段階的規模見積りモデルの概念とその作成方法†

宮崎幸生†† 山田松治††† 板倉稔††

本論文では，事務処理分野のソフトウェアを対象に，設計作業の進展と共に段階的に正確さを増すと考えられる3種類のソフトウェア規模見積りモデルの概念と，回帰式を用いてこのモデルの係数値を決定する方法を提案する．次に，この3種類のうち2種類のモデルについて提案する方法により係数値を決定し，回帰式に使用したデータで評価する限りは，一般的方法より正確度の高いモデルが作成できることを示す．また，同時にモデルが段階的に正確さを増すことも示す．段階的規模見積りモデルの入力は，画面数，帳票数，ファイル数である．ファイル数の数え方の詳細度により，概算，中間，詳細の3種類のモデルがある．規模見積りは開発の初期にわかる少数の入力変数により見積もることが理想である．主にこの理由により，画面や帳票の項目数はモデルに用いていない．この点とファイル数の数え方を明確にし，その詳細度により3種類の段階的なモデルがある点が，既存のファンクションポイント法[6]などと大きく異なる点である．規模や工数の見積りモデルの正確度を評価する際には，これらの実績値がシステムにより大きく変わるため，相対誤差が一般に使われる．一方，モデルの係数値を回帰式により決定する際には，絶対誤差の二乗和を最小にする通常の最小二乗法が用いられる．このため，モデルの正確度が向上しないという問題がある．本稿で提案する係数値の決定方法は，この問題を解決するため，相対誤差の二乗和を最小にする方法を使う．

1. まえがき

ソフトウェア開発過程の定量的データを分析するには，二つの重要なフェーズがある．一つは仮説を立てデータを収集するフェーズである．このフェーズではソフトウェア開発過程の知見や洞察から仮説を立て，その仮説が検証できるような定量的データ（ファイル数など）を，定義を厳密に行って集めることになる．データ収集に際して重要なことは，次の二つである．

- データが定義に一致しているかを繰り返しチェックすること．
- 定義に不十分な点があれば，それを改善し再度データを収集しなおすこと．

もう一つの重要なフェーズは，データを分析し仮説の検証や見直しを行うフェーズである．このフェーズでは，ソフトウェアのデータの特徴を考慮に入れて適切に分析することが重要である．実際に役立つ成果を得るためには，この二つのフェーズが何回か繰り返されることが多い．

今のところ，この二つのフェーズでの標準的な方法はない．このことが，例えばソフトウェアコストモデルの場合，数多くのモデルが出現し[1)~4)]さまざまな評価結果が得られる[5)~7)]原因の一つであろう．規模見積りモデルに関しては，まだ文献[8)~10)]も少なくコストモデルほどには多くのモデルは現れていない．しかし，やはりさまざまな評価結果が得られており[11)~13)]，そろそろ見積りモデルを作成するための方法論を真剣に議論すべき時期にきているような気がする．

本論文は二つのフェーズのうち，後者のデータ分析のフェーズを中心に，段階的規模見積りモデルの概念とその作成方法について述べる．データ収集のフェーズは極めて重要な段階であり，この段階で大きな誤りをおかせば分析のフェーズは意味がない．しかし，データ収集の段階については，収集データの定義を厳密に述べるにとどめ，実際の収集過程を詳述することは避けた．ソフトウェア開発の現場からのデータ収集とチェックの手法は，現場の人間への動機づけに代表される心理学的側面を伴う試行錯誤の過程であり，未だ断片的なノウハウにとどまり，十分体系的に説明しえない面が多いからである．

2. 段階的規模見積りモデルの概念

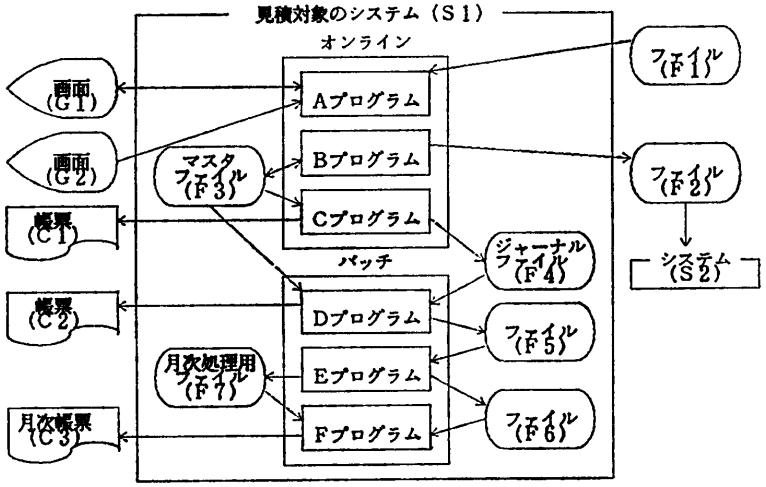
本論文で提案する段階的規模見積りモデルとは，画面数，帳票数，ファイル数を入力とし，設計作業の進展と共に段階的に正確さを増すと考えられる規模見積りモデルであり，次の3種類のモデルから成る．

①概算モデル

画面数，帳票数，システム全体の静的ファイル数（中間ファイルを除いたファイル数．**図1**参照）から見積り対象のシステム全体の規模（ステップ数，

† The Concept of Stepwise Software Sizing Model and a Method to Develop the Model by YUKIO MIYAZAKI (SDAS Systems Department, Systems Engineering Group, FUJITSU LIMITED), SYOUJI YAMADA (2nd Systems Development Department, Systems Engineering Group, FUJITSU LIMITED) and MINORU ITAKURA (SDAS Systems Department, Systems Engineering Group, FUJITSU LIMITED).

†† 富士通(株)システム本部 SDAS システム部
††† 富士通(株)システム本部第二システム開発部



(1) システム全体の静的ファイル数
F5，F6の中間ファイルを除くので，ファイル数は5となる。
(2) オンラインサブシステムの静的ファイル数
F1，F2，F3，F4が含まれるので，ファイル数は4となる。
(3) バッチサブシステムの静的ファイル数
F3，F4，F7が含まれるのでファイル数は3となる。
(4) オンラインプログラムの動的入出力ファイル数
Aプログラムは入力1（F1），Bプログラムは入力1（F3），出力2（F2，F3），Cプログラムは入力1（F3），出力1（F4）であるので，動的入力ファイル数は3（＝1＋1＋1），動的出力ファイル数も3（＝2＋1）となる。
(5) バッチサブシステムの動的入出力ファイル数
Dプログラムは入力2（F3，F4），出力1（F5），Eプログラムは入力1（F5），出力2（F6，F7），Fプログラムは入力2（F6，F7）であるので，動的入力ファイル数は5（＝2＋1＋2），動的出力ファイル数は3（＝1＋2）となる。

図 1 段階的規模見積りモデルにおけるファイル数の数え方
Fig. 1 Counting method of the number of files for an example case.

4.1 節参照）を見積もるモデル[14]．

②中間モデル

システム全体をバッチサブシステムとオンラインサブシステムに分け，バッチは帳票数とバッチサブシステムの静的ファイル数（図1参照）から，オンラインは画面数，帳票数，オンラインサブシステムの静的ファイル数（図1参照）からそれぞれステップ数を見積もるモデル．

③詳細モデル

中間モデルと同様であるが，ファイルを入力ファイルと出力ファイルに分け，動的に数える（プログラムのフローを追って数えるため重複がある．図1参照）点が異なる．

できる限り早い段階で見積もりたいという開発者の要求に応えるため，画面や帳票の項目数はモデルに用いていない．この点とファイル数の数え方を明確にしその相違により，3種類の段階的なモデルがある点が，既存のファンクションポイント法[8]などと大きく異なる点である．

事務処理分野のソフトウェアの場合，その規模は段階的規模見積りモデルの独立変数のほかに，次のような要因に大きく影響されると考えられる．

①開発技術
- サブルーチンによる処理の共通化の程度．
- コピー句による共通化の程度．
- ソースコードの雛形（スケルトン，パラダイムなどと呼ばれる）による再利用の程度．
- データの正規化の程度．
- 画面や帳票の設計方法．

②設計者の能力

③計算あるいは変換の量

これらの要因のうち開発技術に関するものは，特定の組織あるいはグループ内ではある程度標準化されていることが多い．この要因による影響を吸収するため，段階的規模見積りモデルでは，モデルの利用者の環境での実績データに適合するように，モデルの係数を決める．したがって，収集する実績データを特定の組織に絞れば絞るほど，開発技術に関する要因が吸収され，正確度の高いモデルが作成できると考えられる．

モデルの関数形については，例えば概算モデルの場合，Z をシステム全体の見積り規模，X_1 を画面数，X_2 を帳票数，X_3 をファイル数，A_1, A_2, A_3, A_4 を正の係数とすると，

$$Z=A_1X_1+A_2X_2+A_3X_3$$
$$Z=(A_1X_1+A_2X_2+A_3X_3)^{A_4}$$
$$Z=A_1X_1+A_2X_2+A_3X_3^{A_4}$$

など様々な形が考えられる．本稿では，直観的わかりやすさと利用する際の簡便さを考えて，線形モデルを仮定した．概算，中間，詳細のどのモデルにどの関数形が最適か，あるいは，どのような開発技術を使っている環境でどの関数形が最適かなどについては，今後の課題とする．

3. 規模見積りモデルの評価方法

規模見積りのように，見積もる対象（規模）の実績

値が様々に変化する場合には，見積り値の正確さを評価する際に相対誤差を使うのが一般的である[15]．10Kステップのシステムを1Kステップと見積もった場合と100Kステップのシステムを91Kステップと見積もった場合が，同じ9Kステップの誤差ではおかしいという考え方である．一般に，相対誤差は分母を実績値として以下のように定義される．

$$RY_i=(Z_i-Y_i)/Y_i$$

ここで，Y_i は i 番目のシステムの規模の実績値であり，N 件収集されているものとする．Z_i は見積り式による i 番目のシステムの見積り規模である．Z_i はすべての i について正であるべきであり，この時，RY は -1 以下にはならない．したがって，相対誤差に下限のみあり上限はないことになり，評価の適正さを欠くため次のような相対誤差 R を定義した．

$Z_i-Y_i\geq 0$ ならば $R_i=(Z_i-Y_i)/Y_i$

$Z_i-Y_i<0$ ならば $R_i=(Z_i-Y_i)/Z_i$

R により評価を行えば，相対誤差に上限も下限もなく過剰見積りの時も，過少見積りの時も，偏りのない評価結果になることがわかる．見積り式の正確度を評価する際には，この相対誤差 R を使った次の四つの指標を用いることにする．

①相対誤差 R の二乗平均の平方根（RMSR）．小さい方が良いモデルである．

$$\mathrm{RMSR}=\sqrt{(1/N)\sum_{i=1}^{N}R_i^2}$$

② R の絶対値が25%以内に入っているデータの割合（AR 25）．6.2節のバッチの中間モデルの例だと11件のサブシステム中何%が，R の絶対値25%以内に入っているかという意味になる．この値は，大きい方が良いモデルとなる．

③相対誤差 R の絶対値の平均値（AAR）．小さい方が良いモデルである．

④相対誤差 R の平均値（AR）．0に近い方が良いモデルとなる．

4. 相対誤差を用いた最小二乗法

最小二乗法は，実績値に基づく予測モデルを作成する際にはなくてはならない道具である．しかし同時に二乗和の対象を正しく選ばないと，誤った結論を導くことがあるので注意する必要がある．規模の見積りモデルの場合，次の4種類の最小二乗法が考えられる．

a．$S1=\sum_{i=1}^{N}(Z_i-Y_i)^2$ を最小にする．

b．$S2=\sum_{i=1}^{N}[(Z_i-Y_i)/Y_i]^2$ を最小にする．

c．$S3=\sum_{i=1}^{N}[(Z_i-Y_i)/Z_i]^2$ を最小にする．

d．$S4=\sum_{i=1}^{N}R_i^2$ を最小にする．

aは一般に用いられる最小二乗法である．この方法は $S1$ の定義から明らかなように，Z_i と Y_i の差が大きなデータにフィットするため，規模の大きなデータの影響を一般に受けやすい．見積り式を評価する際は，相対誤差を使うのが一般的であることを考えると，絶対誤差の二乗和を最小にするこの方法は，好ましくない．相対誤差で評価するなら，相対誤差の二乗和が最小になるように見積り式を決めるべきであろう．したがってa以外の方法が望ましい．Z_i が正であると，

$$RY_i=(Z_i-Y_i)/Y_i$$

は -1 以下にはならないため，bの方法によると R_i が負のデータが多くなり（RY_i に上限がないから），ARが負になる傾向があるという意味で過少見積りの見積り式（以下過少見積り式と呼ぶ）が得られる．一方，

$$(Z_i-Y_i)/Z_i$$

は $+1$ 以上にはならないため，cの方法によると，逆に過剰見積りの傾向のある見積り式（以下過剰見積り式と呼ぶ）が得られる．

dの方法は R の2乗和を最小にするため，RMSRの指標で最も良い見積り式が得られる．この見積り式（以下最適見積り式と呼ぶ）は，その他の指標でもかなり良い評価結果になると考えられる．しかしこの方法だけでは，異常値があった場合にそれを検出できない可能性がある．そこで，過少見積り式と過剰見積り式を，異常値を検出するために使用する．異常値を取り除いた後に，dの方法により見積り式を求めれば，本稿で定めた評価指標による評価の高いモデルを作成できよう．

5. 段階的規模見積りモデルの作成方法

段階的規模見積りモデルの作成方法の要点を以下に述べる．

①データ収集

作成しようとするモデルに応じた実績データを収集する．例えば，バッチの中間モデルの場合には，バッチサブシステムの規模（ステップ数），帳票数，静的ファイル数を収集すれば良い．これらのデータの定義

については，6.1 節を参照されたい．

②関数形の仮定

入力変数から規模を見積もるのに最も適していると思われる関数の形を仮定する．本稿では線形を仮定する．

③異常値の削除

過少見積り式または過剰見積り式で R の平均から R の標準偏差の2倍以上離れている異常値を取り除く．異常値については，その原因をよく分析しておき，同様な原因で異常値となるデータがいくつか現れた時点でモデル化を考える．

④モデルの決定

異常値を取り除いた後，R の二乗和を最小にする最適見積り式を求める．この見積り式がソフトウェア開発の専門家達の感覚と矛盾しなければ，見積り式に基づいてモデルを作成する．係数はなるべく端数のない覚えやすい値が実用的である．

6. 段階的規模見積りモデルの作成例

段階的規模見積りモデルのうち，バッチの中間モデルと詳細モデルについて，我々の環境での実績データを用いて作成方法を説明する．

6.1 収集データの範囲と定義

本論文で分析の対象とするのは，11 件のプロジェクトから取得した一般の事務処理分野のバッチのサブシステムであり，COBOL により開発されたものである．各サブシステムについて収集したデータ項目はステップ数（規模），ファイル数，帳票数である．収集したデータの定義を以下に詳述する．

①ステップ数（規模）

COBOL のソースプログラムのコメントを除いた行数である．コピーライブラリに登録された行数（展開後ではない），共通利用のサブルーチンの行数は含まれる．画面および帳票の定義体の行数，ジョブ制御言語等 COBOL 以外のソース行数は含まれない．

②静的ファイル数

一般ファイルの場合もデータベースの場合もファイル数はレコードのフォーマットの種類の数と定義する．静的ファイル数とは，中間ファイルを除いたファイル数のことであり，見積り対象サブシステムの入出力ファイルと，サブシステム内で使われるファイルだがデータが蓄積され，週，月等の周期で利用されるファイルの合計である（図1参照）．

表 1 収集データ
Table 1 Collected data.

サブシステム	規模	帳票数	静的ファイル数	動的入力ファイル数	動的出力ファイル数
A	67,443	117	18	90	107
B	50,956	3	103	316	72
C	29,619	19	10	89	67
D	17,761	6	35	59	46
E	11,262	6	7	42	11
F	76,349	52	43	326	132
G	38,332	23	15	76	62
H	7,563	0	21	47	32
I	35,803	25	30	—	—
J	142,619	79	84	—	—
K	29,036	19	14	—	—

③動的入力ファイル数と動的出力ファイル数

ファイル数については静的ファイル数と同様にレコードのフォーマットの種類の数と定義する．動的入力ファイル数は，見積り対象サブシステム内のプログラムごとの入力ファイル数の合計である．ここで言うプログラムとは，COBOL のメインプログラムと，メインプログラムから直接あるいは間接（CALL されたサブルーチンからさらに CALL される）に CALL されるサブルーチンの集合である．動的出力ファイル数も同様に，プログラムごとの出力ファイル数の合計である．あるプログラムの出力が別のプログラムの入力となることはよくあるので，重複して数えられるファイルが多い（図1参照）．また更新されるファイルは，入力ファイルと出力ファイルの両方に数えられる．動的入力ファイル数と動的出力ファイル数については，11 件のサブシステムのうち8件についてのみ収集できた．

④帳票数

ファイル数と同様にフォーマットの種類の数を帳票数と定義する．

表1に収集データを示す．

6.2 バッチの中間モデル

（1） 関数形についての仮定

バッチの中間モデルの入力変数は段階的規模見積りモデルの概念で述べたように，帳票数と静的ファイル数である．見積り式は以下の線形関数を仮定する．

$$Z=A_1X_1+A_2X_2 \qquad (1)$$

ここで，Z は見積もられる規模，X_1 は帳票数，X_2 は静的ファイル数，A_1，A_2 は正の係数である．(1)式は定数項を持つ通常の線形関数ではない．これは，定

数項をつけるとモデルの意味が説明しにくくなるからである．定数項がプログラムとして存在するために最低限必要なステップ数の合計であれば，システムの規模と共に増加するはずであり，定数ではあり得ないからである．

（2） 異常値の削除

①過少見積り式の計算

過少見積り式を求めるには，

$$S2=\sum_{i=1}^{11}[(Z_i-Y_i)/Y_i]^2$$

$$=\sum_{i=1}^{11}[(A_1X_{1i}+A_2X_{2i}-Y_i)/Y_i]^2$$

を最小にする A_1, A_2 を決めればよい．よって，

$U_{1i}=X_{1i}/Y_i$, $U_{2i}=X_{2i}/Y_i$ とおくと，

$$\partial S2/\partial A_1=2\sum_{i=1}^{11}(A_1U_{1i}^2+A_2U_{1i}U_{2i}-U_{1i})=0$$

$$\partial S2/\partial A_2=2\sum_{i=1}^{11}(A_1U_{1i}U_{2i}+A_2U_{2i}^2-U_{2i})=0$$

となり A_1, A_2 は，

$$\begin{pmatrix}A_1\\A_2\end{pmatrix}=\begin{pmatrix}\sum_{i=1}^{11}U_{1i}^2 & \sum_{i=1}^{11}U_{1i}U_{2i}\\ \sum_{i=1}^{11}U_{1i}U_{2i} & \sum_{i=1}^{11}U_{2i}^2\end{pmatrix}^{-1}\begin{pmatrix}\sum_{i=1}^{11}U_{1i}\\ \sum_{i=1}^{11}U_{2i}\end{pmatrix}$$

となる．表1のデータにより見積り式を求めると，

$$Z=837.28X_1+433.83X_2$$

②過剰見積り式の計算

過剰見積り式を求めるには，

$$S3=\sum_{i=1}^{11}[(Z_i-Y_i)/Z_i]^2$$

$$=\sum_{i=1}^{11}(1-Y_i/(A_1X_{1i}+A_2X_{2i}))^2$$

を最小にする A_1, A_2 を決めればよい．$S3$ は，A_1, A_2 について2回連続微分可能なので，連立方程式の場合のニュートン法[16]を用いて解を求める．求める A_1, A_2 は，適当な初期値 $A_{1,0}, A_{2,0}$ から始めて収束するまで，次の式に逐次代入を行えばよい．

$$\begin{pmatrix}A_{1,n}\\A_{2,n}\end{pmatrix}=\begin{pmatrix}A_{1,n-1}\\A_{2,n-1}\end{pmatrix}-J^{-1}(A_{1,n-1},A_{2,n-1})\begin{pmatrix}f_1\\f_2\end{pmatrix}$$

ここに，

$$f_1=\sum_{i=1}^{11}[(A_1X_{1i}+A_2X_{2i}-Y_i)X_{1i}Y_i/(A_1X_{1i}+A_2X_{2i})^3]$$

$$f_2=\sum_{i=1}^{11}[(A_1X_{1i}+A_2X_{2i}-Y_i)X_{2i}Y_i/(A_1X_{1i}+A_2X_{2i})^3]$$

$$J(A_1,A_2)=\begin{pmatrix}\partial f_1/\partial A_1 & \partial f_1/\partial A_2\\ \partial f_2/\partial A_1 & \partial f_2/\partial A_2\end{pmatrix}$$

$$\partial f_1/\partial A_1=\sum_{i=1}^{11}[(3X_{1i}^2Y_i^2-2X_{1i}^2Y_i(A_1X_{1i}+A_2X_{2i}))/(A_1X_{1i}+A_2X_{2i})^4]$$

$$\partial f_1/\partial A_2=\sum_{i=1}^{11}[(3X_{1i}X_{2i}Y_i^2-2X_{1i}X_{2i}Y_i(A_1X_{1i}+A_2X_{2i}))/(A_1X_{1i}+A_2X_{2i})^4]$$

$$\partial f_2/\partial A_1=\partial f_1/\partial A_2$$

$$\partial f_2/\partial A_2=\sum_{i=1}^{11}[(3X_{2i}^2Y_i^2-2X_{2i}^2Y_i(A_1X_{1i}+A_2X_{2i}))/(A_1X_{1i}+A_2X_{2i})^4]$$

表1のデータにより過剰見積り式を求めると，

$$Z=1210.22X_1+402.62X_2$$

となり，過少見積り式と比べて帳票数の係数が大きく異なることがわかる．

③見積り式の評価と異常値の検出

過少見積り式と過剰見積り式は，それぞれ以下のようになることがわかった．

$$Z=837.28X_1+433.83X_2$$

$$Z=1210.22X_1+402.62X_2$$

参考のため，最適見積り式（$S4$ を最小にする）と一般の最小二乗法による解（$S1$ を最小にする）を求めると，それぞれ次のようになる．最適見積り式は，$S4$ が A_1, A_2 について2回連続微分可能なので，過剰見積り式と同様にニュートン法で解を求められる．

$$Z=957.45X_1+442.55X_2$$（最適見積り式）

$$Z=708.46X_1+681.76X_2$$（一般の最小二乗解）

二乗和の対象によってさまざまな解が得られることがわかる．表2にこれらの4種類の見積り式による各サブシステムの相対誤差 R と，見積り式の評価結果を示す．一般の最小二乗解は，RMSR, AR 25, AAR の三つの指標で最も悪い評価となる．したがって，やはり相対誤差の二乗和を最小にする方法を選択すべきであることがわかる．過少見積り式によると，極端に大きな誤差のあるサブシステムはないが 10% 前後の正確さで見積もられているものも少ない．最適見積り式は過少見積り式より正確な見積り式になっているが，サブシステムAの誤差が大きくなっている．過剰見積り式によるとサブシステム A だけが，121%という異常に大きな誤差となり，残りのサブシステムはすべて +20% 以下 −13% 以上になることがわかる．明らかにサブシステム A は，過剰見積り式の R の平均から R の標準偏差の2倍以上離れているので，異常値として取り除く．サブシステムAは移行のためのシステ

ムで，開発者側のみで使用し帳票も簡単なものがほとんどである．このためエンドユーザに提供する通常のアプリケーション（その他のサブシステム）に比べ，異常なデータとなったのであろう．異常値については，同様な原因で異常値になるデータを集め，別の中間モデルを作成すれば，この段階でのモデルはさらに

表 2 見積り式の評価（中間モデル）
Table 2 Evaluation of software sizing equations (Intermediate model).

サブシステム	実績規模	過少見積式 見積規模	R(%)	過剰見積式 見積規模	R(%)	最適見積式 見積規模	R(%)	一般の最小二乗解 見積規模	R(%)
A	67443	105771	57	148843	121	119988	78	95162	41
B	50956	47196	-8	45101	-13	48455	-5	72347	42
C	29619	20247	-46	27020	-10	22617	-31	20278	-46
D	17761	20208	14	21353	20	21234	20	28112	58
E	11262	8060	-40	10080	-12	8843	-27	9023	-25
F	76349	62193	-23	80244	5	68817	-11	66156	-15
G	38332	25765	-49	33874	-13	28660	-34	26521	-45
H	7563	9110	20	8455	12	9294	23	14317	89
I	35803	33947	-5	42334	18	37213	4	38164	7
J	142619	102587	-39	129427	-10	112813	-26	113236	-26
K	29036	21982	-32	28631	-1	24387	-19	23005	-26
RMSR(%)		—	35	—	38	—	32	—	44
AR25(%)		—	45	—	91	—	55	—	27
AAR(%)		—	30	—	21	—	25	—	38
AR(%)		—	-14	—	11	—	-3	—	5

表 3 サブシステムAを除いた場合の見積り式の評価（中間モデル）
Table 3 Evaluation of software sizing equations excluding subsystem A (Intermediate model).

サブシステム	実績規模	過少見積式 見積規模	R(%)	過剰見積式 見積規模	R(%)	最適見積式 見積規模	R(%)	一般の最小二乗解 見積規模	R(%)
B	50956	41698	-22	44118	-15	43051	-18	49344	-3
C	29619	27526	-8	28302	-5	27788	-7	28116	-5
D	17761	20419	15	21386	20	20914	18	22977	29
E	11262	10107	-11	10440	-8	10240	-10	10580	-6
F	76349	81091	6	83569	9	82011	7	83869	10
G	38332	34387	-11	35392	-8	34742	-10	35317	-9
H	7563	7734	2	8210	9	8006	6	9298	23
I	35803	42421	18	43824	22	42984	20	44451	24
J	142619	130073	-10	134261	-6	131712	-8	135684	-5
K	29036	28999	0	29866	3	29313	1	29887	3
RMSR(%)		—	12.3	—	12.3	—	12.1	—	15.0
AR25(%)		—	100	—	100	—	100	—	90
AAR(%)		—	10.4	—	10.6	—	10.6	—	11.8
AR(%)		—	-2.1	—	2.1	—	-0.2	—	6.1

正確なものとなろう．

（3） モデルの決定

サブシステムAを除いた 10 件のサブシステムにより，4種類の見積り式を求めると次のようになる．

$Z=1254.91\,X_1+368.28\,X_2$（過少見積り式）

$Z=1283.82\,X_1+390.94\,X_2$（過剰見積り式）

$Z=1261.89\,X_1+381.22\,X_2$（最適見積り式）

$Z=1246.75\,X_1+442.75\,X_2$（一般の最小二乗解）

これらの評価結果を**表3**に示す．過少見積り式の場合も過剰見積り式の場合も，異常値はない．この例の場合，どの見積り式を用いても結果に大きな差異はないが，厳密には，やはり最適見積り式に基づいて係数を決定すべきである．そこで，帳票の係数を1,260とし，ファイルの係数を380とする．

今のところ移行システム用のモデルはないので，サブシステムAを加えた 11 件で，このモデルを評価すると，RMSR は40%，AR 25 は91%，AAR は21%，AR は11%となる．

6.3 バッチの詳細モデル

（1） 関数形についての仮定

バッチの詳細モデルの入力変数は，帳票数およびプログラムごとの動的な入力ファイル数の合計と出力ファイル数の合計である．中間モデルと同様に，線形と仮定すれば，見積り式は以下の形の関数によって表される．

$Z=A_1X_1+A_2X_2+A_3X_3$

ここで，Z は見積もられる規模であり，X_1 は帳票数，X_2 は動的入力ファイル数，X_3 は動的出力ファイル数であり，A_1, A_2, A_3 は正の係数である．

（2） 異常値の削除

①過少見積り式の計算

中間モデルと同様の方法（独立変数が一つ増えただけ）により，過少見積り式を求めると以下のようになる．

$Z=485.99\,X_1+152.60\,X_2+51.98\,X_3$

②過剰見積り式の計算

中間モデルと同様の方法により，過剰見積り式を求めると以下のようになる．

$Z=481.39\,X_1+148.76\,X_2+84.49\,X_3$

③見積り式の評価と異常値の検出

最適見積り式と一般の最小二乗解を求めると次のようになる．

$Z=489.61\,X_1+151.32\,X_2+64.88\,X_3$

$Z=314.43\,X_1+112.02\,X_2+192.91\,X_3$

表4に四つの見積り式による各サブシステムの相対誤差 R と，見積り式の評価結果を示す．一般の最小二乗解は，AAR で他よりわずかに良い結果となっているが，RMSR と AR 25 では他の見積り式よりかなり悪い結果となっている．また，規模の大きなデータにフィットするという一般の最小二乗法の弱点が顕著に現れ，規模の小さなサブシステムHが異常値となっている．相対誤差をベースにした3種類の見積り式の評価結果は，あまり大きな違いはない．過少見積り式と過剰見積り式のどちらにも異常値はない．中間モデ

表4 見積り式の評価（詳細モデル）
Table 4 Evaluation of software sizing equations (Detailed model).

サブシステム	実績	過少見積り式		過剰見積り式		最適見積り式		一般の最小二乗解	
	規模	見積規模	R(%)	見積規模	R(%)	見積規模	R(%)	見積規模	R(%)
A	67443	76157	13	78751	17	77845	15	67511	0
B	50956	53424	5	54535	7	53956	6	50231	-1
C	29619	26298	-13	28047	-6	27117	-9	28869	-3
D	17761	14311	-24	15552	-14	14850	-20	17370	-2
E	11262	9897	-14	10066	-12	10007	-13	8713	-29
F	76349	81882	7	84680	11	83353	9	78333	3
G	38332	33590	-14	35388	-8	34458	-11	34981	-10
H	7563	8836	17	9695	28	9188	21	11438	51
RMSR(%)		—	14.4	—	16.5	—	14.0	—	24.2
AR25(%)		—	100	—	82.5	—	100	—	75
AAR(%)		—	13.3	—	12.9	—	13.1	—	12.8
AR(%)		—	-2.9	—	2.9	—	-0.1	—	1.1

ルの場合に異常値となったサブシステムAが，より詳細なデータに基づいて見積り式を求めることにより，異常値でなくなることがわかる．このように段階的規模見積りモデルは，概算モデルから詳細モデルにしたがい，モデルの異常値が出現する確率が減少する見積り方法であるとも言える．

（3） モデルの決定

過少見積り式にも過剰見積り式にも異常値がないので最適見積り式をそのままモデルにすれば良い．モデルの端数をなくすという意味から，次のモデルを設定した．

$$Z=490X_1+150X_2+65X_3$$

このモデルの評価結果は最適見積り式の評価結果とほとんど同じであるので，サブシステムAを含めた中間モデルの評価（RMSR が 40%，AR 25 が 91%，AAR が 21%，AR が 11%）より良いものとなる．

7. むすび

本論文では，画面数，帳票数，ファイル数から規模を見積もる段階的規模見積りモデルを提案し，その作成方法を実際のデータに基づいて示した．以下に要点をまとめる．

①規模見積りは開発の初期に行うことが重要である．したがって，初期にわかる少数の入力変数により見積もることが理想である．画面数，帳票数，ファイル数という少数の入力変数でも，ファイル数の数え方により，3種類の段階的に見積りの正確さを増すと思われるモデルが考えられる．

②モデルの正確度を評価する際には，実績値を分母とした通常の相対誤差ではなく，次に定義する相対誤差を用いるべきである．

- $Z_i-Y_i\geq 0$ ならば $R_i=(Z_i-Y_i)/Y_i$
- $Z_i-Y_i<0$ ならば $R_i=(Z_i-Y_i)/Z_i$

ここで，Y_i は i 番目のシステムの規模の実績値であり，Z_i は見積り式による i 番目のシステムの見積り規模である．

③モデルの係数値を回帰式により決定する場合には，通常の最小二乗法ではなく，R_i の二乗和を最小にする方法を用いるべきである．

④異常値の検出には，過少見積り式と過剰見積り式を使う方法が有効である．通常の最小二乗法では発見できない異常値が発見でき，より正確なモデルが作成できる．

⑤段階的に見積りの正確さが増すことについては，静的ファイル数を用いた中間モデルより，ファイル数を動的に数える詳細モデルの方が正確度が向上することが示せた．

今後は，概算，中間，詳細のどのモデルにどんな関数形が最適かなどについて検討するとともに，一貫性のあるデータの収集をさらに継続し，我々の環境における事務処理分野向けの規模見積りモデルを作成し，普及させていきたい．

謝辞 データの提供とモデル作成にあたって有益な議論をしていただいた，伊豆則夫氏をはじめとする富士通の見積りワーキンググループのメンバ諸氏および同ワーキンググループの推進をしてこられた中條方敏氏，平春雄氏に感謝いたします．また富士通大型機ユーザ会であるラージシステム研究会「ソフトウェアの生産性尺度分科会」では，63年度より本論文で提案している概算モデルの研究を続けており，メンバ諸氏と実りある議論をさせていただきました．ここに感謝の意を表します．

参考文献

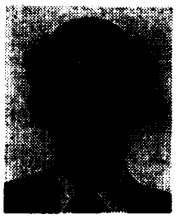
1) Boehm, B. W.: *Software Engineering Economics*, Prentice-Hall, Inc., Englewood Cliffs, N. J. (1981).
2) Putnam, L. H.: A General Empirical Solution to the Macro Software Sizing and Estimating Problem, *IEEE Trans. Softw. Eng.*, Vol. SE-4, No. 4, pp. 345-361 (1978).
3) Bailey, J. J. and Basili, V. R.: A Meta-model for Software Development Resource Expenditures, *Proc. 5th International Conference on Software Engineering*, pp. 107-116 (Mar. 1981).
4) Walston, C. E. and Felix, C. P.: A Method of Programming Measurement and Estimation, *IBM Syst. J.*, Vol. 16, No. 1, pp. 54-73 (1977).
5) Miyazaki, Y. M. and Mori, K.: COCOMO Evaluation and Tailoring, *Proc. 8th International Conference on Software Engineering*, pp. 292-299 (Aug. 1985).
6) Kemerer, C. F.: An Empirical Validation of Software Cost Estimation Models, *Comm. ACM*, Vol. 30, No. 5, pp. 416-429 (1987).
7) Kitchenham, B. and Taylor, N. R.: Software Cost Models, *ICL Tech. J.*, Vol. 4, No. 1, pp. 73-102 (1984).
8) Albrecht, A. J. and Gaffney, J. E.: Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation, *IEEE Trans. Softw. Eng.*, Vol.

SE-9, No. 6, pp. 639-647 (1983).
9) Itakura, M. and Takayanagi, A.: A Model for Estimating Program Size and Its Evaluation, *Proc. 6th International Conference on Software Engineering*, pp. 104-109 (1982).
10) Chrysler, E.: The Impact of Program and Programmer Characteristics on Program Size, *Proc. National Computer Conference*, pp. 581-587 (1978).
11) Symons, C. R.: Function Point Analysis: Difficulties and Improvements, *IEEE Trans. Softw. Eng.*, Vol. 14, No. 1, pp. 2-11 (1988).
12) 建石雄三：ファンクションポイントによるソフトウェアの機能的生産性評価，情報処理学会ソフトウェア工学研究会資料，37-2 (1984).
13) ソフトウェアの規模見積り手法の調査研究報告書，情報処理振興事業協会技術センター (1989).
14) ソフトウェア開発の生産性尺度，ラージシステム研究会昭和63年度分科会研究活動報告書（第12分冊）(1989).
15) Conte, S. D., Dunsmore, H. E. and Shen, V. Y.: *Software Engineering Metrics and Models*, The Benjamin/Cummings Publishing Company, Inc. (1986).
16) 篠原能材：数値解析の基礎，日新出版 (1987).

(平成元年 8 月 30 日受付)
(平成 2 年 11 月 13 日採録)

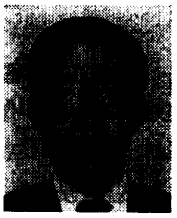
宮崎 幸生（正会員）

昭和23年生．昭和46年早稲田大学理工学部数学科卒業．昭和48年同大学院修士課程修了．同年4月，富士通(株)入社．シミュレーション言語の開発，ソフトウェア開発作業標準の開発，見積り手法および品質管理手法の研究・開発等に従事．ソフトウェアの定量的側面に興味を持つ．



山田 松治（正会員）

昭和26年生．昭和49年大阪市立大学理学部物理学科卒業．同年富士通(株)入社．金融系システム開発，ネットワークシステム開発に従事．定量的なプロジェクト管理手法に興味を持つ．



板倉 稔（正会員）

昭和19年生．昭和42年慶応義塾大学経済学部卒業．同年，富士通(株)に入社．以降，リアルタイムバンキングシステムの開発に従事．平成元年から，システム開発技術の開発・普及に従事．規模見積り，測定尺度などに興味を持つ．