大きく異なる．また，バグ含有確率によってまだテストしていないバグ含有モジュールをランキングしたとしても，どれだけバグ含有モジュールが残っているかの情報は単体テストをいつ停止するかを判断するために必要である．従前までのソフトウェア工学における文脈では，機械学習モデルの予測精度の向上に主眼がおかれており，残存バグ含有モジュール数の見積もりについては考慮されてこなかった．すなわち，まだ単体テストを完了していないモジュールに対してバグ含有確率を予測し，バグ含有モジュール数の確率分布にポアソン 2 項分布を直接適用すればよいことがわかる．

以下では，まず最初に全モジュール数を 10 等分に分割し，最初の 10%分のモジュールをテストすることで残り 90%分に相当するモジュールに対するバグ含有確率を予測する．次に，単体テストを行っていない残りのモジュールをバグ含有確率の大きい順に並び替え，次の 10%分のモジュールをテストする．n 個のモジュールに含まれるソフトウェアメトリクスデータ $\boldsymbol{x}_i$ $(i = 1, 2, \ldots, n)$ は単体テストを開始する前までに計測されており，各予測時点 $i = n/10, 2n/10, \ldots, 9n/10$ において 2 値確率変数 X_i の実現値は各モジュールのテスト結果として与えられる．ここでは NASA で開発された静止衛星制御ソフトウェア PC4 のバグ予測を行った結果を紹介する．PC4 は 2,017 個のモジュールから構成され，最終的に全モジュール数の 15.5%がバグ含有モジュールであり，各モジュールを特徴づけるために $m = 34$ 種類のメトリクスデータが観測されている．訓練データの選び方はバグ予測の精度に影響を与えるため，$n = 2{,}017$ から 10% 分のモジュールをランダムに抽出し，50 セットの初期データを作成する．次に，抽出されたモジュールの単体テストを実施しバグの有無を確認した後，適当な機械学習モデル用いて残りすべてのモジュールのバグ含有確率を求め，テストしていないモジュールをバグ含有確率の大きい順にランキングした後，次にテストを実施する 10%分のモジュールを選定する．この段階で，バグ残存モジュール数の情報はポアソン 2 項分布の平均，分散，確率関数として得られる（バグ含有モジュール信頼度のような新しい信頼性評価尺度も導出することは可能である）．50 セットのサンプルに対して バグ残存モジュール数の平均，分散，確率関数の各算術平均をとることで，評価結果のバイアスを削減する．また，事後的にすべてのモジュールに対するバグの有無はわかっているので，バグ含有確率の予測時点における実際のバグ残存モジュール数や F 値を導出する．

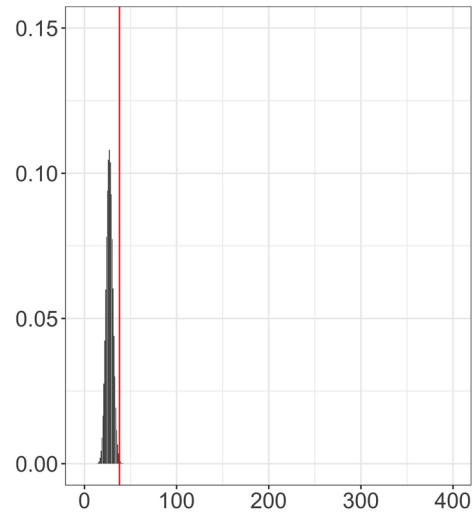

図 1　バグ残存モジュール数の予測分布（30% 予測時点）

表 2 は，機械学習モデルとして XG ブーストを適用したとき，各予測時点で正解率，F 値，実バグ残存モジュール数の 50 サンプルの算術平均を求めた結果である．正解率はすべての予測フェーズで高い水準を維持するが，F 値の平均は相対的に低く，30%予測時点から急激に低下しほぼ 0 の値となる．これはテストしていないモジュールにバグが含まれていないことによる．一方，PC4 では 60%予測時点において実バグ残存モジュール数の平均は 1 以下となるが，ポアソン 2 項分布の平均でもある期待バグ残存モジュール数も同様な値を示し，その分散も平均と同じくらい小さい値をとる．故に，F 値による予測性能が低下したとし

ても，バグ残存モジュール数の予測は高い精度で行われていることがわかる．図 1 は 30% 予測時点におけるバグ残存モジュール数の予測分布である．図中の縦線で示された値は実バグ残存モジュール数の 50 サンプルの算術平均であり，30% 予測時点ではばらつきも大きく，単体テストを停止するほどテストは進捗していないことがわかる．

6. まとめ

本稿ではポアソン 2 項分布の計算アルゴリズムを概観し，並列コンポーネントシステムの一斉取替え問題とソフトウェアのバグ予測を行う際に有効であることを示した．ここではポアソン 2 項分布に関連した信頼性問題について紹介したが，このほかにも判別問題を含む多くの OR の問題において厳密解法を適用する場面は案外多いのではないかと思われる．

参考文献

[1] Y. Hong, “On computing the distribution function for the Poisson binomial distribution,” *Computational Statistics and Data Analysis*, **59**, pp. 41–51, 2013.

[2] H. A. David and H. N. Nagaraja, *Order Statistics*, *Third Edition*, Wiley, 2003.

[3] D. A. Sprott, “The method of maximum likelihood applied to the Poisson binomal distribution,” *Biometrics*, **14**, pp. 97–106, 1958.

[4] W. Hoeffding, “On the distribution of the number of successes in independenrt trials,” *Annals of Mathematical Statisctics*, **27**. pp. 713–7211, 1956.

[5] J. N. Darroch, “On the distribution of the distribution of the number of successes in independenrt trials,” *Annals of Mathematical Statisctics*, **35**. pp. 1317–1321, 1964.

[6] S. M. Samuels, “On the number of successes in independent trials,” *Annals of Mathematical Statistics*, **36**, pp. 1272–1276, 1965.

[7] L. J. Gleser, “On the distribution of the number of successes in independenrt trials,” *Annals of Probability*, **3**. pp. 182–1881, 1975.

[8] Y. H. Wang, “On the number of successes in independent trials,” *Statistica Sinica*, **3**, pp. 295–312, 1993.

[9] L. Le Cam, “An approximation theorem for the Poisson binomial distribution,” *Pacific Journal of Mathematics*, **10**, pp. 1181–1197, 1960.

[10] J. L. Hodges, Jr. and L. Le Cam, “The Poisson approximation to the Poisson binomial distribution,” *Annals of Mathematical Statistics*, **31**, pp. 737–740, 1960.

[11] A. D. Barbour and P. Hall, “On the rate of Poisson convergence,” *Mathematical Proceedings of the Cambridge Philosophical Society*, **95**, pp. 473–480, 1984.

[12] W. Ehm, “Binomial approximation to the Poisson binomial distribution,” *Statistics & Probability Letters*, **11**, pp. 7–16, 1991.

[13] K. P. Choi and A. Xia, “Approximating the number of successes in independent trials: Binomial versus Poisson,” *Annals of Applied Probability*, **12**. pp. 1139–1148, 2002.

[14] E. A. Pekoz, A. Rollin, V. Cekanavicius and M. Shwartz, “A threee-parameter binomial approximation,” *Journal of Applied Probability*, **46**, pp. 1073–1085, 2009.

[15] W. Kuo and M. J. Zuo, *Optimal Reliability Modeling*, Wiley, 2003.

[16] S. X. Chen and J. S. Liu, “Statistical application of the Poisson-binomial and conditional Bernoulli distributions,” *Statistica Sinica*, **7**, pp. 875–892, 1997.

[17] C. G. Jr. Boncelet, “Algorithms to compute order statistic distributions,” *SIAM Journal of Scientific and Statistical Computing*, **80**, pp. 868–876, 1987.

[18] R. E. Barlow and K. D. Heidtmann, “Computing k-out-of-n system reliability,” *IEEE Transactions on Reliability*, **33**, pp. 322–323, 1984.

[19] M. Ram and T. Doh (eds.), *Systems Engineering - Reliability Analysis Using k-out-of-n Structures*, CRC Press, 2019.

[20] M. Fernandez and S. Williams, “Closed-form expression for the Poisson-binomial probability density function,” *IEEE Transactions on Aerospace Electronic Systems*, **46**, pp. 803–817, 2010.

[21] W. Biscarri, S. D. Zhao and R. J. Brunner, “A simple and fast method for computing the Poisson binomial distribution function,” *Computational Statistics and Data Analysis*, **122**, pp. 92–100, 2018.

[22] Y. Sakata, T. Dohi and H. Okamura, “Comparison of computation algorithms for Poisson binomial distributions,” *IEICE Technical Report on Reliability*, **120**, no. R-60, pp. 21–26, 2020.

[23] S. Eryilmaz, “The number of failed components in a k-out-of-n system consisting of multiple types of components,” *Reliability Engineering and System Safety*, **175**, pp. 246–250, 2018.

[24] J. Wang, J. Ye and L. Wang, “Extended age maintenance models and its optimization for series and parallel systems,” *Annals of Operations Research*, published online, 2019.

[25] I. B. Gertsbakh, “Optimal group preventive maintenance of a system with observable state parameter,” *Journal of Applied Probability*, **16**, pp. 923–925, 1984.

[26] D. Assaf and J. G. Shanthikumar, “Optimal group maintenance policies with continuous and periodic inspections,” *Management Science*, **33**, pp. 1440–1452, 1987.

[27] K. Okumoto and E. A. Elsayed, “An optimum group maintenance policy,” *Naval Research Logistics Qualterly*, **30**, pp. 667–674, 1983.

[28] Y. Sakata, T. Dohi, H. Okamura and C.-H. Qian, “Optimal group replacement policies for a parallel system with non-identical components,” under submission.

[29] L. C. Briand, W. L. Melo and J. Wust, “Assessing the applicability of fault-proneness models across object-oriented software projects,” *IEEE Transactions on Software Engineering*, **28**, pp. 706–720, 2002.

[30] C. Tantithamthavorn, S. McIntosh, A. E. Hassan and K. Matsumoto, “An empirical comparison of model validation techniques for defect prediction models,” *IEEE Transactions on Software Engineering*, **43**, pp. 1–18, 2017.

