岡本安晴　2001.1.31；2001.8

行列の計算－１

加減乗算・逆行列・行列式・階数

行列とベクトル

数値を次の形

$$\begin{pmatrix} 3 & 1 & 2 \\ 5 & 2 & 7 \end{pmatrix}$$

のように、長方形の形に並べたものは行列と呼ばれている[1]、[2]。行列は( )あるいは[ ]で囲んで表す。行列における横の並びを行、縦の並びを列という。上の行列の場合、2つの行

$$(3\ 1\ 2) \quad と \quad (5\ 2\ 7)$$

あるいは、３つの列

$$\begin{pmatrix} 3 \\ 5 \end{pmatrix}、\quad \begin{pmatrix} 1 \\ 2 \end{pmatrix} と \begin{pmatrix} 2 \\ 7 \end{pmatrix}$$

から成り立っていると考えられる。

上のように、２行３列からなる行列は(2,3)型の行列[1]、あるいは２×３行列[2]というように呼ぶ。

一般に、m行n列からなる行列は(m,n)型の行列、あるいはm×n行列などという。

行列の１つ１つの数値は、行列の成分[1]あるいは要素[3]と呼ばれている。第i行、第j列にある成分は(i,j)成分などと呼ぶ。成分は、数値を値とする変数、あるいは関数のときもある。

行列Aの(i,j)成分を a_{ij} で表わすと、

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ & \vdots & & \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

と書ける。あるいは、簡単に

$$A = (a_{ij})$$

と書くこともある。

一般に、行列は大文字、成分は小文字で表わされる。

（m,1）型の行列は縦ベクトル、あるいは列ベクトルということがある。ベクトルは太

い小文字で表わされる。列ベクトルxの(i,1)成分をx_iで表わすと

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix}$$

となる。

(1,n)型の行列は横ベクトル、あるいは行ベクトルと呼ばれることがある。行ベクトルyの(1、j)成分をy_jで表わすと

$$\mathbf{y} = (y_1 y_2 \cdots y_n)$$

となる。

行列と配列

行列をプログラムで扱うときは配列が用いられる。

(m,n)型の行列Aは、配列ａを

```
var a : array[1..m,1..n] of Extended;
```

と宣言しておけば、行列Aの(i,j)成分a_{ij}を配列ａの要素 a[i,j]で表わすことによって、行列Aを配列ａで表わすことができる。

(n,1)型の列ベクトルa、あるいは(1,n)型の行ベクトルbは、１次元の配列を

```
var  a, b : array[1..n] of extended;
```

と宣言しておいて、列ベクトルaの(i,1)成分a_iに配列ａの要素a[i]、行ベクトルbの(1,i)成分b_iに配列ｂの要素 b[i]を対応させると、列ベクトルaは配列ａで、行ベクトルbは配列ｂで表わすことができる。

いろいろな型（サイズ）の行列を扱うときは、最大のサイズ、すなわち、行数と列数の最大値あるいはそれより大きい値で配列を宣言しておくと、プログラムでの型の管理が簡単になる。例えば、列数および行数の最大値を NDim とするとき、行列を表わす配列の型を

```
type TMat = array[1..NDim,1..NDim] of Extended;
```

と宣言しておき、行列を表わす配列は

```
var a, b, c : TMat;
```

というように宣言しておく。このとき、Ndim を超えない正整数 m と n に対して、a は(m,n)型の行列を表すことができる。a の要素 a[i,j]の添字は$1 \le i \le m$、$1 \le j \le n$の範囲の値となり、配列の左上の部分が行列の大きさに合わせて用いられる。

行列の特殊なものとしての行（列）ベクトルは、２次元の配列ではなく１次元の配列を用意して用いる方が自然である。すなわち、

```
type  TVector = array[1..NDim] of Extended;
```

というように宣言する。

行列の加減算

２つの行列が同じ型のときは、それらの和あるいは差が定義される。和あるいは差は、対応する位置にある要素同士の和と差で与えられる。行列

$$\begin{pmatrix} 3 & 1 & 2 \\ 5 & 2 & 7 \end{pmatrix}$$

と次の行列

$$\begin{pmatrix} 5 & 7 & 1 \\ 6 & 2 & 3 \end{pmatrix}$$

との和は

$$\begin{pmatrix} 3 & 1 & 2 \\ 5 & 2 & 7 \end{pmatrix}+\begin{pmatrix} 5 & 7 & 1 \\ 6 & 2 & 3 \end{pmatrix}=\begin{pmatrix} 3+5 & 1+7 & 2+1 \\ 5+6 & 2+2 & 7+3 \end{pmatrix}=\begin{pmatrix} 8 & 8 & 3 \\ 11 & 4 & 10 \end{pmatrix}$$

で、差は

$$\begin{pmatrix} 3 & 1 & 2 \\ 5 & 2 & 7 \end{pmatrix}-\begin{pmatrix} 5 & 7 & 1 \\ 6 & 2 & 3 \end{pmatrix}=\begin{pmatrix} 3-5 & 1-7 & 2-1 \\ 5-6 & 2-2 & 7-3 \end{pmatrix}=\begin{pmatrix} -2 & -6 & 1 \\ -1 & 0 & 4 \end{pmatrix}$$

で与えられる。

一般に、(m,n)型の行列

$$A=(a_{ij}) \quad と \quad B=(b_{ij})$$

の和および差は、

$$A+B=(a_{ij}+b_{ij})$$

および

$$A - B = (a_{ij} - b_{ij})$$

で与えられる。

行列AとBが配列ａとｂで表わされているとき、その和および差を表わす配列ｃおよびｄは以下のようにして求めることができる。

＜和＞

```
for i:=1 to m do
    for j:=1 to n do
        c[i,j]:=a[i,j]+b[i,j];
```

＜差＞

```
for i:=1 to m do
    for j:=1 to n do
        d[i,j]:=a[i,j]-b[i,j];
```

スカラー積

行列に対して数値をスカラーという。数値ｓと行列Aの積sAはスカラー積といい、各成分をｓ倍したものである。すなわち、

$$sA = s(a_{ij}) = (sa_{ij})$$

で与えられる。

したがって、積sAを表わす配列をｐで表わすと、ｐの各要素の値を設定するプログラムは次のようになる。

＜スカラー積＞

```
for i:=1 to m do
    for j:=1 to n do
        p[i,j]:=s*a[i,j];
```

内積と長さ（ノルム）

行ベクトルｘと列ベクトルｙの積は内積ｘｙとして以下のように与えられる。

$$\mathbf{xy} = \begin{pmatrix} x_1 x_2 \cdots x_n \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n$$

行列Aの(i,j)成分を(j,i)成分とする行列はＡの転置行列といい、A'、A^tあるいはA^Tなどで表わされる。この記号を用いると、行ベクトルｘあるいは列ベクトルｙの自分自身との積

が次のように与えられる。

$$\mathbf{xx}' = (x_1 \cdots x_n)(x_1 \cdots x_n)' = (x_1 \cdots x_n)\begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = x_1^2 + \cdots + x_n^2$$

$$\mathbf{y}'\mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}'\begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = (y_1 \cdots y_n)\begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = y_1^2 + \cdots + y_n^2$$

行（列）ベクトルの自分自身との積の平方根をベクトルの長さといい、記号 $\|\ \|$ で表わす。

$$\|\mathbf{x}\| = \sqrt{\mathbf{xx}'}\ 、\quad \|\mathbf{y}\| = \sqrt{\mathbf{y}'\mathbf{y}}$$

長さの概念を一般的に拡張したものはノルムと呼ばれている。$\|\mathbf{x}\|$ や $\|\mathbf{y}\|$ はノルムともいう。

行列の積

行列 A と B の積は、A の列数と B の行数が等しいときに定義されている。

行列 A が(p、q)型、B が(q、r)型であるとする。行列 A および B を、それぞれ行ベクトルおよび列ベクトルで区切って表わす。すなわち、

$$A = \begin{pmatrix} a_{11}a_{12}\cdots a_{1q} \\ \hline a_{21}a_{22}\cdots a_{2q} \\ \hline \vdots \\ \hline a_{p1}a_{p2}\cdots a_{pq} \end{pmatrix} = \begin{pmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \vdots \\ \mathbf{a}_p \end{pmatrix}$$

および

$$B = \left(\begin{array}{c|c|c|c} b_{11} & b_{12} & & b_{1r} \\ b_{21} & b_{22} & \cdots & b_{2r} \\ \vdots & \vdots & & \vdots \\ b_{q1} & b_{q2} & & b_{qr} \end{array}\right) = (\mathbf{b}_1\mathbf{b}_2\cdots\mathbf{b}_r)$$

というように、行ベクトル $\mathbf{a}_i$ および列ベクトル $\mathbf{b}_k$ を並べたものとして表わす。ここで、

$$\mathbf{a}_i = \left(a_{i1} a_{i2} \cdots a_{iq}\right) \quad \text{および} \quad \mathbf{b}_j = \begin{pmatrix} b_{1j} \\ b_{2j} \\ \vdots \\ b_{qj} \end{pmatrix}$$

である。

このとき、AとBの積Cは(p、r)型の行列となり、その(i,j)成分c_{ij}は行ベクトル$\mathbf{a}_i$と列ベクトル$\mathbf{b}_j$との内積で与えられる。すなわち、

$$AB = \begin{pmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \vdots \\ \mathbf{a}_p \end{pmatrix} \left(\mathbf{b}_1 \mathbf{b}_2 \cdots \mathbf{b}_r\right) = \begin{pmatrix} \mathbf{a}_1\mathbf{b}_1 & \mathbf{a}_1\mathbf{b}_2 & \cdots & \mathbf{a}_1\mathbf{b}_r \\ \mathbf{a}_2\mathbf{b}_1 & \mathbf{a}_2\mathbf{b}_2 & \cdots & \mathbf{a}_2\mathbf{b}_r \\ & & \vdots & \\ \mathbf{a}_p\mathbf{b}_1 & \mathbf{a}_p\mathbf{b}_2 & \cdots & \mathbf{a}_p\mathbf{b}_r \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1r} \\ c_{21} & c_{22} & \cdots & c_{2r} \\ & & \vdots & \\ c_{p1} & c_{p2} & \cdots & c_{pr} \end{pmatrix}$$

ここで、

$$c_{ij} = \mathbf{a}_i \mathbf{b}_j$$

である。

上の積の(i,j)成分を与える式は次のようにも書ける。

$$c_{ij} = \sum_{k=1}^{q} a_{ik} b_{kj}$$

したがって、行列AとBの積Cを求めるプログラムは次のようになる。

```
for i:=1 to p do
    for j:=1 to r do
        begin
            c[i,j]:=0.0;
            for k:=1 to q do
                c[i,j]:=c[i,j]+a[i,k]*b[k,j];
        end;
```

ここで、行列A,BおよびCを配列a、bおよびcで表わしている。

行列の演算のためのユニットファイルUMatCalc.pas

行列の加減算や積を計算するための関数をユニットファイルUMatCalc.pasに宣言した。

UMatCalc.pass には加減算と積を求める関数の他に、この解説の末尾にあるリスト１に挙げられている関数や手続きが interface 部に宣言されている。

行列は TMatCalc 型として宣言されていまる。TMatCalc 型はユニットファイル UTypeDefMat.pas において次のように宣言されている。

```
const   NDimMat = 100;
type    TMatCalc = array[1..NDimMat,1..NDimMat] of Extended;
```

NDimMat は、プログラム中で扱う行列のサイズ以上の値に設定しておく。多種類のサイズの行列を用いるときは、最大のサイズ以上であるようにしておく。

ユニット UTypeDefMat は、ユニット UMatCalc の関数あるいは手続きを使用するユニットの uses 節でも使用を宣言しておく。

ユニット UMatCalc の関数を用いて配列ａとｂの和を求めるときは

```
c := MatAdd( a, b, m, n : Longint );
```

配列ｄとｂの積を求めるときは

```
e := MatMul( d, b, L, m, n : Longint );
```

というようにする。ここで、ａ、ｂ、ｃ、ｄ および ｅ は、TMatCalc 型の配列で、それぞれ(m,n)型、(m,n)型、(m,n)型、(L,m) 型 および (L,n) 型の行列を表わすものとする。TMatCalc 型の配列のサイズ NDimMat は、ｍ，ｎ，Ｌ以上の数を設定しておく。

リスト１に挙げられている関数の使用例を、プロジェクト PCheckMat.dpr として用意した。このプロジェクトの実行時のフォームは図１のようになっている。上２つのグリッドに行列の値を設定してボタンをクリックすると、クリックしたボタンに対応する計算がリスト１の関数を用いて行われる。

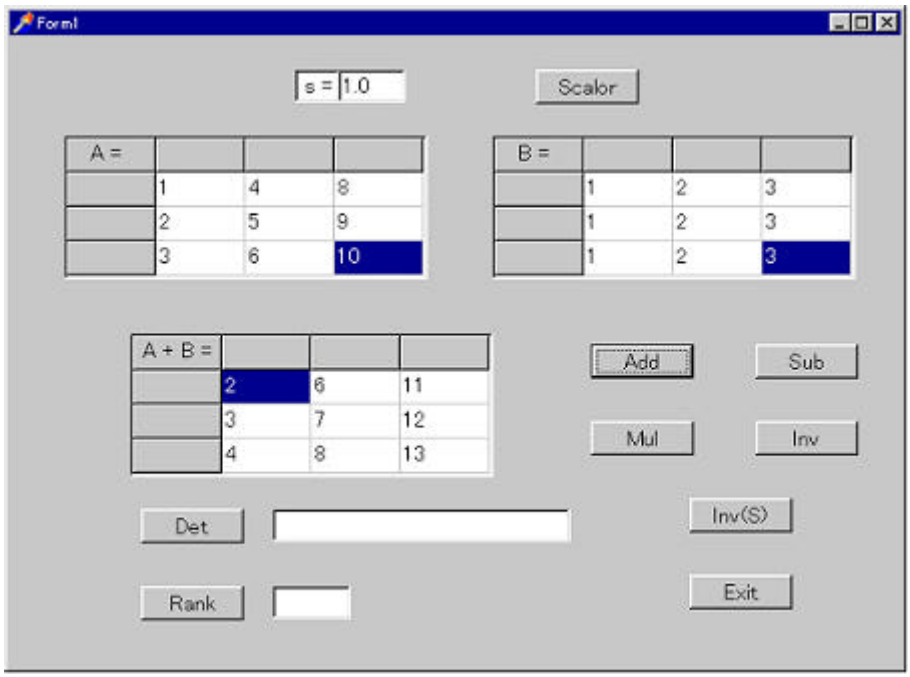

図１　プロジェクト PCheckMat.dpr の実行時のフォーム

ボタン「Add」、「Sub」、「Mul」をクリックすると、それぞれ和、差、積が求められて下のグリッドに表示される。図１は「Add」ボタンをクリックした場合である。

「Scalar」ボタンをクリックすると、フォーム上部の「ｓ＝」の右側のエディット・コンポーネントに設定した数値と、左上の「A=」と表示のあるグリッドに設定された行列とのスカラー積が計算されて、下のグリッドに表示される。

「Inv」、「Inv(S)」、「Det」、「Rank」ボタンのクリックで、それぞれ逆行列（ガウスの消去法)、逆行列（Cholesky 分解の利用)、行列式、階数の計算が行われる。これらについては後で説明する。

３次元の回転

行列の積を用いた例として、３次元空間での回転を表わすプログラムを作成した。

回転は、オイラーの角と呼ばれる３つの角、q 、f 、y 、によって次のように表わすことができる[(1)]。

Y 軸の回りの角 θ の回転を Y_q 、Z 軸の回りの角 f および角 y の回転を Z_f および Z_y で表わすと、回転 T は３つの回転の積

$$T = Z_f Y_q Z_y$$

として書き表すことができる。すなわち、

$$T=\begin{pmatrix}\cos f & -\sin f & 0\\ \sin f & \cos f & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}\cos q & 0 & \sin q\\ 0 & 1 & 0\\ -\sin q & 0 & \cos q\end{pmatrix}\begin{pmatrix}\cos y & -\sin y & 0\\ \sin y & \cos y & 0\\ 0 & 0 & 1\end{pmatrix}$$

と書ける。

プログラム PRotation.dpr では、オイラーの角から上式によって回転 T を求め、Image コンポーネントに描かれている立方体を回転 T によって回転している。実行開始時のフォーム（図２）において、オイラーの角を設定してから「GO」ボタンをクリックする。

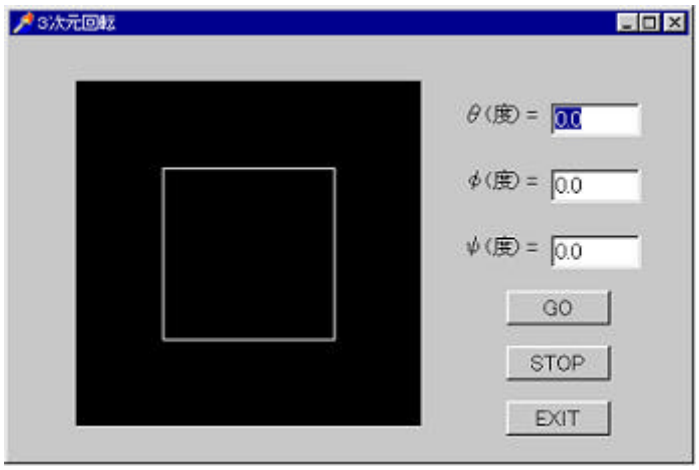

図２　PRotation.dpr の実行開始時の画面

設定されたオイラーの角に基づいて回転 T が計算され、OnTimer イベントの生起によって回転 T による立方体の回転と回転後の立方体の再描画が行われる。

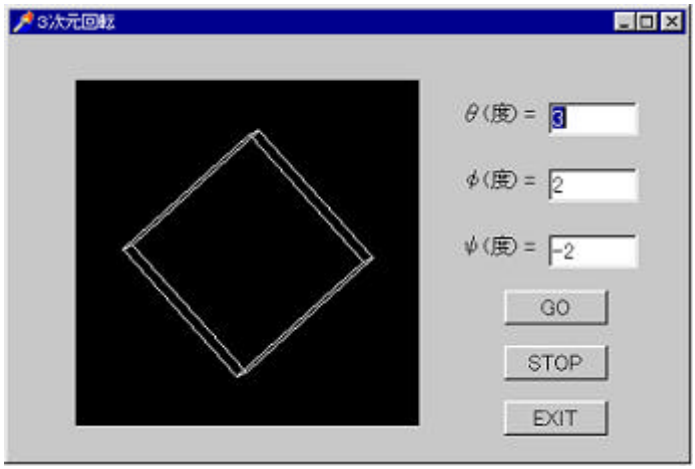

図３　回転中の立方体

OnTimer イベントは「STOP」ボタンがクリックされるまで設定された interval で生起し続けるので、立方体もそれに合わせて回転を続けることになる（図３）。

行列式と階数

(n,n)型の行列は正方行列という。

正方行列

$$A=\begin{pmatrix} a_{11}a_{12}\cdots a_{1n} \\ \vdots \\ a_{n1}a_{n2}\cdots a_{nn} \end{pmatrix}=\left(\mathbf{a}_1\mathbf{a}_2\cdots\mathbf{a}_n\right)$$

に対して、次式

$$\det(\mathbf{a}_1\mathbf{a}_2\cdots\mathbf{a}_n)=\sum_s \operatorname{sgn} s\cdot a_{s(1)\cdot 1}a_{s(2)\cdot 2}\cdots a_{s(n)\cdot n}$$

で与えられる値を行列 A の行列式という。ここで、

$$\{s(1),s(2),\cdots,s(n)\}$$

は

$$\{1,2,\cdots,n\}$$

を並べかえたものであり、$\sum_s$ は、この並べかえ方の全ての方法についての和であることを表わす。$\operatorname{sgn} s$ は、並べかえ方 s を、対(i,j)の並べかえの組み合わせ（積）で表わしたとき、偶数個の対の並べかえの積で表わされるとき＋１、奇数個の対の並べかえの積で表わされるとき－1 の値をとるものである。並べかえ方 s は、偶数個の対の並べかえの積で表わされるか、奇数個の対の並べかえの積で表わされるかのいずれかである。ある並べかえ方 s が偶数個の対の積で表わされたり奇数個の対の積で表わされたりすることはない。偶数個の対の積で表されるとき、他の対の組み合わせの積で表しても常に偶数個の積になる。奇数個の場合も同様である。

行列式は $\det(\mathbf{a}_1\mathbf{a}_2\cdots\mathbf{a}_n)$ で表わしたり、$|A|$ で表わしたりする。

行列式の絶対値 $\left|\det(\mathbf{a}_1\mathbf{a}_2\cdots\mathbf{a}_n)\right|$ は、ベクトル $\mathbf{a}_1$、$\mathbf{a}_2$、・・・、$\mathbf{a}_n$ で張られる平行体の体積（面積）を表わす。行列 A が（２，２）型のときは、２つのベクトル $\mathbf{a}_1$ と $\mathbf{a}_2$ で張られる平行四辺形の面積を表す。（３，３）型のときは、３つのベクトル $\mathbf{a}_1$、$\mathbf{a}_2$ と $\mathbf{a}_3$ で張られる平行六面体の体積を表わす。行列式の符号はベクトルが右手系であれば正、左手系であれば負となる。

行列 A が（２，２）型であるとき、$\mathbf{a}_1$ と $\mathbf{a}_2$ が同一直線上にあれば $\mathbf{a}_1$ と $\mathbf{a}_2$ で張られる平行四辺形の面積は０である。行列 A が（３，３）型のとき、$\mathbf{a}_1$、$\mathbf{a}_2$ と $\mathbf{a}_3$ が同じ平面上にあれば体積は０である。

行列 A の列ベクトル$\mathbf{a}_1$、$\mathbf{a}_2$、・・・、$\mathbf{a}_n$がｋ次元の空間内にある（ｋ次元空間を張る）とき、行列 A の階数はｋであるといい、

$$r(A)=k$$

などと表わします(2)。

行列 A の階数$r(A)$がｋであるとき、行列

$$A=(\mathbf{a}_1\mathbf{a}_2\cdots\mathbf{a}_n)=\begin{pmatrix}\mathbf{b}_1\\ \mathbf{b}_2\\ \vdots\\ \mathbf{b}_n\end{pmatrix}$$

の行ベクトル$\mathbf{b}_1$、$\mathbf{b}_2$、・・・、$\mathbf{b}_n$もｋ次元空間内にある（ｋ次元空間を張る）。すなわち、行列の列ベクトルと行ベクトルの張る空間の次元数は同じで、その次元数が階数である。

行列式と階数を求める関数 MatDet と MatRnk をユニットファイル UMatCalc.pas に用意した。MatDet のヘッダーは次のようになっている。

```
function  MatDet( a    : TMatCalc;
                  n    : Longint;
                  zero : Extended ) : Extended;
```

第１パラメータａに行列式を求める行列を表わす配列、第２パラメータｎに行列の型(n,n)の値ｎを設定する。第３パラメータ zero は計算結果をゼロとみなす基準値である。これは、MatDet では行列式を掃出し法を用いて求めているが、そのときに成分の絶対値がこの値 zero より小さいときは０であると判定している。

階数を求める MatRnk のヘッダーは次のようになっています。

```
function MatRnk( a : TMatCalc;
                 m, n  : Longint;
                 ZeroV : Extended ) : Longint;
```

行列 A の階数は、行列が正方行列でないときも列ベクトル、あるいは行ベクトルの張る空間の次元数として与えられる。行列の型(m,n)を表す２つの数値、行数ｍと列数ｎ、を第２、第３パラメータに設定する。０と判定するときの基準値は第４パラメータ ZeroV に設定する。

MatDet および MatRnk の使用例もプロジェクト PCheckMat.dpr に用意してある。図１のフォームで「Det」ボタンをクリックすると、「A =」の表示のある左上のグリッドに設定された行列の行列式が「Det」ボタンの右側のエディットコンポーネントに表示される。「Rank」

ボタンのクリックのときも同様に、「A =」の表示のある左上のグリッドに設定された行列の階数が「Rank」ボタンの右側のエディットコンポーネントに表示される。

逆行列

(n,n)型の正方行列 A の階数が n であるとき、(n,n)型の正方行列Bが存在して

$$BA = AB = I \qquad (1)$$

が成り立つ。ここで、

$$I = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ & & \ddots & & \\ 0 & \cdots & 0 & 1 & 0 \\ 0 & \cdots & 0 & 0 & 1 \end{pmatrix}$$

である。I のように、行列の対角線上の成分のみが 1 で、他の成分は 0 である行列は単位行列と呼ばれている。（1）式における B を A の逆行列といい、A^{-1} で表す。（1）式を A^{-1} を用いて書き表すと

$$A^{-1}A = AA^{-1} = I$$

となる。

(n,n)型の行列 A の階数が n で逆行列が存在するとき、Aは正則行列であるという。

いま、単位行列 I を次のように行ベクトルに分けて表す。

$$I = \begin{bmatrix} \mathbf{e}_1 \\ \vdots \\ \mathbf{e}_n \end{bmatrix}$$

ここで、$\mathbf{e}_i$ は i 番目の要素が 1 で、他の要素は 0 である行ベクトルを表す。

次の $\mathbf{e}_i$ を用いて表される 3 つのタイプの(n,n)型の正則行列 $P_n(i, j)$、$Q_n(i;c)$、$R_n(i, j;c)$ は基本行列と呼ばれている。

$$P_n(i,j)=\begin{bmatrix}\mathbf{e}_1\\ \vdots\\ \mathbf{e}_{i-1}\\ \mathbf{e}_j\\ \mathbf{e}_{i+1}\\ \vdots\\ \mathbf{e}_{j-1}\\ \mathbf{e}_i\\ \mathbf{e}_{j+1}\\ \vdots\\ \mathbf{e}_n\end{bmatrix}、\quad Q_n(i;c)=\begin{bmatrix}\mathbf{e}_1\\ \vdots\\ \mathbf{e}_{i-1}\\ c\mathbf{e}_i\\ \mathbf{e}_{i+1}\\ \vdots\\ \mathbf{e}_n\end{bmatrix}、\quad R_n(i,j;c)=\begin{bmatrix}\mathbf{e}_1\\ \vdots\\ \mathbf{e}_{i-1}\\ \mathbf{e}_i+c\mathbf{e}_j\\ \mathbf{e}_{i+1}\\ \vdots\\ \mathbf{e}_n\end{bmatrix}$$

すなわち、$P_n(i,j)$ は単位行列 I の第 i 行と第 j 行を入れ換えたものである。

$$P_n(i,j)=\begin{bmatrix}1&&&&&&&&&\\ &\ddots&&&&&&&&\\ &&1&&&&&&&\\ &&&0&\cdots&1&&&&\\ &&&&1&&&&&\\ &&&\vdots&\ddots&\vdots&&&&\\ &&&&&1&&&&\\ &&&1&\cdots&0&&&&\\ &&&&&&1&&&\\ &&&&&&&\ddots&&\\ &&&&&&&&1\end{bmatrix}$$

$Q_n(i;c)$ は単位行列 I の(i, i)成分を c で置き換えたものである。

$$Q_n(i;c)=\begin{pmatrix}1&&&&&&\\ &\ddots&&&&&\\ &&1&&&&\\ &&&c&&&\\ &&&&1&&\\ &&&&&\ddots&\\ &&&&&&1\end{pmatrix}$$

$R_n(i,j;c)$ は単位行列 I の(i, j)成分を c で置き換えたものである。

$$R_n(i,\,j;c)=\begin{pmatrix}1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & \cdots & c & & \\ & & & \ddots & \vdots & & \\ & & & & 1 & & \\ & & & & & \ddots & \\ & & & & & & 1\end{pmatrix}$$

基本行列$P_n(i,\,j)$を行列Aの左から掛けたもの$P_n(i,\,j)A$は行列Aのｉ行とｊ行を入れ換えたものになる。すなわち、

$$P_n(i,\,j)A=P_n(i,\,j)\begin{pmatrix}\mathbf{a}_1\\ \vdots\\ \mathbf{a}_i\\ \vdots\\ \mathbf{a}_j\\ \vdots\\ \mathbf{a}_n\end{pmatrix}=\begin{pmatrix}\mathbf{a}_1\\ \vdots\\ \mathbf{a}_j\\ \vdots\\ \mathbf{a}_i\\ \vdots\\ \mathbf{a}_n\end{pmatrix}$$

となる。

基本行列$Q_n(i;c)$を行列Aの左から掛けたものは行列Aのｉ行をｃ倍したものになる。すなわち、

$$Q_n(i;c)A=Q_n(i;c)\begin{pmatrix}\mathbf{a}_1\\ \vdots\\ \mathbf{a}_{i-1}\\ \mathbf{a}_i\\ \mathbf{a}_{i+1}\\ \vdots\\ \mathbf{a}_n\end{pmatrix}=\begin{pmatrix}\mathbf{a}_1\\ \vdots\\ \mathbf{a}_{i-1}\\ c\mathbf{a}_i\\ \mathbf{a}_{i+1}\\ \vdots\\ \mathbf{a}_n\end{pmatrix}$$

となる。

基本行列$R_n(i,\,j;c)$を行列Aの左から掛けたものは行列Aのｊ行のｃ倍をｉ行に加えたものになる。すなわち、

$$R_n(i,j;c)A = R_n(i,j;c)\begin{pmatrix}\mathbf{a}_1\\ \vdots\\ \mathbf{a}_i\\ \vdots\\ \mathbf{a}_j\\ \vdots\\ \mathbf{a}_n\end{pmatrix} = \begin{pmatrix}\mathbf{a}_1\\ \vdots\\ \mathbf{a}_i + c\mathbf{a}_j\\ \vdots\\ \mathbf{a}_j\\ \vdots\\ \mathbf{a}_n\end{pmatrix}$$

となる。

基本行列のこれらの性質はガウスの消去法(3)、(4)における行の操作に対応しているので、行列Aの左から適当な基本行列を掛けていくことによりガウスの消去法を行うことができる。すなわち、求める逆行列A^{-1}をXとおき、

$$AX = I$$

の左からガウスの消去法の操作に対応して適当な基本行列E_1、・・・、E_pを順番に掛けていくと

$$E_p \cdots E_1 A = \begin{pmatrix}1 & a'_{12} & a'_{13} & \cdots & a'_{1n}\\ 0 & 1 & a'_{23} & \cdots & a'_{2n}\\ & & \ddots & & \\ 0 & & \cdots & 1 & a'_{n-1,n}\\ 0 & & \cdots & 0 & 1\end{pmatrix}$$

$$E_p \cdots E_1 AX = E_p \cdots E_1 I = B \qquad (2)$$

と変形できる。ここで、$B = E_p \cdots E_1$とおいている。

式（2）は

$$\begin{pmatrix}1 & a'_{12} & a'_{13} & \cdots & a'_{1n}\\ 0 & 1 & a'_{23} & \cdots & a'_{2n}\\ & & \ddots & & \\ 0 & & \cdots & 1 & a'_{n-1,n}\\ 0 & & \cdots & 0 & 1\end{pmatrix} X = B$$

という形なので、上式はXについての方程式とみて、第n行から順番にx_{ij}の値を求めるこ

とができる[3]、[4]。
すなわち、

$$X=\begin{bmatrix}\mathbf{x}_1\\ \vdots\\ \mathbf{x}_n\end{bmatrix}、\quad B=\begin{bmatrix}\mathbf{b}_1\\ \vdots\\ \mathbf{b}_n\end{bmatrix}$$

とおくと、式（２）は次式の形になる。

$$\begin{bmatrix}1 & \mathbf{a}'_{12} & \mathbf{a}'_{13} & \cdots & \mathbf{a}'_{1n}\\ 0 & 1 & \mathbf{a}'_{23} & \cdots & \mathbf{a}'_{2n}\\ \vdots & \ddots & \ddots & \ddots & \vdots\\ \vdots & & \ddots & 1 & \mathbf{a}'_{n-1,n}\\ 0 & \cdots & \cdots & 0 & 1\end{bmatrix}\begin{bmatrix}\mathbf{x}_1\\ \\ \vdots\\ \\ \mathbf{x}_n\end{bmatrix}=\begin{bmatrix}\mathbf{b}_1\\ \\ \vdots\\ \\ \mathbf{b}_n\end{bmatrix}$$

上式を行ごとに表すと式（2.1）～（2.ｎ）の連立方程式になる

$$\mathbf{x}_1+a'_{12}\mathbf{x}_2+\quad\cdots\quad+a'_{1n}\mathbf{x}_n=\mathbf{b}_1 \qquad (2.1)$$

$$\mathbf{x}_2+a'_{23}\mathbf{x}_3+\cdots+a'_{2n}\mathbf{x}_n=\mathbf{b}_2 \qquad (2.2)$$

$$\vdots$$

$$\mathbf{x}_{n-1}+a'_{n-1,n}\mathbf{x}_n=\mathbf{b}_{n-1} \qquad (2.n\text{-}1)$$

$$\mathbf{x}_n=\mathbf{b}_n \qquad (2.n)$$

上の連立方程式の解$\mathbf{x}_i$は、まず式（2.n）により$\mathbf{x}_n$が与えられ、$\mathbf{x}_n$と式（2.n-1）より$\mathbf{x}_{n-1}$が与えられ、・・・、$\mathbf{x}_n$～$\mathbf{x}_{i+1}$と式（2.i）より$\mathbf{x}_i$が与えられ、・・・、$\mathbf{x}_n$～$\mathbf{x}_2$と式（2.1）より$\mathbf{x}_1$が与えられるという形で求めることができる。

上の方法によって逆行列を求める手続きを Mat_Inv_Gauss としてユニットファイル UMatCalc.pas に用意している。ユニットファイル UMatCalc.pas では、この手続き Mat_Inv_Gauss を用いて求めた逆行列を値とする関数 MatInv を用意した。MatInv のヘッダーは次のようになっている。

```
function MatInv( a : TMatCalc;
                 n : Longint;
                 ZeroV          //  ゼロの基準値
                   : Extended ) : TMatCalc;
```

第１パラメータに逆行列を求める行列を表わす配列を、第２パラメータに行列の型(n,n)を与える値ｎを設定する。第３パラメータには０とみなす基準値を設定する。この値は、MatInv 内で逆行列を計算するために呼び出す手続き Mat_Inv_Gauss において数値を０とみなす基準値の実パラメータとして用いられている。これは、計算結果が理論的には０であっても実数型の有効桁数の制限のために０にはならない場合に対処するためのものである。

行列Ａが対称行列のときは Cholesky 分解を用いて逆行列を求めることができる。ここで、行列Ａが対称行列であるとは、Ａの転置行列 A' が自分自身に等しい、すなわち、

$$A = A'$$

が成り立つことをいう。これを A の要素の関係で表すと

$$a_{ij} = a_{ji}$$

となる。

行列Ａが対称行列のときは、次の Cholesky 分解

$$A = SS' 、 \qquad (3)$$

ただし、

$$S = \begin{pmatrix} s_{11} & 0 & 0 & \cdots & 0 \\ s_{21} & s_{22} & 0 & \cdots & 0 \\ & & \ddots & & \\ s_{n-1,1} & & \cdots & s_{n-1,n-1} & 0 \\ s_{n1} & s_{n2} & \cdots & s_{n,n-1} & s_{nn} \end{pmatrix} 、$$

を行って、逆行列を

$$A^{-1} = (S^{-1})' S^{-1}$$

によって求めることができる。

Cholesky 分解（３）式は、

$$a_{ii} = \sum_{j=1}^{i} s_{ij} s_{ij}$$

および

$$a_{ik} = \sum_{j=1}^{i} s_{ij} s_{kj} \qquad k > i$$

の関係式から求めることができる[4]。

まず、$i=1$の場合から計算を始める。このとき、

$$a_{11}=s_{11}s_{11}、\quad a_{1k}=s_{11}s_{k1}$$

であるので、

$$s_{11}=\sqrt{a_{11}}、\quad s_{k1}=\frac{a_{1k}}{s_{11}}$$

となる。

i-1 列までの$s_{k,j}$が求まったとすると、次式

$$s_{ii}=\sqrt{a_{ii}-\sum_{j=1}^{i-1}s_{ij}s_{ij}}、\quad s_{ki}=\frac{a_{ik}-\sum_{j=1}^{i-1}s_{ij}s_{kj}}{s_{ii}}$$

より i 列のs_{ki}が求まる。

上の Cholesky 分解を用いた方法(4)で逆行列を求める手続き Mat_Inv_S をユニットファイル UMatCalc.pas に用意した。Cholesky 分解を用いて求めた逆行列を値とする関数 MatInvS も UMatCalc.pas に用意した。ヘッダーは

```
function MatInvS( a : TMatCalc;
                  n : Longint;
                  ZeroV : Extended ) : TMatCalc;
```

となっていて、MatInv の場合と同じである。

関数 MatInv と MatInvS の使用例もプログラム PCheckMat.dpr に含めた。

「Inv」ボタン（図１）のクリックで「A=」と表示のある左上のグリッドに設定された行列の逆行列が関数 MatInv の呼び出しによってガウスの消去法で求められ、下のグリッドに「Inv(A)=」のタイトルとともに表示される。右上のグリッドには左上に設定された行列とその逆行列との積が表示される。この行列は理論的には単位行列Iであるべきものであるが、計算精度の限界のため０となるべき(i,j)成分に０でない非常に小さい値が表示されることがある。計算では Extended 型を用いているが、Extended 型の精度は約１９～２０桁である。

「Inv(S)」ボタン（図１）をクリックすると「A =」と表示のある左上のグリッドに設定されている行列の値Ａから対称行列AA'が求められて「A =」の表示のあるグリッドに表示され、その逆行列が MatInvS によって算出される。算出した逆行列は下のグリッドに「Inv(A) =」のラベルとともに表示されます。右上のグリッドには行列とその逆行列との積$(AA')(AA')^{-1}$が表示される。この積の値も理論的には単位行列Iであるべきものであ

るが、計算精度の限界のため０となるべき(i,j)成分が０でない非常に小さい値になることがある。

単回帰分析

逆行列の使用例として単回帰分析のプログラム PAnal2Vars.dpr を作成した。

単回帰分析とは、２変量のデータ (x_1, y_1)、・・・、(x_N, y_N) が与えられたとき、各 y_i を x_i で最もよく説明する１次式を求めるものである。いま、１次式を

$$y = ax + b$$

で与えたとき、y_i を上の１次式で説明するときの誤差の２乗和を SS で表わすと、

$$SS = \sum_{i=1}^{N} \{y_i - (ax_i + b)\}^2$$

となる。

y_i を x_i で最もよく説明する１次式を SS を最小にするものとして与えるときのａとｂの値は、SS のａおよびｂによる偏導関数を０にするものとして求めることができる。

偏導関数は次のように与えられる。

$$\frac{\partial SS}{\partial a} = \sum_{i=1}^{N} 2\{y_i - (ax_i + b)\}(-x_i)$$

$$= -2\sum_{i=1}^{N} (y_i x_i - ax_i x_i - bx_i)$$

$$\frac{\partial SS}{\partial b} = \sum_{i=1}^{N} 2\{y_i - (ax_i + b)\}(-1)$$

$$= -2\sum_{i=1}^{N} (y_i - ax_i - b)$$

上の２つの偏導関数を０とおいて次式を得る。

$$\sum_{i=1}^{N} y_i x_i = a\sum_{i=1}^{N} x_i x_i + b\sum_{i=1}^{N} x_i$$

$$\sum_{i=1}^{N} y_i = a\sum_{i=1}^{N} x_i + b\sum_{i=1}^{N} 1$$

ずなわち、

$$\begin{pmatrix} \sum_{i=1}^{N} y_i x_i \\ \sum_{i=1}^{N} y_i \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{N} x_i x_i & \sum_{i=1}^{N} x_i \\ \sum_{i=1}^{N} x_i & \sum_{i=1}^{N} 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} \qquad (4)$$

したがって、

$$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{N} x_i x_i & \sum_{i=1}^{N} x_i \\ \sum_{i=1}^{N} x_i & \sum_{i=1}^{N} 1 \end{pmatrix}^{-1} \begin{pmatrix} \sum_{i=1}^{N} y_i x_i \\ \sum_{i=1}^{N} y_i \end{pmatrix} \qquad (5)$$

としてａとｂが求まる。

いま、

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}、\quad X = \begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \\ x_N & 1 \end{pmatrix}、\quad b = \begin{pmatrix} a \\ b \end{pmatrix}$$

とおくと、（４）式は

$$X'\mathbf{y} = X'Xb$$

（５）式は

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

と書る。

また、

$$y_i \approx ax_i + b$$

はまとめて

$$\mathbf{y} \approx Xb$$

と書ける。ここで、記号 ≈ は近似式であることを表わす。

単回帰分析を行うプログラム PAnal2Vars.dpr を実行すると図４のようなフォームが表示される。グリッドのセルに x_i と y_i の値を設定して、「散布図」ボタンをクリックすると単回帰分析が行われる。「計算」ボタンをクリックすると x_i および y_i それぞれの平均値や標準偏差などが計算される。

図４　プログラム PAnal2Vars.dpr の実行開始時のフォーム

グリッド内のセルのデータの設定は、セルをクリックしてアクティブにしてからキーボードで入力する。データ設定用のセルをクリックして「追加」ボタンをクリックするとそのセルの下に新しい行が挿入される。「削除」ボタンをクリックすると、そのセルの行が削除される。グリッド内のデータの設定は空のセルがないように全てのデータ設定用セルに値を設定する（図５）。グリッドの上のラベルの設定用エディットコンポーネントには２つの変量 x_i および y_i のラベルを設定する。設定された文字列は散布図（図７）の出力で用いられる。

図５ グリッドに設定されたデータ

設定したデータは、「保存」ボタンのクリックでファイルに保存することができる。ファイル名は、「保存」ボタンのクリックで表示されるダイアログボックスにおいて設定する。このファイルはテキストファイルとして書き出されるので、テキストエディタで開いて見ることもできる。

「保存」ボタンのクリックで保存したファイルは、「読出」ボタンのクリックで読み込むことができる。読み込むファイルの名前は、「読出」ボタンのクリックで表示されるダイアログボックスで設定する。

図６ 「散布図」ボタン（図５）のクリックで表示されるフォーム

図５の状態で「散布図」ボタンをクリックすると、図６のようなフォームが表示される。「描画」ボタンをクリックすると、出力用ファイル名の設定を求めるダイアログボックスが表示される。ここで設定された名前のファイルに回帰分析の計算結果が出力される。

ダイアログボックスにファイル名を設定すると散布図の描かれたフォームが表示され、

求められた１次式 $y = ax + b$（回帰直線という）のグラフが散布図内に描かれる（図７）。フォーム上部の「印刷」ボタンをクリックすると散布図がプリンタに出力される。散布図は印刷用紙が縦長の方向であることを想定して出力されるので、「印刷」ボタンのクリックで表示されるダイアログボックスにおいて印刷用紙の向きが縦長であることを確認する。表示されたダイアログボックスにおいて用紙の向きの設定が横長であれば縦長に設定し直してから「OK」ボタンをクリックする。

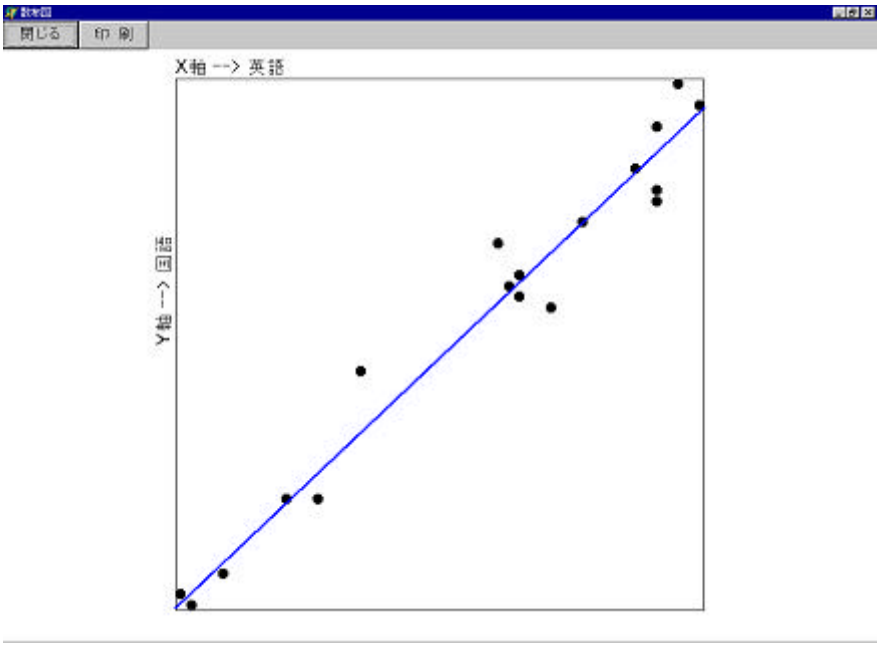

図７　散布図と回帰直線

「閉じる」（図７）、「終了」（図５）ボタンのクリックでプログラムを終了後、「散布図」、「描画」ボタンのクリックで表示されたダイアログボックスにおいて設定した名前の出力用ファイルをテキストエディタで開くと、回帰直線の係数ａとｂの値、および x_i と y_i の相関係数や決定係数などが書き出されているのを見ることができる。決定係数は相関係数を２乗したもので、回帰直線がデータ y_i の変動をどれぐらい説明しているかを示すものである。詳しくは文献 [4] などを参照されたい。

参 考 文 献

（1）斎藤正彦「線型代数入門」、Pp.278、東京大学出版会、1966.
（2）J.D.Carroll & P.E.Green. Mathematical tools for applied multivariate analysis (revised edition). Pp.376, Academic Press, 1997
（3）戸川隼人「マトリクスの数値計算」、Pp.316、オーム社、1971.
（4）岡本安晴「Delphi で学ぶデータ分析法」、Pp.275、ＣＱ出版社、1998.

リスト１ UMatCalc.pas **に宣言されている関数**

```
function MatAdd( a, b : TMatCalc;
                 m, n : Longint ) : TMatCalc;
```

(m,n)型の行列ａとｂの和を関数値とする

```
function MatSub( a, b : TMatCalc;
                 m, n : Longint ) : TMatCalc;
```

(m,n)型の行列ａとｂの差を関数値とする

```
function MatMul( a, b : TMatCalc;
                 L, m, n : Longint ) : TMatCalc;
```

(L,m)型の行列ａと(m,n)型の行列ｂの積を関数値とする

```
function MatScl( s : Extended;
                 a : TMatCalc;
                 m, n : Longint ) : TMatCalc;
```

行列ａのスカラーｓ倍を関数値とする

```
function Transpose( a : TMatCalc;
                    m, n : Longint ) : TMatCalc;
```

行列ａの転置行列を関数値とする

```
procedure Mat_Inv_Gauss( a     : TMatCalc;  //  a の逆行列を求める
                         var b : TMatCalc;  //  a の逆行列が返される
                         n     : Longint;   //  行列のサイズ
                         ZeroV : Extended;  //  ゼロの基準値
                         var ECode          //  逆行列が求まれば０を返す
                               : Longint );
```

ガウスの消去法によって行列ａの逆行列を求め、求めた逆行列をｂに返す手続き

```
function MatInv( a : TMatCalc;
                 n : Longint;
                 ZeroV          //  ゼロの基準値
                   : Extended ) : TMatCalc;
```

行列ａの逆行列をガウスの消去法で求め、求めた逆行列を関数値とする

```
procedure Mat_Inv_S( a     : TMatCalc;      //  a の逆行列を求める
                     var b : TMatCalc;      //  a の逆行列が返される
                     n     : Longint;       //  行列のサイズ
                     ZeroV : Extended;      //  ゼロの基準値
                     var ECode              //  逆行列が求まれば０を返す
                           : Longint );
```

行列ａの逆行列を Cholesky 分解を用いて求め、求めた逆行列をｂに返す手続き

```
function MatInvS( a : TMatCalc;
                  n : Longint;
                  ZeroV            //  ゼロの基準値
                    : Extended ) : TMatCalc;
```

行列ａの逆行列を Cholesky 分解を用いて求め、求めた逆行列を関数値とする

```
function  MatDet( a    : TMatCalc;
                  n    : Longint;
                  zero             //  ゼロの基準値
                       : Extended ) : Extended;
```

行列ａの行列式を関数値とする

```
function MatRnk( a : TMatCalc;
                 m, n  : Longint;
                 ZeroV            //  ゼロの基準値
                       : Extended ) : Longint;
```

(m,n)型の行列ａの階数を関数値とする