連載講座

行列演算用言語LAMAX-S(2)

本郷 茂，八巻 直一，内田 智史

1. はじめに

前回では、LAMAX-Sの概要と比較的簡単なLAMAX-Sプログラムについて紹介しましたが、今回は、引き続きLAMAX-Sのプログラムを通して、よりきめ細かい文法を説明して行きます。

今回からは、行列表現の数式をそのままLAMAX-Sで書き下すための仕方を示すために、本格的な3つのプログラム例で紹介します。

教育的な配慮のために、効率を考えずに行列の数式をそのままLAMAX-Sで表現しています。しかし、我々は、LAMAX-Sの最終目標として、高度な最適化機能を実現し、極力効率を落とさないように動作することを狙っています。したがって、今回ここで紹介するプログラムはすべて将来的なLAMAX-Sの最適化機能を期待して、特に効率化のための考慮は払わず、記述性を重視しています。

まず、最初に、固有値・固有ベクトルをヤコビ法で求めるプログラムを紹介します。後半のプログラム例で紹介する多変量解析では、固有値を求めることになりますが、最初からライブラリを利用するよりは実際に自分で固有値を求めるアルゴリズムを確認するためのプログラミングが必要となるでしょう。ここで紹介するプログラムリストを見ていただければ、LAMAX-Sのプログラムと実際のアルゴリズムとの一致性を確認していただけると同時に、プログラミングが不慣れな人でも教科書(教科書はたくさんありますが、今回のプログラムでは[1])を見ながら容易にコーディングできることが分かるでしょう。次に多変量解析の中から、重回帰分析と主成分分析を取りあげ、LAMAX-Sでの基本的な統計計算の仕方から多変量解析の具体的なプログラミングを示します。

なお、LAMAX-Sの文法は、1992年の日本OR学会事例研究奨励賞ソフトウェア部門授賞後も、フルセットLAMAX-S開発に向けての改訂作業が行われています。ここに示すプログラムは、全て改訂されたLAMAX-Sの新文法に沿ってコーディングされたものであり、そのままでは、LAMAX-S(PC98版)では動作しないことをお断りしておきます。実際に動作させるためには、新しい文法に対応させる簡易コンバータが用意されていますので、必要な方に配布できるよう準備しています。もちろん、プログラムロジックなどは、LAMAX-S(PC98版)を用いてチェックしてあります。

ほんごう しげる 専修大学
〒214 川崎市多摩区三田2－1－1
やまき なおかず ㈱システム計画研究所
うちだ さとし 神奈川大学

2. ヤコビ法による固有値・固有ベクトル

固有値・固有ベクトルは、LAMAX-Sの組込み関数でも求めることができますが、本節ではLAMAX-Sの例題として、実対称行列の全固有値とそれに対する固有ベクトルを求めるヤコビ法のプログラムを紹介します。このプログラムは、メインプログラムと固有値・固有ベクトルを求めるサブルーチン(JcbMtd)、およびいくつかの補助サブルーチンからなります。メインプログラムでは、(1)単にデータをセットして、(2)サブルーチンJcbMtdを呼び出して固有値・固有ベクトルを計算し、(3)その値を表示しているだけです。実際の計算は、サブルーチンJcbMtdで行われ、その内容は、次のヤコビ法のアルゴリズムに従っています。行列Aの固有値・固有ベクトルを求めるとします。

ステップ0　t=0として，固有ベクトルの初期点を決める。EVCの初期値として単位行列をセットする。

ステップ1　t=t+1として，行列Aの非対角要素の中で絶対値最大のものを探す。

$$max\ a_{ij} (i \neq j)$$

ここでiが行、jが列の番号を示す。

ステップ2　回転角θの計算と座標変換行列Pの作成

$$\cos\theta = \frac{1}{\sqrt{2}}\sqrt{1+\frac{\varepsilon\alpha}{\beta}}, \quad \sin\theta = \frac{\varepsilon a_{IJ}}{2\beta\cos\theta}$$

ただし,

$$\alpha = \frac{a_{ii}-a_{ij}}{2}, \quad \beta = \sqrt{\alpha^2 + a_{ij}^2}$$

$$\varepsilon = sign(a_{ii} - a_{jj})$$

これを用いて，次の座標変換のための行列Pを作成する。PはP_{ii}, P_{jj}以外の対角部分は1.0で、それ以外の要素は0の行列ですが、$P_{ii} = \cos\theta, P_{ij} = \sin\theta, P_{ji} = -\sin\theta, P_{jj} = \cos\theta$となる行列です。

ステップ3　行列Aの更新と固有ベクトルの更新

$$A = PAP^T、\qquad EVC = EVC \cdot P^T$$

$\boxed{\text{ステップ 4}}$ 収束チェック

(1) 行列 A が数値的に十分対角行列になったら正常終了 (固有値は A の対角部分)

(2) 反復回数が n^2 を越えたら終了 (失敗)

$\boxed{\text{ステップ 5}}$ ステップ 1 へ戻る。

```
c      ヤコビ法：固有値・固有ベクトル（メインルーチン）
       parameter(n=4)            <* 行列の次元数
       parameter(eps=1e-6)       <* 計算誤差の許容値
       parameter(epssub=1e-3)    <* 一致比較の許容値
       parameter(maxcnt=20)      <* 最大繰り返し回数
c
       real:matrix[n,n:symmetric] A <* 求める行列
       real:matrix[n,n] EVC         <* 固有ベクトル
       real:vector[n] EVL           <* 固有値
c
       A = | 1, 2, 3,  4 ! <* 固有値・固有ベクトルを
     -     ! 2, 5, 6,  7 ! <* 求める行列のセット
     -     ! 3, 6, 8,  9 !
     -     ! 4, 7, 9, 10 |
c
       call JcbMtd( AA, EVL, EVC, n, eps, ierr)
c
       call mprint(EVL)  <* 固有値の表示
       call mprint(EVC)  <* 固有ベクトルの表示
c
       stop
       end

c      ヤコビ法：固有値・固有ベクトル（サブルーチン）
       subroutine JcbMtd( A, EVL, EVC, n, eps, ierr)
c
c 注意 (1) 行列 A の固有値・固有ベクトルを求める
c       (2) 行列 A は，対称行列である。
c       (3) 行列 A は，破壊される。
c
       integer n                          <* 行列の次元数
       real:matrix[n,n:symmetric] A <* 求める行列
       real:vector[n] EVL     <* 行列 A の固有値
       real:matrix[n,n] EVC   <* 行列 A の固有ベクトル
       real eps          <* 計算誤差の許容値
       integer ierr      <* エラーフラグ 0:正常、1:収束せず
c
       real:matrix[*,*:transform] P <* 座標変換行列
       real alpha, beta, es  <* θ計算用変数
       logical isdiag        <* 対角行列かどうかチェック
       integer t             <* 繰り返し数
c
c========= ステップ 0 初期設定
       EVC = 1::diagonal[n]
       t = 0
c
   10 continue
c========= ステップ 1 非対角要素の中で絶対値最大のものを探す。
           t = t + 1
           i = asmaxr(A, -diagonal)
           j = asmaxc(A, -diagonal)
c========= ステップ 2 回転角θの計算と座標変換行列 P の作成
           alpha = ( A[i,i]-A[j,j])/2
           beta = sqrt(alpha**2+A[I,J]**2)
           es = sign( 1.0e0, A[i,i]-A[j,j])
           P = 1::diagonal[n]
           P[i,i] = sqrt((1+es*alpha/beta)*0.5)
           P[i,j] = (es*A[i,j])/(2*beta*P[i,i])
           P[j,i] = -P[i,j]
           P[j,j] = P[i,i]
c========= ステップ 3 行列 A の更新と固有ベクトルの更新
           A = P*A*P'
           EVC = EVC*P'
c========= ステップ 4 収束チェック
           if( isform(A, diagonal, eps)) go to 20
           if(t.gt.n*n) go to 30 <* 反復回数>n^2 なら終了
c========= ステップ 5 ステップ 1 へ戻る。
       go to 10
c
c========= ステップ 6 正常終了
   20 continue
       ierr = 0
       EVL = A<0> <* 固有値のセット
       return
c========= ステップ 7 異常終了
   30 continue
       ierr = 1
       return
       end
```

LAMAX-S でのサブルーチンの使い方は、FORTRAN に完全に準拠しています。すなわち、行列 (ベクトルを含む) をサブルーチンに引き渡す場合には、その構造、要素の型、数学的性質 (正定値など) が完全に一致していなければなりません。

サブルーチン JcbMtd では、変数 A, EVL, EVC が寸法 n 行 n 列の行列として宣言されています。寸法 n は、親のルーチンから受け取るので、この宣言は FORTRAN の整合配列に似ています。

座標変換行列 P の宣言で、* は P の寸法を意味します。LAMAX-S では、このように宣言された行列を動的行列と呼び、プログラムの実行中に寸法が変化します。これに対し、寸法が明示された行列を静的行列と呼び、寸法がプログラムの実行中変化しません。静的行列の寸法は定数式でなくてはならないので、P の寸法の宣言に n が記述できないのです (n は仮引数です)。A, EVL, EVC は仮引数であるので n を寸法として記述できます。そこで P は、n にどんな値が代入されても良いように、動的行列として宣言されています。動的行列は、データが代入された時点か allocate 文によって明示的に指定された時点で記憶領域が割り当てられます。free 文によって明示的に領域を解放することもできます。

また、`P[*:transform]` の `transform` は、これが座標変換行列であることを意味します。これについては後述します。`P = 1::diagonal[ n ]` は、P に単位行列をセットすることを意味しています。LAMAX-S では、「値::構造」という表現を同値要素行列と呼び、これは要素がすべて「値」のその構造をした行列を意味します。たとえば、`0::matrix[2, 3]` は 2 行 3 列のゼロ行列を意味します。`1::matrix[10, 10 :upper_tri]` は 10 行 10 列で上三角部分の要素だけが 1 であとの要素はゼロの行列を意味します。

asmaxr 関数は、絶対値が最大の行列要素の値のある行の位置を返します。asmaxr 関数には色々な制限や条件を付けることができます。asmaxr(A,-diagonal) は「対角要素を除く」という条件です。asmaxc 関数は同様に列の位置を返します。asmaxr と asmaxc が別々に記述されていても、LAMAX-S の最適化機能によって最大値の検索は 1 度しか行われませんので、効率は低下しません。

isform 関数は、与えられた行列の非ゼロの配置が特定の構造になっているかどうかを判断します。isform(A, diagonal, eps) は、行列 A が与えられた計算機イプシロンの中で対角行列になっているかどうかを判断します。

`A<0>` という表現は、行列 A の対角部分を取りだし、そ

れをベクトルとして扱うことを意味します。`EVL=A<0>`は、行列Aの対角部分をベクトルEVLに代入することを意味します。

このプログラムの特徴は、座標変換行列Pにあります。教科書の記述と全く同じに記述するという目的でプログラムを作成したので、Pという座標変換行列を作成した訳ですが、実際には行列の回転を行うだけであり、計算量はそれほど多くありません。`P*A*P'`という計算をその記述どうりまじめに行うと、乗算が2回必要となってしまいます。しかし、座標変換行列という指定をすることで、LAMAX-Sプリプロセッサの最適化機能によって無駄な計算がなくなります。

3. 統計の初歩の計算と重回帰分析・主成分分析

多変量解析の教科書の解説で代表的な重回帰分析と主成分分析を題材にして，それぞれの解析のアルゴリズムを行列表現のままLAMAX-Sでプログラムしてみましょう。ただし，おことわりしておきますが，この節ではプログラムの記述性を重視しましたので，実行効率や数値計算上での誤差が少なくなるような工夫はしてありません。実際に，アプリケーションプログラムを作成するときのLAMAX-Sでの工夫の仕方は別の機会にします。

まず最初に，統計の初歩の計算である1変量の平均・分散などの基本的な表現から，多変量を扱う場合での分散共分散行列や相関係数行列をいかにしてLAMAX-Sで表現したらよいかについて説明します。

1変量での平均は，n個のデータを成分とするn次元ベクトル$\mathbf{x} = (x_1, x_2, x_3, \ldots, x_n)^t$の列ベクトルと，n個の1を各要素とするn次元ベクトル$\mathbf{1}_n = (1, 1, \cdots, 1)$の行ベクトルとを用いて表現することができます。その数式をLAMAX-Sでは次のように記述します。

平均　$\overline{x} = \mathbf{1}_n \cdot x/n$

```
real:vector[n] X
real           Xm
Xm = 1::rvector[n]*X/n
```

あるいは，行和をもとめるLAMAX-Sの組込み関数を用いて記述することもできます。

```
Xm = rowsum(X)/n
```

各要素が全て1のベクトルを，LAMAX-Sでは

```
1::vector[n],  1::rvector[n]
```

と記述します。ここでのvectorは列ベクトルを，rvectorは行ベクトルを意味しています。

同様に1変量の分散は，n個の1を各要素とするn次元ベクトル$\mathbf{1}_n = (1, 1, \cdots, 1)^t$の列ベクトルを用いて偏差の2乗和で表現します。

分散 $S^2 = \frac{1}{n}(\mathbf{x} - \overline{x} \cdot \mathbf{1}_n)^t(\mathbf{x} - \overline{x} \cdot \mathbf{1}_n)$

```
real:vector[n] X
real           Xm,S2
Xm = 1::rvector[n]*X/n
S2 = (X-Xm*1::vector[n])'*(X-Xm*1::vector[n])
```

`1::vector[n]*Xm`の記述で全ての要素が平均値となるn次元列ベクトルが計算でき，その値をn次元ベクトルxから引くことで偏差になります。

次に，データがn個でp変数の多変量の場合の平均と分散共分散行列について表してみましょう。多変量の平均と分散共分散行列は，1変量の場合のベクトルの表現が行列に変わりますが，同じ様な記述により表すことができます。いま，多変量のデータ$\mathbf{X}$を

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}$$

の$(n \times p)$行列とすると，平均はp次元ベクトルとなり分散共分散は$(p \times p)$行列$\mathbf{S}$となります。

平均$\overline{x} = \frac{1}{n}\mathbf{1}_n\mathbf{X}$

分散 $\mathbf{S} = \frac{1}{n}(\mathbf{X} - \mathbf{1}_{n\times p}\overline{x}d)^t(\mathbf{X} - \mathbf{1}_{n\times p}\overline{x}d)$

ただし，$\mathbf{1}_{n\times p}$は全ての要素が1の$(n \times p)$行列で，$\overline{x}d$は対角要素が平均値の$(n \times p)$対角行列です。

$$\overline{x}d = \begin{pmatrix} \overline{x}_1 & & & \\ & \overline{x}_2 & 0 & \\ 0 & & \ddots & \\ & & & \overline{x}_p \end{pmatrix}$$

上式の$\mathbf{X} - \mathbf{1}_{n\times p}\overline{x}d$は，各変数の偏差を計算しています。次式でも分散共分散行列を求めることもできます。$\overline{x}$はp次元行ベクトルです。

$$S = \frac{1}{n}\mathbf{X}^t\mathbf{X} - \overline{x}^t\overline{x}$$

平均値ベクトルと分散共分散行列をLAMAX-Sで記述してみましょう。

```
real:matrix[n,p]              X
real:matrix[p,p:diagonal]  Xmd
real:matrix[p,p]              S
Xmd<0> = (1::rvector[n]*X)'
S = (X - 1::matrix[n,p]*Xmd)'*
-    (X - 1::matrix[n,p]*Xmd)/n
```

あるいは

```
real:matrix[n,p]              X
real:rvector[p]               Xm
real:matrix[p,p]              S
Xm = rowsum(X)/n
S  = X'*X/n - Xm'Xm
```

今までの節でもLAMAX-Sの文法を述べていますが，多少解りにくい箇所がありますので，この平均値を対角行列に持つ変数Xmdについて LAMAX-S の記述の仕方を説明します。Xmdは対角行列として宣言しておくと対角以外の要素は全て0とみなして演算してくれます。LAMAX-Sで注意すべきことは，対角の要素に平均値ベクトルの要素を代入するとき，`1::rvector[n]*X` で求めた平均値（行ベクトル）を列ベクトルにしてからでないと対角行列の対角要素に代入することはできません。`Xmd = (1::rvector[n]*X)'` と記述すると，その演算結果のXmdは，次のような $(p \times p)$ 対角行列になります。

$$\begin{pmatrix} \overline{x}_1 & & & \\ & \overline{x}_2 & 0 & \\ 0 & & \ddots & \\ & & & \overline{x}_p \end{pmatrix}$$

ここで，転置の' を付けないと正しく実行されません。各変数の平均偏差は，`X - 1::matrix[n,p]*Xmd` で求めています。`1::matrix[n,p]` は，全ての要素が1の $(n \times p)$ 行列を表しています。したがって，`1::matrix[n,p]*Xmd` の演算結果は，

$$\mathbf{X} = \begin{pmatrix} \overline{x}_1 & \overline{x}_2 & \cdots & \overline{x}_p \\ \overline{x}_1 & \overline{x}_2 & \cdots & \overline{x}_p \\ \vdots & \vdots & & \vdots \\ \overline{x}_1 & \overline{x}_2 & \cdots & \overline{x}_p \end{pmatrix}$$

の $(n \times p)$ 行列となります。

統計の初歩の計算の最後として，p変数の相関係数行列をLAMAX-Sで表現してみましょう。相関係数は，

$$r_{ij} = \frac{s_{ij}}{\sqrt{s_{ii}}\sqrt{s_{jj}}}$$

で求めます。なお、ここで、s_{ij}は変数i,jの共分散、s_{ii}, s_{jj}は変数i,jの分散です。このr_{ij}を要素とする$(p \times p)$相関係数行列Rは，$\sqrt{s_{11}}, \cdots, \sqrt{s_{pp}}$を対角要素とする$(p \times p)$対角行列$D$を用いて，次のように表現するとができます。

$$R = D^{-1}SD^{-1}$$

この式をLAMAX-Sで記述してみましょう。

```
real:matrix[n,p]              X
real:matrix[p,p]              S
real:matrix[p,p:diagonal]  D
real:matrix[p,p]              R
real:vector[*]                DiagSqrt
S = X'*X/n - rowsum(X)'*rowsum(X)
D<0> = DiagSqrt(S)
R = D**(-1)*S*D**(-1)
```

ここで，DiagSqrtは分散共分散行列の対角要素だけを取りだしその平方根を求め結果を列ベクトルとして戻す関数です。この結果は，各変数の標準偏差となっています。DiagSqrt関数は，LAMAX-Sの組み込み関数ではありません。各自で作成してみてください。この他に，データを規準化（平均0，分散1）してから相関係数行列を求めることもできます。具体的な記述については，ここでは省略しますが，実際のプログラム例は主成分分析の最初の部分に表現してありますので参考にしてください。

多変量解析の基礎的な数式を行列表現で記述した本も多く出版されています [2,3]。また，FORTRAN,BASICなどのプログラミング言語で書かれたプログラムリストも一緒に掲載されている本もありますので参考にされるとよいでしょう [3]。

さて，今まで書いてきました平均，分散，相関係数行列のLAMAX-Sプログラムを使って，重回帰分析と主成分分析プログラムを作ってみます。どちらも，１００行程度のプログラムで表現できています。計算方法の詳細についてはここでは述べません。具体的には，参考文献 [2, 3] などをご覧ください。LAMAX-Sで記述したプログラムは，参考文献 [2] の内容を基に作成してみました。

重回帰分析と主成分分析プログラムの説明は，紙面の都合で詳しくできませんが，プログラム内に注釈を入れておきましたので参考にしてください。

例示したプログラムでは，次のような内容の計算が行われています。

重回帰分析のプログラム概要

- データの平均，分散共分散行列，相関係数行列
- データの中から規準変数（目的変数 y）と説明変数（X）を選び重回帰分析
- 偏回帰係数の推定値の計算：正規方程式 $\mathbf{X}^t\mathbf{X}b = \mathbf{X}^t y$ の係数 b を連立一次方程式を解く solve 文を用いて求めます。
- 重相関係数，推定値 b の標準偏差，t 値

主成分分析のプログラム概要

- データの規準化，相関係数行列
- 固有値，固有ベクトル
- EIGENVV プログラムリストは省略しています。
- 主成分負荷量，主成分得点

4. おわりに

第1回では，FORATAN と LAMAX-S のプログラムを対比しながら例示しましたが，今回の第2回目では，LAMAX-S の文法を含めてより具体的な応用として行列の固有値解法と多変量解析のプログラムを例示しました。プログラムの説明では多少不足しているところもありますが，LAMAX-S で記述したプログラムの記述性を見ていただけたかとおもいます。私共は，この LAMAX-S を用いて教育の現場で優れた教育効果があげられるよう使っていただきたいと願っております。次回は，本連載の最終回となります。最終回では，数理計画の事例を紹介して，さらに，今後の LAMAX-S の将来構想についても述べたいとおもいます。

参考文献

1. 戸川隼人：詳解数値計算演習、共立出版、1980
2. 竹内啓　他：多変量解析の基礎，東洋経済新報社，1972
3. 柳井春夫　他：多変量解析ハンドブック，現代数学社，1986


```
C---  重回帰分析
      real*8:matrix[*,*]   Xr     <* データ
      real*8:rvector[*]    MEAN   <* データの平均値
      real*8:vector[*]     SD     <* データの標準偏差
      real*8:matrix[*,*]   COVAR  <* データの分散共分散行列
      real*8:matrix[*,*]   CORR   <* データの相関係数行列
      real*8:matrix[*,*]   D      <* 作業用変数
      real*8:vector[*]     Y      <* 基準変数
      real*8:matrix[*,*]   X      <* 説明変数
      real*8:vector[*]     B      <* 偏回帰係数（β）の推定値(b)
      real*8:vector[*]     YE     <* 基準変数の予測値
      real*8:vector[*]     UE     <* 残差
      real*8               SGM2   <* 残差の分散
      real*8               SGM    <* 残差の標準偏差
      real*8:matrix[*,*]   VARB   <* b の分散
      real*8:vector[*]     SDB    <* b の標準偏差
      real*8:vector[*]     Tv     <* β i=0 検定の t 値
      real*8               Fv     <* β 2=…=β p=0 検定の F 値
      real*8               R2     <* 決定係数
      real*8               R      <* 重相関係数
      real*8               RF2    <* 自由度調整済み決定係数
      real*8               RF     <* 自由度調整済み重相関係数
      integer            IS(100)  <* 作業用変数
      real*8:vector[*]   diagsqrt <* 対角要素の平方根を求める関数
      integer            N,P,P0
c
      read(*,*) N,P                 <* N:観測データの数、P:変数の数
      allocate Xr:matrix[N,P]
      read(*,*) ((Xr[i,j],j=1,P),i=1,N)
      MEAN  = rowsum(Xr)/dble(N)              <* 平均の計算
      COVAR = Xr'*Xr/dble(N) - MEAN'*MEAN <* 分散共分散行列の計算
      SD    = diagsqrt(COVAR,P)               <* 標準偏差の計算
      D     = 0.0d0::matrix[P,P]
      D<0>  = SD
      CORR  = (D**(-1))*COVAR*(D**(-1))   <* 相関係数行列の計算
      write(*,*) 'データ'
      call mprint(Xr)
      write(*,*) '平均値, 標準偏差'
      call mprint(|MEAN',SD|)
      write(*,*) '分散共分散'
      call mprint(COVAR)
      write(*,*) '相関係数'
      call mprint(CORR)
c
1000  continue
      write(*,*) '--------------------------------------'
      write(*,*) '重回帰分析:'
      write(*,*) '  基準変数の番号:'
      read (*,*)          NDEP
      if(NDEP.le.0) goto 2000
      write(*,*) '  分析用に取り込む説明変数の個数と番号:'
      read (*,*)          P0,(IS(I),I=1,P0)
      write(*,*) '- - - - - - - - - - - - -'
      write(*,*) '  基準変数の番号:',NDEP
      write(*,*) '  分析用に取り込んだ説明変数の番号:',(IS(I),I=1,P0)
      Y=Xr[1:N,NDEP] <* 説明変数の抜き取り
      P=P0+1
      X=1.0d0::matrix[N,P]        <* 1カラム目は定数項とする
      do 20 i=1,P0
        X[1:N,i+1] = Xr[1:N,IS(i)]    <* 説明変数の抜き取り
20    continue
      solve (X'*X) * B = X'*Y <* 偏回帰係数βの推定値bの計算
      YE = X*B                <* 基準変数の予測値の計算
      UE = Y - YE             <* 残差の計算
      R2 =    (YE'*Y - (rowsum(Y)**2)/dble(N))
     -      / ( Y'*Y - (rowsum(Y)**2)/dble(N))    <* 決定係数
      R  = SQRT(R2)                               <* 重相関係数
      RF2=1.0d0 - dble(N-1)/dble(N-P)*(1.0d0-R2) <* 自由度調整
      RF  = DSQRT(RF2)             <* 自由度調整済み重相関係数
      SGM2= UE'*UE / dble(N-P)     <* 残差の分散
      SGM = sqrt(SGM2)             <* 残差の標準偏差
      VARB= SGM2/(X'*X)            <* b の分散
      SDB = diagsqrt(VARB,P)       <* b の標準偏差
      Tv  = B % SDB                <* t 値
      Fv  =(R2/dble(P-1))/((1.0d0-R2)/dble(N-P)) <* F 値
      write(*,*) '説明変数; 観測値の数 (N), 変数の数 (P):',N,P
      call mprint(X)
      write(*,*) '偏回帰係数βの推定値b, bの標準偏差, t 値'
      write(*,*) '                    t 値の自由度 (N-P)=',N-P
      call mprint(|B,SDB,Tv|)
      write(*,*) '決定係数                      : ',R2
      write(*,*) '重相関係数                    : ',R


      write(*,*) '自由度調整済み決定係数   : ',RF2
      write(*,*) '自由度調整済み重相関係数 : ',RF
      write(*,*) 'F 値                         : ',Fv
      write(*,*) 'F 値の自由度 (P-1,N-P)    : ',P-1,N-P
      write(*,*) '残差の分散                  : ',SGM2
      write(*,*) '残差の標準偏差             : ',SGM
      write(*,*) '基準変数, 基準変数の予測値, 残差'
      call mprint(|Y,YE,UE|)
      goto 1000
2000  continue
      end
```

```
c---- 主成分分析
      integer n,p                       <* データ数,変数の数
      real*8:matrix[*,*] Xr             <* データ
      real*8:matrix[*,*] Xp             <* 偏差
      real*8:matrix[*,*] Xd             <* 基準化
      real*8:rvector[*]  MEAN           <* 変数の平均値
      real*8:matrix[*,*] S              <* 変数の標準偏差
      real*8:matrix[*,*] Md             <* 作業用変数
      real*8:matrix[*,*] Sd             <* 作業用変数
      real*8:matrix[*,*] Cov            <* 分散共分散行列
      real*8:matrix[*,*] R              <* 相関係数行列
      real*8:vector[*]   CRATE          <* 寄与率
      real*8:vector[*]   CUMCR          <* 累積寄与率
      real*8:matrix[*,*] FScore         <* 主成分得点
      real*8:matrix[*,*] FLoad          <* 主成分負荷量
      real*8:vector[*]   EVAL           <* 固有値
      real*8:matrix[*,*] EVEC           <* 固有ベクトル
      real*8:vector[*]   CUM            <* 固有値の累積
      real*8:matrix[*,*] D              <* 固有値の平方根（対角行列）
      real*8:matrix[*,*] CHIDF          <* カイ2乗値と自由度
      real*8:vector[*]  diagsqrt        <* 対角要素の平方根を求める関数
      real*8   ws,rm1,rm2,rm3,nb        <* 作業用変数
      integer            pm,pz,m        <* 作業用変数
c
      call minput(Xr)
      n = irow(Xr)
      p = icol(Xr)
      MEAN  = rowsum(Xr)/dble(n)              <* 平均値
      Md = 0::matrix[p,p]
      Md<0> = MEAN'
      Xp  = Xr - 1::matrix[n,p]*Md            <* 偏差
      Cov =  Xp'*Xp/dble(n)                   <* 分散共分散行列
      S   = diagsqrt(Cov,p)                   <* 標準偏差
      Sd = 0::matrix[p,p]
      Sd<0> = S
      Xd = Xp*(Sd**(-1))                      <* 変数の基準化
      R  = Xd'*Xd/dble(n)                     <* 相関係数行列
      EVAL = 0.0d0::vector[p]
      EVEC = 0.0d0::matrix[p,p]
      call EIGENVV(R,p,EVAL,EVEC)             <* 固有値,固有ベクトル
      CUM = 1::lower_tri[p]*EVAL              <* 固有値の累積
      CRATE = EVAL/CUM[p]                     <* 寄与率
      CUMCR = CUM/CUM[p]                      <* 累積寄与率
      D = 0.0d0::matrix[p,p]
      pm = 1
      pz = 1
      do 10 i=1,p
         if(EVAL[i].gt.0.999999d0) pm = i <* 主成分の抽出個数
         if(EVAL[i].gt.0.0d0)      pz = i <* 0より大きい固有値の数
         D[i,i] = sqrt(max(0.0d0,ws))     <* 固有値の平方根
10    continue
      FLoad  = EVEC * D                       <* 主成分負荷量
      FScore = Xd * FLoad[*,1:pm]
     -    * ( D[1:pm,1:pm]'*D[1:pm,1:pm])**(-1)  <* 主成分得点
      CHIDF=0.0d0::matrix[p,2]
      do 20 m=0,pz-1
        rm1=0
        rm2=0
        do 30 j=m+1,pz
          rm1=rm1+log(EVAL[j])
          rm2=rm2+EVAL[j]/(p-m)
30      continue
        rm3=0
        do 40 j=1,m
          rm3=rm3+rm2**2/(EVAL[m]-rm2)**2
40      continue
        nb=n-m-1-dble(2*(p-m)**2+p-m+2)/dble(6*(p-m))+rm3
        CHIDF[m+1,1]=-nb*rm1+nb*(p-m)*log(rm2) <* カイ2乗値
        CHIDF[m+1,2]=(p-m-1)*(p-m+2)/2         <* 自由度
20    continue
      write(*,*) '相関係数行列'
      call mprint(R)
      write(*,*) '固有値,寄与率,累積寄与率,カイ2乗値,自由度'
      call mprint(|EVAL,CRATE,CUMCR,CHIDF|)
      write(*,*) '正規化された固有ベクトル'
      call mprint(EVEC)
      write(*,*) '主成分負荷量'
      call mprint(FLoad)
      write(*,*) '1以上の固有値に対応する主成分得点'
      write(*,*) '                1以上の固有値の個数:',pm
      call mprint(FScore)
      end
```