統計数理（1996）
第44巻 第2号 163-179

SAS/IMLによる非線形最適化とその統計学への応用

SAS Institute* 岸 本 淳 司
東京大学** 浜 田 知久馬

（1996年7月 受付）

1. SAS/IMLとは何か

SAS/IMLは，巨大なSASシステムのごく一部のプロダクトである．SASシステムは改めて説明する必要がないほど有名な統計ソフトウェアであって，PC，UNIXからメインフレームまでの環境で動作し，特に医薬関係の統計解析において世界的に標準的な地位を得ている．既成統計手法をツールとして利用するユーザーにとっては，信頼できる統計ソフトウェアの存在はありがたい．しかし，統計手法自体を研究する人にとっては，発想したアルゴリズムをできるだけ容易にプログラムできるソフトウェアが必要である．SAS/IMLはそのような要求に応じる行列演算言語である．同様の要求に対してSとかAPLのようなシステムもあり，それぞれ特徴をもっている．SAS/IMLはSASシステムの統計ソフトウェアとしての著名度と比較すればマイナーな存在であるが，一部のファンからは熱烈に支持されている．今日，たとえばGEEのような新しい有用なアルゴリズムが誕生したとき，統計研究者がSAS/IMLでプログラムを書いて，それを一般に公開して知識を共有することがよく行われている．

SASシステムが初めて商用ソフトウェアとして公開されたのは1976年である．そのときのSASシステムには，行列演算言語PROC MATRIXがシステムの一部として既に備えられていた．このPROC MATRIXが発展して今日のSAS/IMLに進化した．統計パッケージは不完全であることを自ら認識し，行列演算言語を提供して補完するという発想は評価されるべきであろう．

SAS/IMLは1つの変数が行列を表すインタプリタ型言語である．たとえば，

```
a={1  1  1,
   1  2  3,
   2  6  4 };
```

と指定すれば，変数Aに3×3の行列がディメンション宣言なしで代入される．また，式

$$b=(X'X)^{-1}X'y$$

を表現するためには，

```
B=INV(X'*X)*X'*Y;
```

と書けばよい．このように直観的にわかりやすい表現で線形代数の式をそのまま表現できるプ

* 〒104 東京都中央区勝どき1-13-1 イヌイビル・カチドキ8F.
** 医学部 薬剤疫学教室：〒113 東京都文京区本郷7-3-1.

ログラミング言語であるというところにSAS/IMLの魅力がある.

このSAS/IMLに,リリース6.08より非線形最適化の機能が追加された.最適化自体が幅広い応用範囲をもっているが,統計学においても非線形最小二乗法や最尤推定などで重要な役割を果たすことはいうまでもない.先に述べたような行列代数を直接記述できる機能と最適化が容易に実施できる機能が融合したことにより,SAS/IMLは統計研究者にとって非常に魅力あるソフトウェアになったといえる.

2. SAS/IMLの最適化サブルーチン群

一般に最適化とは非線形な関数$f(x_1, x_2, \ldots, x_n)$について,その最小値(または最大値)と,そのときのパラメタの値を求める技法である.パラメタ間の関係に等式または不等式の制約条件を含むこともある.最適化には,問題の性質に応じてさまざまな解法のバリエーションが存在する.共通するアルゴリズムとして,なんらかの方法で求めた初期解からより良い解を反復して推定する逐次近似法が行われる.

SAS/IMLには,次のような最適化サブルーチン群が用意されている.

NLPCG	共役勾配法
NLPDD	ダブルドッグレッグ法
NLPNMS	Nelder-Mead シンプレックス法
NLPNRA	直線探索つき Newton-Raphson 法
NLPNRR	リッジ安定化 Newton-Raphson 法
NLPQN	(双)準 Newton 法 (BFGS, DFP)
NLPQUA	2次最適化
NLPTR	信頼領域法

特に非線形最小二乗問題を解くために,次のサブルーチンがある.有名なGauss-Newton法は,安定性に劣るので採用されていない.

NLPLM	Levenberg-Marquardt 最小二乗法
NLPHQN	ハイブリッド準 Newton 最小二乗法

NLPNMS以外の方法では,導関数の計算が必要である.解析的な導関数を与えることもできるが,与えないときは有限差分により近似される.また,すべての方法で線形式の等式または不等式の制約を課すことができるが,特にNLPNMSとNLPQNでは,パラメタに非線形の制約を課すこともできる.各手法の特徴については,最適化法の教科書(たとえば今野・山下(1978)),あるいは岸本(1993),Hartmann (1994),SAS Institute (1995)等を参照していただきたい.

最適化のための補助サブルーチンとして,次の2つがある.

NLPFDD	有限差分による導関数近似
NLPFEA	制約条件下での許容点探索

NLPFDDは,あるパラメタx_0を与えたときの関数値f・勾配ベクトルg・ヘッセ行列hを返す.たとえば尤度関数を最適化したパラメタ値を与えれば,そのときのヘッセ行列からWaldの分散が容易に計算できる.

3. 簡単な実行例

3.1 最初の例題

最適化の例題として，Rosenbrock 関数

$$f(x)=\frac{1}{2}\{100(x_2-x_1^2)^2+(1-x_1)^2\}$$

を用いよう．この関数は，$x^*=(1,1)$ で最小値 $f(x^*)=0$ をとる．初期値は $x^0=(-1.2,1)$ とする．SAS/IML によるプログラムは図 1，結果は図 2 のようになる．

解説：

・最適化する関数の定義

——最適化する関数を start～finish の構文で記述する．最適化するパラメタを引数として，関数値 f が返るような関数を定義する．

——パラメタが複数ある場合は引数はベクトルになる．関数内で引数ベクトルの要素を引用するには，x[1], x[2] というように要素の番号を指定する．

・最適化サブルーチンの呼び出し

——NLPNRR はリッジ安定化 Newton-Raphson 法のサブルーチンである．

```
proc iml;                          /* Invoke IML */

start rosen(x);                    /* Define Rosenbrock Function */
  f = 0.5*(100*(x[2]-x[1]*x[1])**2+(1-x[1])**2);
  return(f);
finish rosen;

x0 = { -1.2  1 };                  /* Set Starting Values */

call nlpnrr(rc, xr, "ROSEN", x0);  /* Call Newton-Raphson Ridge Optimization */

print rc, xr, (rosen(xr));         /* Print Results */
```

図 1.

```
                    RC
                     3

                   XR
          0.999991  0.999982

                #TEM1001
               4.036E-11
```

図 2. 出力 3.1：Rosenbrock 関数最適化の計算結果．

——3番目の引数に最適化する関数，4番目の引数に初期値を指定する．最適化の結果のパラメタ値は，2番目の引数に入って返ってくる．1番目の引数には最適化がうまくいったかどうかを判定するリターンコードが入って返ってくる．
——オプション指定がない場合，目的関数は最小化される．
——導関数を指定していないので，有限差分近似が利用される．

・最適化の結果
——リターンコードは，最適化が終了した原因を示す．1〜10の正の数のときは，最適化は成功していて，満たした収束規準が識別される．−1〜−10の負の数のときは最適化は失敗していて，失敗した原因が識別される．今回の結果は3であるから，最適化は成功している．
——最適化したときのパラメタ値は，$x=(0.999991, 0.999982)$である．これは正確な値$x=(1,1)$に極めて近い．デフォルトの収束条件では，この程度の誤差が含まれる．
——最適化したときの関数値を求めるには，最適化したパラメタを定義した関数に代入すればよい．今回の例では，$f=4.036\times10^{-11}$であり，正確な値$f=0$に極めて近い．

3.2 オプションの指定

SAS/IMLの最適化サブルーチンには，さまざまなオプションをつけることができる．図3に，1階の解析的導関数を指定し，最適化過程の情報をプリントさせた例を図4，図5にその結果を示す．

解説：

・導関数の与えかた
——最適化する関数の各パラメタについて，解析的導関数を指定することができる．最適化する関数と同様に，start〜finishの形式で関数定義する．

```
proc iml;

start rosen(x);
  f = 0.5*(100*(x[2]-x[1]*x[1])**2+(1-x[1])**2);
  return(f);
finish rosen;

start g_rosen(x);          /* Define Gradient */
  g = {0 0};
  g[1] = -200*x[1]*(x[2] - x[1]*x[1]) - (1-x[1]);
  g[2] =  100*(x[2] - x[1]*x[1]);
  return(g);
finish g_rosen;

x0 = { -1.2  1 };

call nlpnrr(rc, xr, "ROSEN", x0, {0 2}) grd="G_ROSEN";
```

図3.

```
                Newton-Raphson Ridge Optimization
                    Without Parameter Scaling
              Hessian Computed by Finite Differences
                    (Using Analytic Gradient)
                 Number of Parameter Estimates 2

Optimization Start: Active Constraints= 0  Criterion= 12.100
Maximum Gradient Element= 107.800

      Iter rest nfun act   optcrit  difcrit maxgrad   ridge     rho
         1    0    2   0    2.3659   9.7341   2.319       0   1.003
         2    0    7   0    2.0534   0.3126   5.521   2.000   1.839
         .
         .
        20    0   33   0 4.72E-18 1.571E-9  1.6E-9       0   1.000

Optimization Results: Iterations= 20 Function Calls= 34 Hessian Calls= 21
Active Constraints= 0  Criterion= 4.7204231E-18
Maximum Gradient Element= 1.59827E-9 Ridge= 0 Rho= 1
```

図4. 出力3.2：Rosenbrock関数最適化のプリント出力（最適化過程）.

```
NOTE:  ABSGCONV convergence criterion satisfied.

                    Optimization Results

                    Parameter Estimates

-------------------------------------------
  Parameter          Estimate     Gradient
-------------------------------------------
  1   X1             1.000000   -1.5983E-9
  2   X2             1.000000   -7.367E-10

Value of Objective Function = 4.720423E-18
```

図5. 出力3.3：Rosenbrock関数最適化のプリント出力（最適化結果）.

——定義した導関数を，最適化サブルーチン中のgrd＝オプションで指定する．一般に有限差分近似を行うよりも解析的導関数を与えた方が高速に処理できる．

——Newton-Raphson法のように2階の導関数を利用する手法のときは，hes＝オプションでヘッセ行列を指定することもできる．また，1階の導関数だけでも解析的に与えた方が速くなる．

・オプションパラメタの与えかた

——最適化サブルーチンの5番目の引数として，オプションベクトルが指定できる．

——1番めの要素は，最小化か最大化かを決定する．0は最小化（デフォルト），1は最大

化になる．最小二乗問題のときは要素のサイズを指定する．

——2番めの要素は，OUTPUT ウィンドウへのプリントアウト出力の量を指定する．0から5までの値をとる．

0 プリント出力しない．デフォルト．
1 問題の要約と反復過程をレポートする．
2 パラメタの初期推定値と最終推定値も出力する．上の出力例は一部である．
3 満たした反復終了規準も出力する．
4 各反復ごとのパラメタベクトル x も表示する．
5 各反復ごとの勾配ベクトル g も表示する．

——3番めから11番めの要素は，最適化法の細かいチューニングを行う．通常ここを操作する必要はないので，上の例では指定していない．

3.3 最小二乗問題

Rosenbrock 関数は，$f(x)=(1/2)\{f_1^2(x)+f_2^2(x)\}$ の形式で表現できる．これは，要素が2つの最小二乗問題である．最小二乗問題には特にそれに対応した効率よい最適化法があり，SAS/IML でもハイブリッド準ニュートン法と Levenberg-Marquadt 法とが用意されている．図6に，NLPLM サブルーチンを使って Levenberg-Marquadt 法を実行した例を示す．

```
proc iml;

start rosen2(x);
  y = {0 0};
  y[1] = 10*(x[2] - x[1]*x[1]);
  y[2] = 1 - x[1];
  return(y);
finish rosen2;

x0 = {-1.2 1};

call nlplm(rc, xr, "ROSEN2", x0, {2 2});
```

図6.

解説：

- 最小二乗法の各要素 $f_1(x), f_2(x),\ldots$ をベクトルで返す関数を定義し，サブルーチン NLPLM の3番目の引数で指定する．
- オプションパラメタの1番目の変数に要素の数（今の例では2）を指定する．
- ここではヤコビ行列を与えていないので，有限差分近似が行われる．ヤコビ行列を与える場合は jac＝オプションで指定する．

4. 統計手法への適用

4.1 非線形最小二乗法

非線形最適化を統計学で活用する例として，非線形最小二乗法をとりあげよう．最小二乗推定では，実測値と予測値との差(残差)の二乗和を最小化する．たとえば$y=(5\ 3\ 4\ 1\ 1)'$, $x=(1\ 2\ 3\ 4\ 5)'$というデータに対して$y=b_1\exp(b_2\cdot x)+\varepsilon$という非線形なモデルを当てはめることを考える．このときには，次の目的関数を最小化するのが問題となる．

$$ss=\sum_{i=1}^{n}\{y_i-b_1\exp(b_2\cdot x_i)\}^2$$

実際のアプリケーションでは，$\sum$を含む目的関数を加算の形で直接記述するのは不便なので，入力データから目的関数を計算するようにしなければならない．SAS/IMLは行列演算言語なので，目的関数を自在に設定することができる．非線形最小二乗法のプログラムは図7のようになる．

```
proc iml;

start nlreg(b) global(y, x);  /* Access y and x using GLOBAL option */
  r = y - b[1]#exp(b[2]#x);   /* Calculate Residual Vector */
  return(r);
finish nlreg;

y = {5 3 4 1 1}';             /* Dependent Data Vector */
x = {1 2 3 4 5}';             /* Independent Data Vector */

b0 = {1 1};                   /* Set Stating Values */
optn = nrow(y) || 2;          /* Get Sample-Size using NROW function */

call nlplm(rc, br, "NLREG", b0, optn);  /* Levenberg-Marquardt Optimization */
```

図7.

解説：

- 非線形最小二乗法のための関数の作り方
  - ——最適化のための関数は，推定するパラメタを引数にとり，その他の変数を引数に与えてはいけない．データベクトルyとxとを関数に渡すためには，引数ではなくGLOBALというオプションを利用する．
  - ——SAS/IMLは行列演算言語であるから，各要素ごとの演算をまとめてプログラムできる．“#”は，行列変数の対応する要素ごとの乗算を意味する．
  - ——非線形最小二乗法のための目的関数は，各オブザベーションごとの残差のベクトルを返すように作成する．
- 最適化ルーチンの実行
  - ——非線形最小二乗法の初期値は慎重に与えなければいけない．

```
start nlreg2(b) global(y, x);
  r  = y - b[1]#exp(b[2]#x);
  ss = ssq(r);                  /* Calculate SS from Residuals */
  return(ss);
finish nlreg2;

b0 = {1 1};
optn = {0 2};                   /* Minimize function and Print Results */

call nlpnrr(rc, br, "NLREG2", b0, optn);  /* Newton-Raphson Ridge Optimization */
```

図8.

——非線形最小二乗法のためのオプションベクトルの1番目の要素はサンプルサイズである．ここではnrow関数を利用している．

——NLPLMサブルーチンを呼び出して，Levenberg-Marquadt法を実行している．

最小二乗法でLevenberg-Marquadt法を使うのは，安定性と効率を考慮しているからである．もし必要なら，Newton-Raphson法のような通常の非線形最適化を利用することもできる（図8参照）．

通常の最適化ルーチンを使うときには，関数は残差ベクトルを返すのではなく，残差平方和を返すようにしなければならない．オプションベクトルの第1要素は最小化を実行するために0にする．

4.2 最尤推定

最尤推定を行う場合も非線形最小二乗推定と同工異曲である．対数尤度関数さえ定義できれば，導関数の指定も必要なく，ただ最大化を指定するだけでよい．たとえば，$y=(1\ 2\ 3\ 4\ 5)'$

```
proc iml;

start mlnorm(parms) global(y);  /* Log-Likelihood Function */
  logL = -0.5 * sum(log(8*atan(1)*parms[2]) + (y-parms[1])##2/parms[2]);
  return(logL);
finish;

y = {1 2 3 4 5};                /* Data Vector */

parms0 = {1 1};                 /* Initial Values */
optn = {1 2};                   /* Maximization (optn[1]=1) */

call nlpnrr(rc, parmsr, "MLNORM", parms0, optn);
```

図9.

というデータが正規分布 $N(\mu, \sigma^2)$ から発生したとして，パラメタの μ と σ^2 とを最尤推定するには，次の対数尤度関数を最大化することになる．

$$\log L = -\frac{1}{2}\sum_{i=1}^{n}\left\{\log(2\pi\sigma^2)+\frac{(y_i-\mu)^2}{\sigma^2}\right\}$$

SAS/IML で最尤推定を実行するには，対数尤度関数をそのまま IML 言語で記述した目的関数を定義する．たとえば，図 9 のようにする．

解説：

- 尤度関数のパラメタベクトルは，parms[1]$=\mu$, parms[2]$=\sigma^2$ となっている．
- 尤度関数の定義に必要なデータは，最小二乗法のときと同様 GLOBAL オプションで渡す．
- SAS/IML 言語の演算規則：a をスカラー，X を行列（またはベクトル）として
  - ——a＊X, a＋X, X－a のように，スカラーと行列との間の乗加減演算が認められている．スカラーを行列の各要素に対して作用させ，結果は X と同じサイズの行列になる．
  - ——X/a という形式で，行列/スカラーの除算も認められている．これは $(1/a)X$ の意味である．
  - ——X ## a というのは，行列 X の各要素を a 乗するという意味である．
- SAS/IML には π の定数が用意されていないので，$4\tan^{-1}(1)$ で求めている．
- 目的関数を最大化したいので，オプションパラメタベクトルの 1 番目の要素を 1 にする．$\log L$ を最大化するかわりに $-2\log L$ を最小化したいなら，関数定義を修正して，オプションパラメタの 1 番目の要素を 0 にする．

5. 制約つきの問題

5.1 線形制約の例題

一般に非線形最適化で，パラメタ間に等式または不等式の制約をつけることがある．たとえば線形制約 Betts 関数

$$f(x)=0.01x_1^2+x_2^2-100$$

```
proc iml;

start betts(x);
  f = 0.01*x[1]*x[1] + x[2]*x[2] - 100;
  return(f);
finish betts;

con = { 2 -50  .  .,          /* Boundary Constraint */
       50  50  .  .,          /* Boundary Constraint */
       10  -1  1 10 };        /* Linear Inequality Constraint */
x0 = {-1 -1};                 /* Infeasible Starting Values */

call nlpcg(rc, xr, "BETTS", x0, {0 2}, con); /* Conjugate-Gradient Optimization */
```

図 10.

```
NOTE: Initial point was changed to be feasible for boundary and linear
      constraints.

                          Optimization Start
                          Parameter Estimates
-------------------------------------------------------------------------
  Parameter       Estimate   Gradient   Lower BC   Upper BC
-------------------------------------------------------------------------
    1 X1           6.800000    0.13600    2.00000   50.00000
    2 X2          -1.000000   -2.00000  -50.00000   50.00000

                Value of Objective Function = -98.5376

                          Linear Constraints
                          ------------------

    [1]       10.00000 <=  +  10.0000 * X1 -   1.0000 * X2 ( 59.0000 )
```

図11. 出力5.1：線形制約つき問題の初期設定.

```
                  Conjugate-Gradient Optimization
          Automatic Restart Update (Powell, 1977; Beale, 1972)
               Gradient Computed by Finite Differences
                   Number of Parameter Estimates 2
                      Number of Lower Bounds 2
                       Number of Upper Bounds 2
                    Number of Linear Constraints 1

Optimization Start: Active Constraints= 0  Criterion= -98.538
Maximum Gradient Element= 2.000

      Iter rest nfun act   optcrit  difcrit maxgrad   alpha   slope
         1    0    3   0 -99.5468   1.0092   0.135   0.502  -4.018
         2    1    7   1 -99.9600   0.4132 0.00272  34.985 -0.0182
         3    2    9   1 -99.9600 1.851E-6       0   0.500  -74E-7

Optimization Results: Iterations= 3 Function Calls= 10 Gradient Calls= 9
Active Constraints= 1  Criterion= -99.96 Maximum Gradient Element= 0
Slope= -7.39836E-6

NOTE:  ABSGCONV convergence criterion satisfied.
```

図12. 出力5.2：線形制約つき問題の反復過程.

では，境界制約

$$2 \leq x_1 \leq 50$$
$$-50 \leq x_2 \leq 50$$

と，線形制約

$$10x_1 - x_2 \geq 10$$

とをもっている．この関数は，$x^* = (2, 0)$ で最小値 $f(x^*) = -99.96$ をとる．初期値としては，制約を満たしていない点 $x^0 = (-1, -1)$ を用いる．線形制約のついた最適化問題を共役勾配法

```
                    Optimization Results
                     Parameter Estimates
-------------------------------------------------------------
  Parameter          Estimate     Gradient   Active BC
-------------------------------------------------------------
  1   X1             2.000000      0.04000    Lower BC
  2   X2         -1.24027E-10            0

          Value of Objective Function = -99.96

        Linear Constraints Evaluated at Solution
        ----------------------------------------
[1]     10.0000 * X1 -   1.0000 * X2 -  10.0000 = 1.000000E+01
```

図 13. 出力 5.3：線形制約つき問題の計算結果.

で解く例を図 10 に，結果を図 11・12・13 に示す.

解説：

・線形制約の指定のしかた

——線形制約は，(一般制約数＋2)×(パラメタ数＋2) のサイズの行列で指定する.

——制約行列の第 1 行は境界制約の下限を，第 2 行は境界制約の上限を，3 行以降は一般制約を表す.

——制約行列の第 1 列は 1 番目のパラメタ，第 2 列は 2 番目のパラメタ…と順に対応し，右から 2 番目の列が一般制約についての関係演算子，1 番右の列が一般制約の右辺の定数に対応する.

$$
\begin{array}{lcccc}
 & x_1 & x_2 & \text{関係演算} & \text{定数} & \\
 & \downarrow & \downarrow & \downarrow & \downarrow & \\
\text{BLC}=\{ & 2 & -50 & . & ., & \leftarrow\text{下限} \\
 & 50 & 50 & . & ., & \leftarrow\text{上限} \\
 & 10 & -1 & 1 & 10\ \}; & \leftarrow\text{一般制約}
\end{array}
$$

——関係演算は，$=\rightarrow 0$, $\geq\rightarrow 1$, $\leq\rightarrow -1$ にそれぞれ対応する.

——境界制約については，右の 2 列は無関係である.

——制約行列は，最適化サブルーチンの 6 番目の引数に指定する.

・制約を満たさない初期値が与えられた場合，制約を満たす初期値を自動的に探索する.

5.2 非線形制約の与えかた

統計学の文脈では頻度は多くないが，パラメタ間に非線形の等式または不等式による制約を課したいことがある. SAS/IML の最適化法群のなかで，NLPQN (準ニュートン法) と NLPNMS (Nelder-Mead シンプレックス法) の 2 つは非線形制約を与えることができる.

たとえば，単位円の内側 $x_1^2+x_2^2\leq 1$ の領域の中で目的関数 $f(x_1, x_2)=x_1x_2$ を最大化することを考えよう. プログラムは図 14 のようになる.

解説：

・非線形等式制約と不等式制約は，次の関数形式で指定する.

```
proc iml;

start fprod(x);
  f = x[1] * x[2];
  return(f);
finish fprod;

start unicirc(x);  /* Nonlinear Inequality Constraint */
  c = 1 - x*x';
  return(c);
finish unicirc;

x0 = {1 1};
optn = {1 2 . . . . . 1 0};

call nlpqn(rc, xr, "FPROD", x0, optn) nlc="UNICIRC";
```

図14.

$$c_i(x)=0, \quad i=1,\ldots,nec,$$
$$c_i(x)\geq 0, \quad i=nec+1,\ldots,nc.$$

たとえば，$x_1^2+x_2^2\leq 1$ の制約は，$c_1(x)=1-x_1^2-x_2^2\geq 0$ という関数で表現できる．

・非線形制約の数を nc 個，そのうち等式制約の数を nec 個とする．

——非線形制約を示す関数は，nc 個の要素のベクトルを返すよう定義する．そのうち1個めから nec 個までは等式制約であり，$nec+1$ 個めから nc 個めまでが不等式制約になる．

——非線形制約の数 nc はオプションパラメタベクトルの10番めの要素で，非線形等式制約の数 nec はオプションパラメタの11番めの要素で指定する．

・非線形制約を定義した関数は，NLPQN あるいは NLPNMS サブルーチン中の nlc＝オプションで指定する．

6. ワイブル生存関数の例題

6.1 ワイブル分布の最尤推定

非線形最適化を統計学へ応用した例題として，打ち切りを含む生存時間データに2パラメタのワイブル分布をあてはめたプログラムを紹介する．記号の定義はすべて Lawless (1982) に従う．

数値例は，Lawless (1982) に記載されているもので，ラットの膣癌についての実験室研究によるデータである．ラットの膣に発癌物質 DMBA を塗布し，carcinoma ができるまでの日数を記録した．その結果，19匹中17匹のラットは観測期間中に carcinoma が発生した．イベントまでの時間のデータを，次のようにベクトルに収める．ただし，最後の2つのオブザベーションは打ち切りデータである（図15参照）．

2パラメタのワイブル分布の確率密度関数は次のようになる．

```
proc iml;

t = { 143 164 188 188 190 192 206
      209 213 216 220 227 230 234
      246 265 304 216 244 };
```

図15.

```
r = 17;

start weib(x) global(t, r);
  logL = r*log(x[2]) - r*x[2]*log(x[1])
       + (x[2]-1)*sum(log(t[1:r])) - sum((t/x[1])##x[2]);
  return(logL);
finish weib;
```

図16.

$$\frac{\beta}{\alpha}\left(\frac{t}{\alpha}\right)^{\beta-1}\exp\left[-\left(\frac{t}{\alpha}\right)^{\beta}\right] \qquad t\geq 0$$

ここで，t は時間変数であり，$\alpha>0$ は尺度パラメタ，$\beta>0$ は形状パラメタと呼ばれるパラメタである．

ワイブル分布の対数尤度関数は次のようになる．

$$\log L(\alpha,\beta)=r\log\beta-r\beta\log\alpha+(\beta-1)\sum_{i\in D}\log t_i-\sum_{i=1}^{n}\left(\frac{t_i}{\alpha}\right)^{\beta}$$

ここで，r は打ち切りを受けていないオブザベーション数である．この関数を最大化するように α と β とを決めればよい．第3項の $\sum_{i\in D}\log t_i$ は，打ち切りを受けていないオブザベーションについて総和を計算するという意味である．対数尤度関数は図16のようにプログラムできる．結果を図17・18・19に示す．

解説：

- 対数尤度関数の第3項にある1：rという表現は，1から r までの整数列を発生させている．
- $\sum$ の計算を行うとき，できるだけDOループを使わないでSUM関数を使うようにすべきである．SAS/IMLはインタプリタ言語なので，DOループの実行は遅い．また，SUM関数を使った方がプログラムも見やすくなる．

準備が整ったので，対数尤度関数の最大化を実行する．ここでは初期値を $\alpha_0=0.5, \beta_0=0.5$ とし，準ニュートン法（DBFGS）により最適化を行う．なお，反復過程でパラメタの値が一時的にでも負の数となると対数尤度中の対数関数の計算ができなくなるため，α も β も 10^{-6} を下限とするように制約をつけておく．

```
x0 = { 0.5 0.5 };
optn = { 1 2 };
con = { 1E-6 1E-6,      /* Lower Bounds */
        .    . };

call nlpqn(rc, xr, "WEIB", x0, optn, con);
```

図17.

```
                    Dual Quasi-Newton Optimization
        Dual Broyden - Fletcher - Goldfarb - Shanno Update (DBFGS)
                Gradient Computed by Finite Differences
                    Number of Parameter Estimates 2
                       Number of Lower Bounds 2
                        Number of Upper Bounds 0

Optimization Start: Active Constraints= 0  Criterion= -444.533
Maximum Gradient Element= 2248.275

      Iter rest nfun act   optcrit  difcrit maxgrad   alpha   slope
         1    0    3   0 -166.6679    277.9   261.8  0.0122  -51962
         2    0    4   0 -157.0344   9.6335   105.1   1.000 -14.623
         .
         .
        19    0   39   0  -88.2327 1.79E-10 5.39E-7   1.000 -39E-11

Optimization Results: Iterations= 19 Function Calls= 40 Gradient Calls= 30
Active Constraints= 0  Criterion= -88.232735
Maximum Gradient Element= 5.3856E-7 Slope= -3.94315E-10

NOTE:  GCONV convergence criterion satisfied.
```

図18. 出力6.1：最尤推定の反復過程.

```
                    Optimization Results
                     Parameter Estimates
-----------------------------------------------------------
   Parameter          Estimate     Gradient   Active BC
-----------------------------------------------------------
   1   X1           234.318611   2.43162E-8
   2   X2             6.083148    5.3856E-7

        Value of Objective Function = -88.23273515
```

図19. 出力6.2：最尤推定によるパラメタ推定値.

解説：

・準ニュートン法は19回の反復で収束した．収束過程に問題は見られない．

・パラメタの推定値は，$\hat{\alpha}=234.3$, $\hat{\beta}=6.083$ となった．これは Lawless (1982) が示してい

```
call nlpfdd(f, g, h, "WEIB", xr);

estimate = xr`;
cov = -inv(h);
stderr = sqrt(vecdiag(cov));
upper_b = estimate + probit(0.975)*stderr;
lower_b = estimate - probit(0.975)*stderr;

print "MLE and Asymptotic Confidence Bounds",
      estimate stderr upper_b lower_b;
```

図 20.

```
MLE and Asymptotic Confidence Bounds

 ESTIMATE    STDERR   UPPER_B   LOWER_B
234.31861 9.6458629 253.22416 215.41307
6.0831481 1.0682468 8.1768734 3.9894229
```

図 21. 出力 6.3：NLPFDD による信頼区間.

る値と等しい.

- ここでは対数尤度関数の導関数は指定しなかった. ワイブル分布では解析的導関数を得ることもできるが, 一般に尤度関数の導関数を解析的に求めることはかなり難しい.
  導関数を指定しなかった場合, 2 階の導関数を用いる Newton-Raphson 法や信頼領域法では, 関数呼び出しだけを使った有限差分近似でヘッセ行列を計算することになるため, 計算量が大きくなる可能性がある. 導関数を指定しないのなら, 1 階導関数だけを使う準ニュートン法を用いる方がよいであろう.

6.2 漸近正規近似による信頼区間

パラメタを最尤推定したとき, その信頼区間を求めたいことがある. そのためには, NLPFDD というサブルーチンを使うと便利である. NLPFDD (f, g, h, "関数", x) は, "関数" とあるパラメタ値 x を与えると, 関数値 f, 勾配 g, ヘッセ行列 h を返す. 最適化したパラメタ値を与えれば, その地点でのヘッセ行列の逆行列により, パラメタ間の分散共分散行列を求めることができ, それにより漸近正規近似による信頼区間を容易に得ることができる (図 20・21 参照).

正規近似に基づく信頼区間は, サンプルサイズが小さいとき不正確である. SAS/IML のマニュアルには, プロファイル尤度に基づく信頼区間の計算法も例題として掲載されている.

7. おわりに

統計研究者がよいアルゴリズムを発見したとしても, それが一般に利用されなければ価値が少なくなる. SAS/IML とその非線形最適化ルーチンを使うことにより, 研究者にとって本質的

な関心に集中して現実のプログラムを作成することができる．また，高度な統計手法を広く伝える方法を確保することができる．統計学あるいは線形代数学の教育についても，このような行列演算言語を前提として実践的教育を行うと効果が大きいのではなかろうか．

参 考 文 献

Hartmann, M. W. (1994)． NLP プロシジャと SAS/IML ソフトウェアによる非線形最適化の生物統計学への応用（岸本淳司 訳)，日本 SAS ユーザー会論文集，25-72.

岸本淳司 (1983)． 非線形最適化を行う SAS の新プロシジャNLP の紹介，オペレーションズ・リサーチ，**38**，142-145.

今野 浩，山下 浩 (1978)．『非線形計画法』，日科技連，東京.

Lawless, J. F. (1982). *Statistical Models and Methods for Lifetime Data*, Wiley, New York.

SAS Institute (1995). *SAS/IML Software*: *Changes and Enhancements through Release 6.11*, SAS Institute, Cary, North Carolina.

Nonlinear Optimization with SAS/IML and Its Applications to Statistics

Junji Kishimoto

(SAS Institute Japan)

Chikuma Hamada

(Faculty of Medicine, University of Tokyo)

SAS/IML is a matrix language in the SAS system. Some nonlinear optimization techniques are available in release 6.08 or later. We will demonstrate their usage and capabilities with statistical examples, such as nonlinear least squares problem or maximum likelihood estimation.

Key words: SAS/IML, optimization, MLE.