行列計算における数式処理ソフト maxima の利用について

足立健朗

Contents

1. イントロダクション

1.1. maxima とは何か. **maxima** は汎用の数式処理ソフトである。そして、数式処理ソフトとは、数式のまま計算をすることのできる計算機プログラムのことである。電卓的に数式を入力計算できるだけでなく、集合、ベクトル、配列、数列、級数、行列、関数などの様々なデータ型を扱うことができ、それらをプログラムして処理することのできる超高水準プログラミング言語としての機能をもったソフトである。

maxima は MIT(マサチューセッツ工科大学) の Project MAC で開発された **Macsyma** の子孫である。**Macsyma** とは MAC's SYmbolic MAnipulator の略であり、記号積分をコンピュータで実行させる研究の成果として、同じく MIT で開発された人工知能用プログラミング言語 Lisp の応用ソフトの一つとして生まれた数式処理ソフトである。Project MAC の解散とともに、Macsyma はいくつかの枝に分岐する。その一つは Symbolics Inc. に売却された Macsyma であり、もう一つは米国エネルギー省の管理の下に置かれた DOE Macsyma である。Symbolics の Macsyma は Symbolics Inc. とその後継者の Macsyma Inc が 1999 年になくなるまで、商用の数式処理ソフトして販売されていた。Texas 大学の William Schelter は個人的に Doe Macsyma のソースコードを所持し、それを GNU Common Lisp に移植しメンテナンスしていたが、1990 年代半ばに、それを **maxima** と名付け公開する決意をした。そして MIT や国防省とのライセンスについての合意の下にそれを公開した。Bill Schelter は 2001 年に、心臓発作のためなくなったが、現在は UCB の R.Fateman らを中心とする、インターネット上のボランティアたちの手によってオープンソースの GPL ソフトウェアとして維持、開発されている。

maxima と同様の機能をもった数式処理ソフトは他にもいくつも存在する。主なものをあげると、**Asir**、**Maple**、**Mathematica**、**MuPAD**、**Reduce**、**Yacas** などがある。このうち、**Maple**、**Mathematica** は有名な商用ソフトであり、**Reduce** は **Macsyma** と同じ時期に開発された商用のソフトである。**Asir** は日本で開発された数式処理ソフトであるが、得意な分野が限定されているので、他のものほどには汎用性は無い。**Yacas** は最近開発された GPL ライセンスで配布されるフリーな数式処理ソフトであるが、**maxima** ほどの機能はもっていないようである。さて、この文書で特に **maxima** を採用しているのは、フリーでオープンソースな汎用数式処理ソフトで、ライセンスを気にせず、高額な投資をすることなく教育目的に利用できるという理由からである。したがって、**Asir**、**MuPAD**、**Yacas** なども同じ目的に使用することができる。

1.2. maxima の入手とインストール. **maxima** は Linux、 FreeBSD、MacOS X など、Unix と同等の機能をもつ OS に Common Lisp 処理形がインストールされた環境が存在するならば、ソースコードをダウンロードして、コンパイル、インストールすることができる。もし、あなたが Windows ファミリーの OS しか使えないならば、**maxima** の実行形式版をダウンロードしてインストールすることによって、**maxima** を使うことができる。あるいは、Windows に cygwin という、疑似 Unix 環境を提供するソフトをインストールしておいて Unix 版の **maxima** をインストールすることも可能である。Windows 版 **maxima** 及びソースコード版 **maxima** は、どちらも

`http://maxima.sourceforge.net/`

からダウンロードできる。好みのウェッブブラウザーを使って上記 URL にアクセスしてみるとよい。Windows 版の場合、ダウンロードしたファイルをダブルクリックすればインストーラが起動するので、適宜質問に答えながらボタンをクリックしていけばインストールできるはずである。インストールが終了すれば、スタートメニューに maxima が

登録されているはずである、また、デスクトップ上に **maxima** のアイコンが作られているはずである。

1.3. maxima の実行と処理、そして終了. Unix 版で Terminal のコマンドラインから **maxima** を立ち上げると、次のように表示される。

```
  i i i i i i i       ooooo    o        ooooooo   ooooo   ooooo
  I I I I I I I      8     8   8           8     8     o  8    8
  I  \ `+' /  I      8         8           8     8        8    8
   \  `-+-'  /       8         8           8      ooooo   8oooo
    `-__|__-'        8         8           8           8  8
        |            8     o   8           8     o     8  8
  ------+------       ooooo    8oooooo  ooo8ooo   ooooo   8

Copyright (c) Bruno Haible, Michael Stoll 1992, 1993
Copyright (c) Bruno Haible, Marcus Daniels 1994-1997
Copyright (c) Bruno Haible, Pierpaolo Bernardi, Sam Steingold 1998
Copyright (c) Bruno Haible, Sam Steingold 1999-2002

Maxima 5.9.0 http://maxima.sourceforge.net
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
This is a development version of Maxima. The function bug_report()
provides bug reporting information.
(C1)
```

また、Unix 版の xmaxima あるいは Windows 版の maxima を起動すると、専用のウィンドウが開いて、単に

```
(C1)
```

と表示される。いずれの場合も (C1) というのがプロンプト (入力促進記号) であり、これにつづけてあなたのコマンドをタイプ入力していくことにより、対話的に **maxima** を利用できるようになる。Unix 版の **maxima** を、TeXmacs や、Emacs+imaxima などの環境で使用すると美しい数式表示が可能になる。以下では、TeXmacs 上で **maxima** を使用した場合の例をあげることにする。青字の部分がユーザーの入力部分である。

(C1) `integrate(1/(1+x^3),x);`

(D1) $-\frac{\log\left(x^2-x+1\right)}{6}+\frac{\arctan\ \frac{2x-1}{\sqrt{3}}}{\sqrt{3}}+\frac{\log(x+1)}{3}$

(C2) `a : (1 + x - 3*y)^5;`

(D2) $(-3y+x+1)^5$

(C3) `expand(D2);`

(D3) $-243y^5+405xy^4+405y^4-270x^2y^3-540xy^3-270y^3+90x^3y^2+270x^2y^2+270xy^2+90y^2-15x^4y-60x^3y-90x^2y-60xy-15y+x^5+5x^4+10x^3+10x^2+5x+1$

(C4) `10!;`

(D4) 3628800

(C5) `factor(D4);`

(D5) $2^8 3^4 5^2 7$

ここで、(C*) につづく文字列が入力したコマンドであり、(D*) に続く文字列が **maxima** の出力をあらわしている。さらに、上の例では、**maxima** が履歴を憶えているシステムであることを利用している。例えば、(C3) における expand(d2); の d2 は (D2) の結果を意味でする。すべての作業が終えるときには、

```
(C*) quit();
```

とする。

maxima を使っていると、しばしばトラブルに出会うだろう。何が起こるかは、状況による。一番目にする可能性が高いのは、あなたがタイプミスをしたときに、表示されるエラーメッセージだ。

```
(C2) B:matrix([1,2:});
Incorrect syntax: Missing ]
B:matrix([1,2:})
              ^
(C2) Incorrect syntax: Premature termination of input at ;.
;
^
```

こういうときには、慌てずもういちど正しくタイプしなおせばいい。。

```
(C2) B:matrix([1,2]);
(D2)                               [ 1  2 ]
```

タイプミスでなく、論理的なミスをすることもある。異なったデータ型の変数を足そうとして、

```
(C3) A+B;
FULLMAP found arguments with incompatible structure.
 -- an error.  Quitting.  To debug this try DEBUGMODE(TRUE);)
```

というようなメッセージがあらわれたら、それは入力した式に論理的な誤りがある可能性を示している。入力した式の意味をよく考え直してみよう。もっと深刻なエラーが起きた場合には、次のようなメッセージがあらわれるかも知れない。

```
*** - POSITION: :START = 1 should not be greater than :END = 0
The following restarts are available:
R1 = Macsyma top-level

1. Break [1]>
```

これは、**maxima** では解決できないようなエラーが生じた場合に表示される。

```
1. Break [1]> R1
```

あるいは

```
1. Break [1]> (run)
```

とすることによって **maxima** がリスタートする。そのまま、**maxima** を終るには、

```
1. Break [1]> (bye)
```

としよう。

1.4. **maximaについての詳しい情報.** **maxima**を使っていて、しばしば特定のコマンドについてより詳しい情報が欲しくなることがあるだろう。たとえば、コマンド (**maxima**の正式な用語では関数と言うが、数学の関数とまぎらわしいので、本文書ではコマンドと言うことにする) **factor**について知りたいとおもったら、プロンプトに続いて**describe**(“factor”)とタイプする。すると

```
(C1) describe("factor");

 0: DONTFACTOR :(maxima.info)Definitions for Matrices and Linear Algebra.
 1: EXPANDWRT_FACTORED :Definitions for Simplification.
 2: FACTOR :Definitions for Polynomials.
 3: FACTORFACSUM :Definitions for Simplification.
 4: FACTORFLAG :Definitions for Polynomials.
 5: FACTORIAL :Definitions for Number Theory.
 6: FACTOROUT :Definitions for Polynomials.
 7: FACTORSUM :Definitions for Polynomials.
 8: GCFACTOR :Definitions for Polynomials.
 9: GFACTOR :Definitions for Polynomials.
 10: GFACTORSUM :Definitions for Polynomials.
 11: MINFACTORIAL :Definitions for Number Theory.
 12: NEXTLAYERFACTOR :Definitions for Simplification.
 13: NUMFACTOR :Definitions for Special Functions.
 14: SAVEFACTORS :Definitions for Polynomials.
 15: SCALEFACTORS :Definitions for Miscellaneous Options.
 16: SOLVEFACTORS :Definitions for Equations.
Enter n, all, none, or multiple choices eg 1 3 :
```

のように表示される。(残念ながら英語である。) さらに、自分に必要な情報の番号を選んで、

```
Enter n, all, none, or multiple choices eg 1 3 : 5
```

と入力すれば、

```
Info from file /usr/local/info/maxima.info:
 - Function: FACTORIAL (X)
     The factorial function.  FACTORIAL(X) = X! .  See also
     MINFACTORIAL and FACTCOMB.  The factorial operator is !, and the
     double factorial operator is !!.

(D1)                                FALSE
```

のように、**factor**の説明と使い方、場合によっては例などが表示される。

より詳しい情報が知りたいときは、**maxima**の配布物に含まれる文書、tutorial、manual、info ファイルなどを参照することになる。xmaxima や Windows版 maxima などを使っていれば, これらをオンラインで参照できる。PDF でフォーマットされた文書が欲しい場合には、以下の URL からダウンロードすればいい。

`http://maxima.sourceforge.net/docs.shtml`

ここで、Maxima Reference Manual、DOE-Maxima Refernce、The Maxima Book などが手に入る。また、

`http://www.bekkoame.ne.jp/~ponpoko/index.html`
には Maxima Reference Manual (maxima-5.6) の日本語訳が置いてある。

2. 行列を与える

ここでは、**maxima** において行列を生成する方法について述べる。第 1 の方法は、行列の全ての成分をあからさまに与える方法である。一般的に、行列

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

を **maxima** に与えるには、コマンドラインから A : **matrix**($[a_{11}, \cdots, a_{1n}]$,$[a_{21}, \cdots, a_{2n}]$,$\cdots$, $[a_{m1}, \cdots, a_{mn}]$); とタイプする。

例 1. (C1) `A : matrix([a, b], [c, d]);`

(D1) $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

例 2. (C2) `B : matrix([-1, 2, 3], [0, 4, -1]);`

(D2) $\begin{pmatrix} -1 & 2 & 3 \\ 0 & 4 & -1 \end{pmatrix}$

maxima において、コロン '**":"**' は、代入あるいは定義を意味する記号である、コロンの右辺の式にコロンの左辺にある名前をつける、あるいは右辺を左辺の変数の値とするときに使う。

第 2 の方法は行列要素を 2 次元配列と見て、配列から行列を生成する方法である。すなわち、$a_{ij} = a[i, j]$ を与えられた 2 次元配列とするとき、**maxima** のコマンドラインから A : **genmatrix**(a, i2, j2, i1, j1); とタイプすれば行列

$$A = \begin{pmatrix} a_{i1,j1} & \cdots & a_{i1,j2} \\ \vdots & \ddots & \vdots \\ a_{i2,j1} & \cdots & a_{i2,j2} \end{pmatrix}$$

が生成される。特に、i1=j1= 1 の場合には、簡略化した構文 A:genmatrix(a, i2, j2); が使える。

例 3. (C1) `a[i,j] := i + j - 1;`

(D1) $a_{i,j} := i + j - 1$

(C2) `A : genmatrix(a, 3, 4);`

(D2) $\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \end{pmatrix}$

注意 1. 配列定義には構文 a[i,j] := "i,j の式" を使う。

注意 2. 配列については 7 節を参照せよ。

第 3 の方法は、**maxima** に用意されている対話的に行列成分を入力するための関数 **entermatrix** を用いる方法である。実際例で説明しよう。

例 4. (C1) `A : entermatrix(2,3);`

```
Row 1 Column 1:   1;
Row 1 Column 2:   0;
```

```
Row 1 Column 3:   -1;
Row 2 Column 1:   4;
Row 2 Column 2:   2;
Row 2 Column 3:   8;
Matrix entered.
```

(D1) $\begin{pmatrix} 1 & 0 & -1 \\ 4 & 2 & 8 \end{pmatrix}$

特に正方行列のときには、入力を簡易なものにするための、オプションを選択できる。

例 5. `(C2)  B : entermatrix(3,3);`

```
Is the matrix 1.  Diagonal 2.  Symmetric 3.  Antisymmetric 4.  General
Answer 1, 2, 3 or 4 :   1;
Row 1 Column 1:   -1;
Row 2 Column 2:   2;
Row 3 Column 3:   4;
Matrix entered.
```

(D2) $\begin{pmatrix} -1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 4 \end{pmatrix}$

ここでは、B を対角行列として、対角成分だけを入力している。

すでに存在している行列からそのコピーをつくるには、単に代入文を使えばいい。例えば、

`(C1)  A : matrix([1, 2], [3, 4]);`

(D1) $\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$

`(C2)  B : A;`

(D2) $\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$

`(C3)  B;`

(D3) $\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$

特別な行列を定義するためには、専用のコマンドが用意されている。対角成分がすべて等しい対角行列を定義するには **diagmatrix** を使う。

例 6. `(C1)  A : diagmatrix(4, 2);`

(D1) $\begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}$

特に n 次単位行列には **ident** を用いて、ident(n); とタイプする。

例 7. `(C1)  ident(3);`

(D1) $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

(m, n)-型の零行列を作りたいなら、**zeromatrix** というコマンドを用いて、A : zeromatrix(m, n); のようにタイプする。

例 8. (C1) `C : zeromatrix(3, 2);`

(D1) $\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$

第 (i, j)-成分が与えられた値 a で、他の成分がすべてゼロであるような (m, n) 型行列を与えるコマンドは **ematrix**(m, n, a, i, j) である。

例 9. (C1) `D : ematrix(4, 4, -1, 2, 3);`

(D1) $\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$

$(m, 1)$ 型の行列を m 次元列ベクトルとも言うが、m 次元列ベクトルを与えるコマンドは **columnvector** である。columnvector を使うためには、事前に **load**(eigen); とタイプして必要なパッケージを読み込ませておく必要がある。

例 10. (C1) `load(eigen);`

```
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
(D1) /usr/local/share/maxima/5.9.0/share/matrix/eigen.mac
(C2) a : columnvector([1, 2, 3]);
```

(D2) $\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$

(C3) `x : [1, 2, 3, 4];`

(D3) $[1, 2, 3, 4]$

(C4) `b : columnvector(x);`

(D4) $\begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}$

3. 行列の和と積

3.1. 和、スカラー倍. 同じ型の 2 つの行列 A, B に対してその和 $A+B$ が定義できる。結果の行列 $A+B$ は A, B と同じ型の行列である。すなわち

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & a_{ij} & \cdot \\ \cdot & \cdot & & \cdot \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}, \qquad B = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & b_{ij} & \cdot \\ \cdot & \cdot & & \cdot \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{pmatrix}$$

のとき、

$$A+B = \begin{pmatrix} a_{11}+b_{11} & a_{12}+b_{12} & \cdots & a_{1n}+b_{1n} \\ a_{21}+b_{21} & a_{22}+b_{22} & \cdots & a_{2n}+b_{2n} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & a_{ij}+b_{ij} & \cdot \\ \cdot & \cdot & & \cdot \\ a_{m1}+b_{m1} & a_{m2}+b_{m2} & \cdots & a_{mn}+b_{mn} \end{pmatrix}$$

である。

maxima で行列の計算をするのは非常に簡単である。A と B をすでに定義された行列とするとき、行列 C = A + B を求めるには、プロンプトから C : A + B; とタイプすればいい。

例 11. (C1) `A : matrix([a, b], [c, d]);`

(D1) $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

(C2) `B : matrix([1, 2], [3, 4]);`

(D2) $\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$

(C3) `C : A + B;`

(D3) $\begin{pmatrix} a+1 & b+2 \\ c+3 & d+4 \end{pmatrix}$

行列のスカラー倍も簡単である、A を任意の行列、c をスカラーとするとき cA を求めるには、c*A; とタイプするだけでいい。上の例から引続き、

例 12. (C4) `3*A;`

(D4) $\begin{pmatrix} 3a & 3b \\ 3c & 3d \end{pmatrix}$

(C5) `f*A;`

(D5) $\begin{pmatrix} af & bf \\ cf & df \end{pmatrix}$

(C6) `n*B;`

(D6) $\begin{pmatrix} n & 2n \\ 3n & 4n \end{pmatrix}$

3.2. **行列の積.** 2つの行列の積はちょっと難しい。勝手な2つの行列に対しては積が定義できるとは限らない。積の定義はつぎの通りである。$A = (a_i)$ を (m, l) 型の行列、$B = (b_{ij})$ を (l, n) 型の行列とするとき、A と B の積 $AB = (c_{ij})$ が定義できて、AB は (m, n) 型の行列になる。各成分 c_{ij} は公式

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

$$= a_{i1} b_{1j} + a_{i2} b_{2j} + \cdots + a_{il} b_{lj} \tag{2}$$

で与えられる。注意すべきことは、AB が定義されても BA が定義されるとは限らないし、また定義される場合でも $AB = BA$ がなりたつとは限らないことである。

特別な場合として A,B がともに n 次正方行列である場合には、AB,BA はともに定義可能で、ともに n 次正方行列になるが、この場合でも、$AB = BA$ とは限らない。このような事情を行列の積は非可換であるという。

行列の積を **maxima** に計算させるには “.” 作用素を使う。A と B が与えられていて AB が計算可能であるとき、**maxima** では、A.B とタイプする。AB が計算可能でないときはエラーとなる。

例をあげよう。

例 13. (C1) `A : matrix([2, 3, 4], [-1, 0, 2]);`

(D1) $\begin{pmatrix} 2 & 3 & 4 \\ -1 & 0 & 2 \end{pmatrix}$

(C2) `B : matrix([3, 0], [-1, 2], [3, 4]);`

(D2) $\begin{pmatrix} 3 & 0 \\ -1 & 2 \\ 3 & 4 \end{pmatrix}$

(C3) `C : A.B;`

(D3) $\begin{pmatrix} 15 & 22 \\ 3 & 8 \end{pmatrix}$

(C4) `D : B.A;`

(D4) $\begin{pmatrix} 6 & 9 & 12 \\ -4 & -3 & 0 \\ 2 & 9 & 20 \end{pmatrix}$

(C5) `C.D;`

```
incompatible dimensions - cannot multiply
-- an error.  Quitting.  To debug this try DEBUGMODE(TRUE);)
```

(C6) `E : matrix([a, b], [c, d]);`

(D6) $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

(C7) `F : matrix([e, f], [g, h]);`

(D7) $\begin{pmatrix} e & f \\ g & h \end{pmatrix}$

(C8) `E.F;`

(D8) $\begin{pmatrix} bg + ae & bh + af \\ dg + ce & dh + cf \end{pmatrix}$

(C9) `F.E;`

(D9) $\begin{pmatrix} cf+ae & df+be \\ ch+ag & dh+bg \end{pmatrix}$

3.3. **行列の巾乗.** 行列の積の定義から、もし行列 A が n 次正方行列であれば、A の k 乗を、帰納的に $A^k := AA^{k-1}$ で定義できる。**maxima** はこのような行列の巾乗を計算する手段も備えている。

与えられた正方行列 A に対して、A^k を求めるには A^^k とタイプすればいい。ただし、k は具体的な自然数でなくてはならない。また、A^ k とすると各成分を k 乗した行列が得られるがこれは A^k ではないことに注意する。

例 14. (C1) `A : matrix([2, 3], [0, -1]);`

(D1) $\begin{pmatrix} 2 & 3 \\ 0 & -1 \end{pmatrix}$

(C2) `A^^5;`

(D2) $\begin{pmatrix} 32 & 33 \\ 0 & -1 \end{pmatrix}$

(C3) `B : matrix([a, b, c], [d, e, f], [g, h, i]);`

(D3) $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$

(C4) `B^3;`

(D4) $\begin{pmatrix} a^3 & b^3 & c^3 \\ d^3 & e^3 & f^3 \\ g^3 & h^3 & i^3 \end{pmatrix}$

B^3 は紙に収まりきらないので、端末版の表示をあげよう。

```
(C6) B^^3;
(D6)

          [                                                  2  ]
          [ g (c i + b f + a c) + d (c h + b e + a b) + a (c g + b d + a ) ]
          [                                                     ]
 Col 1 =  [                              2                      ]
          [ g (f i + e f + c d) + d (f h + e  + b d) + a (f g + d e + a d) ]
          [                                                     ]
          [      2                                              ]
          [ g (i  + f h + c g) + d (h i + e h + b g) + a (g i + d h + a g) ]

          [                                                  2  ]
          [ h (c i + b f + a c) + e (c h + b e + a b) + b (c g + b d + a ) ]
          [                                                     ]
 Col 2 =  [                              2                      ]
          [ h (f i + e f + c d) + e (f h + e  + b d) + b (f g + d e + a d) ]
          [                                                     ]
          [      2                                              ]
          [ h (i  + f h + c g) + e (h i + e h + b g) + b (g i + d h + a g) ]
```

```
         [                                                            2  ]
         [ i (c i + b f + a c) + f (c h + b e + a b) + c (c g + b d + a ) ]
         [                                                               ]
 Col 3 = [                                2                              ]
         [ i (f i + e f + c d) + f (f h + e  + b d) + c (f g + d e + a d) ]
         [                                                               ]
         [     2                                                         ]
         [ i (i  + f h + c g) + f (h i + e h + b g) + c (g i + d h + a g) ]
```

これらの計算は機械的に行なわれるので、大きな行列の巾乗とか、巾乗の指数が大きい場合には計算機のリソースを大量に消費する。したがって、安易に使うことは推奨できない。理論的なアプローチ、例えば座標変換して対角行列にするなどのアプローチが可能かどうかを調べるなどすべきである。

4. 逆行列の計算

正方行列 A に対して、$AX = XA = I$ を満たす A と同じ型の正方行列 X のことを、A の逆行列といって A^{-1} と書く、ただし I は単位行列である (高校では E と書くかもしれない)。逆行列を **maxima** で形式的に求めることは簡単にできる。A を与えられた正方行列とするとき、A^{-1} を **maxima** で求めるには、A^^(-1); あるいは invert(A); とタイプすればよい。

例 15. (C1) `A : matrix([a, b], [c, d]);`

(D1) $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

(C2) `A^^(-1);`

(D2) $\begin{pmatrix} \frac{d}{ad-bc} & -\frac{b}{ad-bc} \\ -\frac{c}{ad-bc} & \frac{a}{ad-bc} \end{pmatrix}$

(C3) `B : matrix([1, 2], [2, 1]);`

(D3) $\begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$

(C4) `B^^(-1);`

(D4) $\begin{pmatrix} -\frac{1}{3} & \frac{2}{3} \\ \frac{2}{3} & -\frac{1}{3} \end{pmatrix}$

(C5) `invert(B);`

(D5) $\begin{pmatrix} -\frac{1}{3} & \frac{2}{3} \\ \frac{2}{3} & -\frac{1}{3} \end{pmatrix}$

ここで注意したいのは、もし $\Delta = ad - bc = 0$ ならば A^{-1} は存在しないのであるが、**maxima** は気にせず形式的計算を実行していることである。このあたりが、真の数学と計算機の理解している数学の差であり、使うものはよく注意する必要がある。これらは良く知られた Cramer の公式を用いて計算されている。これまた良く知られているように、Cramer の公式は理論的には重要であるが、計算量が大きいので、巾乗のところで注意したことがこの場合にもあてはまる。すなわち、大きな行列に対して安易にこの方法で計算することは勧められない。行列の基本変形などを使うことを推奨する。さて、付録として Cramer の公式を **maxima** で表現しておこう。expand(adjoint(A))/expand(determinant(A)) により A の逆行列を計算するのが Cramer の公式である。ここで adjoint(A) は A の余因子行列を、determinant(A) は A の行列式を表す。**expand** は多項式を展開するコマンドである。

例 16. (C1) `A : matrix([a, b], [c, d]);`

(D1) $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

(C2) `expand(adjoint(A))/expand(determinant(A));`

(D2) $\begin{pmatrix} \frac{d}{ad-bc} & -\frac{b}{ad-bc} \\ -\frac{c}{ad-bc} & \frac{a}{ad-bc} \end{pmatrix}$

(C3) `C : matrix([a, b, c], [d, e, f], [g, h, i]);`

(D3) $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$

(C4) `ratsimp( invert(C) - expand(adjoint(C))/expand(determinant(C)) );`

(D6) $$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

ここで、最後に使ったコマンド **ratsimp** は有理多項式を計算して簡約表示するものである。余因子行列、行列式については後で再び取り上げる。

5. 連立一次方程式

5.1. 連立一次方程式を解く. n 個の未知数 $x_1, x_2, \ldots, x_n$ について m 個の一次方程式からなる連立一次方程式

$$
\begin{cases}
a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & = b_1 \\
a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & = b_2 \\
\quad \cdots\cdots & \\
a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n & = b_m
\end{cases} \tag{3}
$$

を **maxima** で取り扱う方法について説明しよう。(3) を解くにはコマンド **solve** を使う。solve は汎用の方程式を解くためのコマンドで、連立一次方程式に限らず使用することができる。(解が求まるとは限らないが！) 一般的な構文は

(C1) solve([方程式１, 方程式２, ... , 方程式m], [未知数１, 未知数２, ... , 未知数n]);

である。ここで、方程式はそのままタイプすると面倒なので、eq1 : 方程式１ のように名前をつけて、solve([eq1, eq2, ... , eq2], [x, y, ... , z]); のように使うといい。

例 17. (C1) eq1 : a*x + b*y = 0;

(D1) $by + ax = 0$

(C2) eq2 : c*x + d*y = 0;

(D2) $dy + cx = 0$

(C3) solve([eq1, eq2], [x, y]);

(D3) $[[x = 0, y = 0]]$

(C4) eq3 : a*x + b*y = e;

(D4) $by + ax = e$

(C5) eq4 : c*x + d*y = f;

(D5) $dy + cx = f$

(C6) solve([eq3, eq4], [x, y]);

(D6) $\left[\left[x = -\frac{de-bf}{bc-ad}, y = \frac{ce-af}{bc-ad}\right]\right]$

(C7) solve([3*x-2*y+z=3, x+y-3*z=7, y+z=9], [x, y, z]);

(D7) $\left[\left[x = \frac{26}{5}, y = \frac{36}{5}, z = \frac{9}{5}\right]\right]$

(C8) solve([3*x-2*y+z=3, x+y-3*z=7], [x, y]);

(D8) $\left[\left[x = \frac{5z+17}{5}, y = \frac{10z+18}{5}\right]\right]$

(C9) solve([3*x - 2*y = 3, x + y = 7, x - y = 9], [x, y]);

```
Inconsistent equations:  (3)
-- an error.  Quitting.  To debug this try DEBUGMODE(TRUE);)
```

上の (C8) は未知数の数が方程式の数より多い場合であるが、このような場合には一般的に解に任意パラメータが含まれる。ここでは、z が任意パラメータである。逆に (C9) は未知数の数よりも方程式の数の方が多い過剰決定系とよばれる場合であるが、多くの場合解を持たない。この場合も解をもたないので、**maxima** が警告を発している。

5.2. 行列表現. 線形代数学で連立一次方程式を理論的に扱う場合には、連立一次方程式を行列とベクトルの間の方程式に表現して扱うのがスマートである。連立一次方程式 (3) に

対して、

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}, \qquad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}, \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}$$

と置くと、(3) は

$$A\mathbf{x} = \mathbf{b} \tag{4}$$

と書くことができる。これを (3) の行列表現と言う。ここで、A を (3) の係数行列という。また、次の行列

$$(A|\mathbf{b}) := \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_m \end{pmatrix}$$

を (3) の拡大係数行列という。**maxima** には、与えられた連立一次方程式から、係数行列および拡大係数行列を取り出すコマンドが用意されている。未知数 x, y, ... , z に対する方程式 eq1, eq2, ... , eqm からなる連立一次方程式の係数行列を求めるための構文は

```
(C1) A : coefmatrix([eq1, eq2, ... , eqm], [x, y, ... , z]);
```

である。これで記号 A は係数行列を意味することになる。また、拡大係数行列を求める構文は

```
(C1) Ab : augcoefmatrix([eq1, eq2, ... , eqm], [x, y, ... , z]);
```

である。これで記号 Ab は拡大係数行列を意味することになる。

注意 3. **maxima** の **augcoefmatrix** は $(A|\mathbf{b})$ ではなく、$(A|-\mathbf{b})$ を求める。しかし、階数の問題については同等である。

5.3. **行列の階数と連立一次方程式の解.** 与えられた連立一次方程式が解けるかどうかを、solve コマンドで実験する以前に調べるために、準備として行列の階数について説明しよう。任意の行列

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

に対して、(1) ある行と別のある行をいれかえる。(2) ある行を定数倍する。(3) ある行に別のある行を加える。という 3 種類の操作を施すことを行列 A の行基本変形という。これらの操作を繰り返すことにより、A を

$$A' = \begin{pmatrix} |\underline{1} & \overline{* \cdots *} & * \cdots & \cdots & \cdots & \cdots & * \\ 0 & 0 \cdots 0 & |\underline{1 * \cdots} & \cdots & \cdots & \cdots & * \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & * \\ 0 & \cdots & \cdots & \cdots 0 & |\underline{1 * \cdots} & \underline{\cdots} & \underline{*} \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

の形の行列に変形できる。A' を階段行列という。A' ですべての成分がゼロである行を除いた残りの階段状になっている部分の行数を A の階数という。これは、途中の変形の仕方によらず A によって一意に定まる数である。A の階数を $\mathrm{rank}(A)$ と書く。階数を用いると、連立一次方程式 (3) が解を持つための必要十分条件を簡明に $\mathrm{rank}(A) = \mathrm{rank}(A|\mathbf{b})$ といってしまうことができる。つまり、係数行列と拡大係数行列の階数が一致するとき、そのときに限って連立一次方程式は解を持つのである。さて、このことを使って **maxima** で遊ぶためのコマンドを説明しよう。コマンド **echelon** は、行列を階段行列に基本変形する。その構文は echelon(A); である。また、コマンド **rank** は行列の階数を求める。構文は rank(A); である。

まずは基本的な例から

例 18. (C1) `A : matrix([a, b], [c, d]);`

(D1) $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

(C2) `rank(A);`

(D2) 2

(C3) `echelon(A);`

(D3) $\begin{pmatrix} 1 & \frac{b}{a} \\ 0 & 1 \end{pmatrix}$

(C4) `B : matrix([1, 2, 3], [4, 5, 6], [7, 8, 9]);`

(D4) $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$

(C5) `rank(B);`

(D5) 2

(C6) `echelon(B);`

(D6) $\begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 0 \end{pmatrix}$

(C7) `C : matrix([1, 2, 3], [4, 5, 6], [7, 8, 0]);`

(D7) $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 0 \end{pmatrix}$

(C8) `rank(C);`

(D8) 3

(C9) `echelon(C);`

(D9) $\begin{pmatrix} 1 & \frac{8}{7} & 0 \\ 0 & 1 & 14 \\ 0 & 0 & 1 \end{pmatrix}$

次に例 17 の (C9) について、可解性の判定をしてみよう。

例 19. (C1) `eq1 : 3*x - 2*y = 3;`

(D1) $3x - 2y = 3$

(C2) `eq2 : x + y = 7;`

(D2) $y + x = 7$

(C3) `eq3 : x - y = 9;`

(D3) $x - y = 9$

(C4) `A : coefmatrix([eq1, eq2, eq3], [x, y]);`

(D4) $\begin{pmatrix} 3 & -2 \\ 1 & 1 \\ 1 & -1 \end{pmatrix}$

(C5) `Ab : augcoefmatrix([eq1, eq2, eq3], [x, y]);`

(D5) $\begin{pmatrix} 3 & -2 & -3 \\ 1 & 1 & -7 \\ 1 & -1 & -9 \end{pmatrix}$

(C6) `rank(A);`

(D6) 2

(C7) `rank(Ab);`

(D7) 3

(C8) `echelon(A);`

(D8) $\begin{pmatrix} 1 & -\frac{2}{3} \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$

(C9) `echelon(Ab);`

(D9) $\begin{pmatrix} 1 & -\frac{2}{3} & -1 \\ 0 & 1 & -\frac{18}{5} \\ 0 & 0 & 1 \end{pmatrix}$

係数行列と拡大係数行列の階数が違うから当然解は存在しない。solve を使う前に調べておけば良かった！

5.4. **同次連立一次方程式.** さて、連立一次方程式 (4) において、$\mathbf{b} = \mathbf{0}$ のとき、すなわち $A\mathbf{x} = \mathbf{0}$ のとき、特に同次連立一次方程式と言う。このとき、$\mathrm{rank}(A) = \mathrm{rank}(A|\mathbf{0})$ は明かであるから、同次連立一次方程式は必ず解を持つ。一般に連立一次方程式の解に含まれる任意パラメータの数は、未知数の数から係数行列の数をひいたものである。すなわち連立一次方程式 (4) の場合ならば、$n - \mathrm{rank}(A)$ である。したがって、$n = \mathrm{rank}(A)$ の場合は任意パラメータを含まない唯一の解を持つ。このとき、特にそれが同次連立方程式ならば、唯一の解は $x_1 = x_2 = \cdots = x_n = 0$ であり、自明解と呼ばれる。

連立方程式 (4) において、方程式の数と未知数の数が等しい場合、つまり $m = n$ の場合、係数行列 A は n 次正方行列になる。そして、この場合 $n = \mathrm{rank}(A)$ であることは $\det(A) \neq 0$ と同値であるので、解の個数が唯一つかどうかを調べるのに A の行列式 $\det(A)$ を使うことができる。さらに、特筆すべきは、$\det(A) \neq 0$ のとき連立一次方程式 (4) の解をあからさまに

$$\mathbf{x} = A^{-1}\mathbf{b}$$

と表現できることである。

例 20. (C1) `A : matrix([1, 2, 3], [4, 5, 6], [7, 8, 9]);`

(D1) $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$

(C2) `rank(A);`

(D2) 2

(C3) `solve([x+2*y+3*z=0, 4*x+5*y+6*z=0, 7*x+8*y+9*z=0], [x,y,z]);`

`Dependent equations eliminated:  (3)`

(D3) $[[x = \%\mathrm{R1}, y = -2\%\mathrm{R1}, z = \%\mathrm{R1}]]$

(C4) `eq1 : x + 2*y + 3*z = 0;`

(D4) $3z + 2y + x = 0$

(C5) `eq2 : 4*x + 5*y + 6*z = 0;`

(D5) $6z + 5y + 4x = 0$

(C6) `eq3 : 6*x + 7*y = 0;`

(D6) $7y + 6x = 0$

(C7) `B : coefmatrix([eq1, eq2, eq3], [x, y, z]);`

(D7) $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 6 & 7 & 0 \end{pmatrix}$

(C8) `rank(B);`

(D8) 3

(C9) `solve([eq1, eq2, eq3], [x, y, z]);`

(D9) $[[x = 0, y = 0, z = 0]]$

(C10) `eq4 : x + 2*y + 3*z = 10;`

(D10) $3z + 2y + x = 10$

(C11) `eq5 : 4*x + 5*y + 6*z = 11;`

(D11) $6z + 5y + 4x = 11$

(C12) `eq6 : 7*x + 8*y = 12;`

(D12) $8y + 7x = 12$

(C13) `C : coefmatrix([eq4, eq5, eq6], [x, y, z]);`

(D13) $$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 0 \end{pmatrix}$$

```
(C14) Cb : augcoefmatrix([eq4, eq5, eq6], [x, y, z]);
```

(D14) $$\begin{pmatrix} 1 & 2 & 3 & -10 \\ 4 & 5 & 6 & -11 \\ 7 & 8 & 0 & -12 \end{pmatrix}$$

```
(C15) rank(C) - rank(Cb);
```

(D15) 0

```
(C16) bb : matrix([10], [11], [12]);
```

(D16) $$\begin{pmatrix} 10 \\ 11 \\ 12 \end{pmatrix}$$

```
(C17) xx : C^^(-1) . bb;
```

(D17) $$\begin{pmatrix} -\frac{28}{3} \\ \frac{29}{3} \\ 0 \end{pmatrix}$$

```
(C18) solve([eq4, eq5, eq6], [x, y, z]);
```

(D18) $\left[\left[x = -\frac{28}{3}, y = \frac{29}{3}, z = 0\right]\right]$

```
(C19) determinant(A);
```

(D19) 0

```
(C20) determinant(B);
```

(D20) 24

```
(C21) determinant(C);
```

(D21) 27

注意 4. 公式 $\mathbf{x} = A^{-1}\mathbf{b}$ を使って解を求めるのは、solve を使うのより効率が悪い。

6. 正方行列

6.1. 行列式とトレース. 行列 A が n 次元正方行列であるというのは、もちろん A が (n,n) 型 行列

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$$

であることを意味する。ここで A の行列式 $\det(A)$ の定義をちゃんと書いておこう。

$$\det(A) := \sum_{\sigma \in \mathfrak{S}_n} \mathrm{sgn}(\sigma) a_{1\sigma(1)} a_{2\sigma(2)} \cdots a_{n\sigma(n)}$$

ここで $\mathfrak{S}_n$ は n 次対称群である。**maxima** の **determinant** は正方行列の行列式を計算するものであるが、上の定義をそのまま実行するものではない。内部的には、三角行列に変形して対角成分をすべて掛け合わせたものとして計算する。これは計算量の問題による。理論的には基本変形のうち、(1),(3) のみを使って三角行列に変形すれば符合を除いては行列式は変わらないことによる。**maxima** においては、コマンド **triangularize** により、与えられた行列にたいする上三角行列を得ることができる。**maxima** の triangularize は基本変形 (2) も使うので行列式の値は変形によって保存されないが、行列式がゼロの場合は変形しても行列式ゼロである。

例 21. (C1) `A : matrix([1, 2, 3], [4, 5, 6], [7, 8, 9]);`

(D1) $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$

(C2) `B : triangularize(A);`

(D2) $\begin{pmatrix} 1 & 2 & 3 \\ 0 & -3 & -6 \\ 0 & 0 & 0 \end{pmatrix}$

(C3) `determinant(A);`

(D3) 0

(C4) `determinant(B);`

(D4) 0

(C5) `C : matrix([1, 2, 3], [4, 5, 6], [7, 8, 0]);`

(D5) $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 0 \end{pmatrix}$

(C6) `D : triangularize(C);`

(D6) $\begin{pmatrix} 7 & 8 & 0 \\ 0 & 3 & 42 \\ 0 & 0 & -27 \end{pmatrix}$

(C7) `determinant(C);`

(D7) 27

(C8) `determinant(D);`

(D8) -567

(C9) `F : matrix([a, b, c], [d, e, f], [g, h, i]);`

(D9) $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$

(C10) `determinant(F);`

(D10) $a\,(ei - fh) - b\,(di - fg) + c\,(dh - eg)$

(C11) `G : triangularize(F);`

(D11) $\begin{pmatrix} a & b & c \\ 0 & ae - bd & af - cd \\ 0 & 0 & (ae - bd)\,i + (cd - af)\,h + (bf - ce)\,g \end{pmatrix}$

(C12) `ratsimp( determinant(F) - col(row(G, 3), 3) );`

(D12) $\begin{pmatrix} 0 \end{pmatrix}$

ここで、**col** 及び **row** はそれぞれ行列から指定された行および列を取り出すコマンドである。col(row(M, i), j) の様に使うと、行列の M の第 (i, j) 成分を取り出すことになる。

行列式と並んで、正方行列の重要な量であるトレースを考える。定義は簡単で、すべての対角成分の総和である。**maxima** で行列 A のトレースを求めるには、コマンド **mattrace** を用いて、mattrace(A); とタイプすればいい。使う前に load("nchrpl") とそれ用のパッケージを読み込む必要がある。

例 22. (C1) `load("nchrpl");`

(D1) `/usr/local/share/maxima/5.9.0/share/matrix/nchrpl.mac`

(C2) `A : matrix([a, b], [c, d]);`

(D2) $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

(C3) `mattrace(A);`

(D3) $d + a$

(C4) `B : matrix([a, b, c], [d, e, f], [g, h, i]);`

(D4) $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$

(C5) `mattrace(B);`

(D5) $i + e + a$

(C6) `sum(col(row(B, i), i), i, 1, 3);`

(D6) $\begin{pmatrix} i + e + a \end{pmatrix}$

高校で習った Caylay-Hamilton の定理を **maxima** で確かめておこう。証明すべき式は、任意の 2 次正方行列

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

に対して、式

$$A^2 - \mathrm{tr}(A)A + \det(A)I = O$$

である。上の例題に引続き、

例 23. (C7) `I : diagmatrix(2, 1);`

(D7) $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

(C8) `A^^2 - mattrace(A)*A + determinant(A)*I;`

(D8) $\begin{pmatrix} -a(d+a)+ad+a^2 & -b(d+a)+bd+ab \\ -c(d+a)+cd+ac & d^2-d(d+a)+ad \end{pmatrix}$
(C9) `ratsimp(D8);`
(D9) $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$

正方行列 A の行列式 $\det(A)$ とトレース $\mathrm{tr}(A)$ は、有用な性質をもっている。G を A と同じ型の正則行列 $(\det(G) \neq 0)$ とするとき、

$$\det(G^{-1}AG) = \det(A),$$
$$\mathrm{tr}(G^{-1}AG) = \mathrm{tr}(A)$$

A と B を同じ型の正方行列とするとき、

$$\det(AB) = \det(A)\det(B)$$
$$\mathrm{tr}(A+B) = \mathrm{tr}(A) + \mathrm{tr}(B)$$
$$\mathrm{tr}(AB) = \mathrm{tr}(BA)$$

次数が小さいときに、**maxima** で確かめておこう。

例 24. (C1) `A : matrix([a, b], [c, d]);`
(D1) $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$
(C2) `G : matrix([s, t], [u, v]);`
(D2) $\begin{pmatrix} s & t \\ u & v \end{pmatrix}$
(C3) `determinant(A) - determinant(G^^(-1) . A . G);`
(D3) $\left(\frac{s(du+cs)}{sv-tu} - \frac{u(bu+as)}{sv-tu}\right)\left(\frac{v(bv+at)}{sv-tu} - \frac{t(dv+ct)}{sv-tu}\right) - \left(\frac{(bu+as)v}{sv-tu} - \frac{t(du+cs)}{sv-tu}\right)\left(\frac{s(dv+ct)}{sv-tu} - \frac{u(bv+at)}{sv-tu}\right) + ad - bc$
(C4) `ratsimp(D3);`
(D4) 0
(C5) `B : matrix([a, b, c], [d, e, f], [g, h, i]);`
(D5) $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$
(C6) `F : matrix([o, p, q], [r, s, t], [u, v, w]);`
(D6) $\begin{pmatrix} o & p & q \\ r & s & t \\ u & v & w \end{pmatrix}$
(C7) `ratsimp( determinant(B) - determinant(F^^(-1) . B . F) );`
(D7) 0
(C8) `load("nchrpl");`
(D8) /usr/local/share/maxima/5.9.0/share/matrix/nchrpl.mac
(C9) `ratsimp(mattrace(A) - mattrace(G^^(-1) . A . G));`
(D9) 0
(C10) `ratsimp(mattrace(B) - mattrace(F^^(-1) . B . F));`
(D10) 0

例 25. (C1) `A : matrix([a, b], [c, d]);`

(D1) $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

(C2) `B : matrix([s, t], [u, v]);`

(D2) $\begin{pmatrix} s & t \\ u & v \end{pmatrix}$

(C3) `ratsimp(determinant(A . B) - determinant(A) * determinant(B));`

(D3) 0

(C4) `C : matrix([a,b,c],[d,e,f],[g,h,i]);`

(D4) $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$

(C5) `D : matrix([o,p,q],[r,s,t],[u,v,w]);`

(D5) $\begin{pmatrix} o & p & q \\ r & s & t \\ u & v & w \end{pmatrix}$

(C6) `ratsimp(determinant(C . D) - determinant(C) * determinant(D));`

(D6) 0

例 26. (C1) `A : matrix([a, b], [c, d]);`

(D1) $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

(C2) `B : matrix([s, t], [u, v]);`

(D2) $\begin{pmatrix} s & t \\ u & v \end{pmatrix}$

(C3) `load("nchrpl");`

(D3) `/usr/local/share/maxima/5.9.0/share/matrix/nchrpl.mac`

(C4) `mattrace(A . B);`

(D4) $dv + bu + ct + as$

(C5) `mattrace(B . A);`

(D5) $dv + bu + ct + as$

(C6) `mattrace(A . B) - mattrace(B . A);`

(D6) 0

(C7) `C : matrix([a, b, c], [d, e, f], [g, h, i]);`

(D7) $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$

(C8) `D : matrix([o, p, q], [r, s, t], [u, v, w]);`

(D9) $\begin{pmatrix} o & p & q \\ r & s & t \\ u & v & w \end{pmatrix}$

(C10) `mattrace(C . D);`

(D10) $iw + fv + cu + ht + es + br + gq + dp + ao$

(C11) `mattrace(D . C);`

(D11) $iw + fv + cu + ht + es + br + gq + dp + ao$

(C12) `mattrace(C . D) - mattrace(D . C);`

(D12) 0

補足 1. 与えられた行列 A の行列式を求めるもう一つのコマンド **newdet** が用意されている。実は、**newdet** の方が **determinant** より良いアルゴリズムを使っている。**determinant** でうまく計算できないときには **newdet** を使ってみるといい。

6.2. **転置行列、対称行列、交代行列、エルミート行列、歪エルミート行列.** 行列 A の行と列を入れ換えたものを A の転置行列と言って ${}^tA$ と書いた。**maxima** で、与えられた行列 A の転置行列を求めるには、コマンド **transpose** を使う。

例 27. (C1) `A : matrix([a, b], [c, d]);`

(D1) $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

(C2) `At : transpose(A);`

(D2) $\begin{pmatrix} a & c \\ b & d \end{pmatrix}$

(C3) `B : matrix([a, b, c], [d, e, f], [g, h, i]);`

(D3) $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$

(C4) `Bt : transpose(B);`

(D4) $\begin{pmatrix} a & d & g \\ b & e & h \\ c & f & i \end{pmatrix}$

(C5) `C : matrix([1, 2, 3, 4], [5, 6, 7, 8]);`

(D5) $\begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{pmatrix}$

(C6) `Ct : transpose(C);`

(D6) $\begin{pmatrix} 1 & 5 \\ 2 & 6 \\ 3 & 7 \\ 4 & 8 \end{pmatrix}$

正方行列 A が条件 $A = {}^tA$ を満たすとき、A を対称行列と言う。また、A が条件 $A = -{}^tA$ を満たすならば A を交代行列と言う。任意の正方行列 A に対して、

$$B = \frac{A + {}^tA}{2}, \qquad C = \frac{A - {}^tA}{2}$$

と置くと、B と C はそれぞれ、対称行列と交代行列になる。これを、A の対称化と交代化と言う。$A = B + C$ のように、任意の正方行列は対称行列と交代行列の和の形に書ける。**maxima** には残念ながら、与えられた正方行列を対称化、交代化するコマンドは用意されていないが、**maxima** の持つプログラム機能を用いて、自分自身の新たなコマンドとして実現できる。

例 28. (C1) `symm(X) := (1/2)*(X + transpose(X));`

(D1) $\mathrm{symm}\,(X) := \frac{1}{2}\,(X + \mathrm{TRANSPOSE}\,(X))$

(C2) `A : matrix([1, 2], [3, 4]);`

(D2) $\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$

(C3) symm(A);

(D3) $\begin{pmatrix} 1 & \frac{5}{2} \\ \frac{5}{2} & 4 \end{pmatrix}$

(C4) B : matrix([3, -1, 2], [4, -2, 1], [9, 9, 5]);

(D4) $\begin{pmatrix} 3 & -1 & 2 \\ 4 & -2 & 1 \\ 9 & 9 & 5 \end{pmatrix}$

(C5) symm(B);

(D5) $\begin{pmatrix} 3 & \frac{3}{2} & \frac{11}{2} \\ \frac{3}{2} & -2 & 5 \\ \frac{11}{2} & 5 & 5 \end{pmatrix}$

(C6) altm(X) := (1/2)*(X - transpose(X));

(D6) $\mathrm{altm}\,(X) := \frac{1}{2}\,(X - \mathrm{TRANSPOSE}\,(X))$

(C7) altm(A);

(D7) $\begin{pmatrix} 0 & -\frac{1}{2} \\ \frac{1}{2} & 0 \end{pmatrix}$

(C8) altm(B);

(D8) $\begin{pmatrix} 0 & -\frac{5}{2} & -\frac{7}{2} \\ \frac{5}{2} & 0 & -4 \\ \frac{7}{2} & 4 & 0 \end{pmatrix}$

(C9) A - (symm(A) + altm(A));

(D9) $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$

(C10) B - (symm(B) + altm(B));

(D10) $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$

この例では、(C1) で対称化のコマンド **symm** を、(C6) で交代化のコマンド **altm** を定義した。

与えられた行列 A に対して、各成分をその共役複素数で置き換えてできる行列を A の複素共役行列といい、$\overline{A}$ と書く。**maxima** で、与えられた行列の複素共役行列をもとめるには、コマンド **conjugate** を使う。conjugate を使うためには、その前に load(eigen); により関連パッケージをロードしておく必要がある。

例 29. (C1) load(eigen);

```
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
```

(D1) /usr/local/share/maxima/5.9.0/share/matrix/eigen.mac

(C2) A : matrix([1 - %I, 2 + 3*%I], [%I, 4]);

(D2) $\begin{pmatrix} 1-i & 3i+2 \\ i & 4 \end{pmatrix}$

(C3) conjugate(A);

(D3) $\begin{pmatrix} i+1 & 2-3i \\ -i & 4 \end{pmatrix}$

正方行列 A が条件 $A = {}^t\overline{A}$ を満たすとき、A をエルミート行列と言う。また、A が条件 $A = -{}^t\overline{A}$ を満たすならば A を歪エルミート行列と言う。A を $A = Re + Im$ と実数部分 Re と純虚数部分に分けて考えることにしよう。A がエルミート行列ならば、

$$Re + Im = A = {}^t\overline{A} = {}^tRe - {}^tIm$$

であるから、Re は対称行列、Im は交代行列になる。また、A が歪エルミート行列ならば、

$$Re + Im = A = -{}^t\overline{A} = -{}^tRe + {}^tIm$$

であるから、Re は交代行列、Im は対称行列になる。このことから、逆に任意の正方行列 A に対し、

$$B = \frac{Re + {}^tRe}{2} + \frac{Im - {}^tIm}{2},$$
$$C = \frac{Re - {}^tRe}{2} + \frac{Im + {}^tIm}{2},$$

と置くと、B はエルミート行列、C は歪エルミート行列になり、$A = B + C$ が成立する。このような、B、C を求めるコマンドを作成してみよう。

例 30. (C1) `load("nchrpl");`

(D1) /usr/local/share/maxima/5.9.0/share/matrix/nchrpl.mac

(C2) `load(eigen);`

Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS

(D2) /usr/local/share/maxima/5.9.0/share/matrix/eigen.mac

(C3) `symm(X) := (1/2)*(X + transpose(X));`

(D3) $\mathrm{symm}\,(X) := \frac{1}{2}\,(X + \mathrm{TRANSPOSE}\,(X))$

(C4) `altm(X) := (1/2)*(X - transpose(X));`

(D4) $\mathrm{altm}\,(X) := \frac{1}{2}\,(X - \mathrm{TRANSPOSE}\,(X))$

(C5) `ream(X) := (1/2)*(X + conjugate(X));`

(D5) $\mathrm{ream}\,(X) := \frac{1}{2}\,(X + \mathrm{CONJUGATE}\,(X))$

(C6) `imgm(X) := (1/2)*(X - conjugate(X));`

(D6) $\mathrm{imgm}\,(X) := \frac{1}{2}\,(X - \mathrm{CONJUGATE}\,(X))$

(C7) `herm(X) := symm(ream(X)) + altm(imgm(X));`

(D7) $\mathrm{herm}\,(X) := \mathrm{symm}\,(\mathrm{ream}\,(X)) + \mathrm{altm}\,(\mathrm{imgm}\,(X))$

(C8) `skew(X) := altm(ream(X)) + symm(imgm(X));`

(D8) $\mathrm{skew}\,(X) := \mathrm{altm}\,(\mathrm{ream}\,(X)) + \mathrm{symm}\,(\mathrm{imgm}\,(X))$

(C9) `A : matrix([1 + %I, 2 - 3*%I], [2, 4*%I]);`

(D9) $\begin{pmatrix} i+1 & 2-3i \\ 2 & 4i \end{pmatrix}$

(C10) `herm(A);`

(D10) $\begin{pmatrix} 1 & 2-\frac{3i}{2} \\ \frac{3i}{2}+2 & 0 \end{pmatrix}$

(C11) `skew(A);`

(D11) $\begin{pmatrix} i & -\frac{3i}{2} \\ -\frac{3i}{2} & 4i \end{pmatrix}$

(C12) `A - (herm(A) + skew(A));`

(D12) $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$

注意 5. **maxima** で新しいコマンド (**maxima**、Lisp での関数のこと) を定義する最も簡単な方法は

$$\text{コマンドの名前}(\text{引数１}, \text{引数 2}, \ldots) := \text{引数１、引数 2、... の式};$$

である。一つの式で定義できない場合は、block 文

$$\text{コマンドの名前}(\text{引数１}, \text{引数 2}, \ldots) := block([\text{局所変数の列}], \\ \text{引数１、引数 2、... の式１}, \ldots, \text{引数１、引数 2、... の式 N});$$

を使う。最後の" 引数１、引数 2、... の式 N" の値がコマンドの返り値になる。

7. maxima 内部における行列の扱いについて

7.1. **配列とリストと行列の関係について.** リストはデータを並べてできるデータのこと、あるいはそのデータ型のことである。リストは **maxima** の最も基本的な構成要素であり、既存のデータ型から新しいデータ型を生成する汎用の糊の役割を果たす。**maxima** でのリストの表現は [a, b, ... , z] のようにデータの並びを "[" と "]" で挟んで表す。**maxima** には、リストを操作するコマンドがいくつも用意されているが、詳しいことはマニュアルにゆずり、用があるときにはその都度使い方を示すことにする。

例 31. (C1) `a : [1, 2, 3];`
(D1) $[1,2,3]$
(C2) `a;`
(D2) $[1,2,3]$

配列 (**array**) は、いくつかの添字により指し示されたデータの集まりで、a[i, j, ..., n] の形で参照できるデータ、あるいはそのデータ型のことである。ここで、添字 i, j, ... , k の数は最大で 5 個まで可能である。添字の数が n 個の場合 n 次元配列とよばれる。配列の詳細もマニュアルにゆずることにする。

例 32. (C3) `b[i] := i;`
(D3) $b_i := i$
(C4) `b;`
(D4) b
(C5) `b[5];`
(D5) 5
(C6) `c[i,j] := i*j;`
(D6) $c_{i,j} := ij$
(C7) `c;`
(D7) c
(C8) `c[3,4];`
(D8) 12

リストと配列は異なるデータ型であるが、**maxima** では、名前付きのリストの各要素にアクセスするのに、そのリストを一次元配列のように取り扱うことができる。例 31 の a は長さ 3 のリストであるが、配列ではない。しかし、a[i] の書式で配列のようにアクセスすることができるのである。

例 33. (C9) `a[1];`
(D9) 1
(C10) `a[2];`
(D10) 2
(C11) `a[3];`
(D11) 3
(C12) `listp(a);`
(D13) true
(C14) `listp(b);`
(D14) false

ここで使ったコマンド **listp** は、その記号の値がリストなのか、そうでないのかをチェックするときに使うものである。式 listp(a); は、もし a の値がリストならば値 **true** を返し、もしそうでなければ値 **false** を返す。

リストというのは、非常に柔軟なしくみである、それ自体の入れ子を許すので、リストのリスト、リストのリストのリスト等 いくらでも考えることができる。

例 34.
```
(C1) a : [ [1, 2, 3], [4, 5, 6], [7, 8, 9] ];
(D1) [[1,2,3],[4,5,6],[7,8,9]]
(C2) a[1];
(D2) [1,2,3]
(C3) a[2];
(D3) [4,5,6]
(C4) a[3];
(D4) [7,8,9]
(C5) a[1][1];
(D5) 1
(C6) a[1][2];
(D6) 2
(C7) a[1][3];
(D7) 3
(C8) a[2][1];
(D8) 4
(C9) a[2][2];
(D9) 5
(C10) a[2][3];
(D10) 6
(C11) a[3][1];
(D11) 7
(C12) a[3][2];
(D12) 8
(C13) a[3][3];
(D13) 9
(C14) a[2,3];
Wrong number of indices:
[2,3]
-- an error.  Quitting.  To debug this try DEBUGMODE(TRUE);)
```

上の例で分かるように名前のついたリストのリストの要素には、それを配列の配列としてアクセスできることがわかる。上の例より a[i][j] は (a[i])[j] を意味する。しかし、2 次元配列としては扱うことができないことを示している。

行列はリストのリストでも、配列の配列でも、2 次元配列でもないが、ある行列のメンバーにアクセスするのに、それをリストのリスト、配列の配列あるいは 2 次元配列と思ってアクセスすることが可能である。

例 35.
```
(C1) A : matrix([a, b, c, d], [e, f, g, h], [i, j, k, l]);
```

```
(D1)
```
$$\begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{pmatrix}$$

```
(C2) A[1];
(D2) [a, b, c, d]
(C3) A[2];
(D3) [e, f, g, h]
(C4) A[3];
(D4) [i, j, k, l]
(C5) A[2][3];
(D5) g
(C6) A[2,3];
(D6) g
(C7) matrixp(A);
(D7) true
(C8) listp(A);
(D8) false
(C9) matrixp(A[1]);
(D9) false
(C10) listp(A[1]);
(D10) true
(C11) B : [[a,b,c,d],[e,f,g,h],[i,j,k,l]];
(D11) [[a, b, c, d] , [e, f, g, h] , [i, j, k, l]]
(C12) B[1];
(D12) [a, b, c, d]
(C13) B[2];
(D13) [e, f, g, h]
(C14) B[3];
(D14) [i, j, k, l]
(C15) B[2][3];
(D15) g
(C16) B[2,3];
Wrong number of indices:
[2, 3]
-- an error.  Quitting.  To debug this try DEBUGMODE(TRUE);)
(C17) matrixp(B);
(D17) false
(C18) listp(B);
(D18) true
(C19) matrixp(B[1]);
(D19) false
(C20) listp(B[1]);
(D20) true
```

ここで使ったコマンド **matrixp** は、その記号の値が行列なのか、そうでないのかをチェックするときに使うものである。式 listp(a); は、もし a の値が行列ならば値 **true** を返し、もしそうでなければ値 **false** を返す。

maxima における行列型のこのような柔軟な取り扱いにより、様々な操作が可能になる。簡単な応用として、行基本変形を扱ってみよう。ここでは、まず 3 種類の n 次正則行列 P(i,j,n), Q(a,i,n), R(a,i,j,n), $(i \neq j)$ を定義する。(n,k) 行列 M に対して、基本変形 (1) "第 i 行と第 j 行を入れ換える" ことと P(i,j,n) を M に左から掛けることは同値である。(n,k) 行列 M に対して、基本変形 (2) "第 i 行を a 倍する" ことと Q(a,i,n) を M に左から掛けることは同値である。(n,k) 行列 M に対して、基本変形 (3) "第 i 行に第 j 行の a 倍を加える" ことと R(a,i,j,n) を M に左から掛けることは同値である。例で見てみよう。まず、次のような内容のファイルをつくり、"em.mac" と名前を付けて保存する。

```
/* em.mac */

P(i, j, n) := block([I, A],
    I : ident(n),
    A : zeromatrix(n, n),
        A[i][i] : -1, A[j][j] : -1,
        A[i][j] :  1, A[j][i] : 1,
    I + A )$

Q(a, i, n) := block([A],
    A : ident(n),
        A[i][i] : a,
    A )$

R(a, i, j, n) := block([I,A],
    I : ident(n),
    A : zeromatrix(n,n),
        A[i][j] : a,
    I + A )$
```

これだけ準備しておくと以下のような計算ができる。

例 36. (C1) `load("./em.mac")$`

(C2) `M : matrix([1,2,3],[4,5,6],[7,8,9]);`

(D2) $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$

(C3) `M1 : R(-4, 2, 1, 3) . M;`

(D3) $\begin{pmatrix} 1 & 2 & 3 \\ 0 & -3 & -6 \\ 7 & 8 & 9 \end{pmatrix}$

(C4) `M2 : R(-7, 3, 1, 3) . M1;`

(D4) $\begin{pmatrix} 1 & 2 & 3 \\ 0 & -3 & -6 \\ 0 & -6 & -12 \end{pmatrix}$

```
(C5) M3 : R(-2, 3, 2, 3) . M2;
```
(D5) $\begin{pmatrix} 1 & 2 & 3 \\ 0 & -3 & -6 \\ 0 & 0 & 0 \end{pmatrix}$

```
(C6) M4 : Q((-1/3), 2, 3) . M3;
```
(D6) $\begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 0 \end{pmatrix}$

```
(C7) echelon(M);
```
(D7) $\begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 0 \end{pmatrix}$

```
(C8) P(2,3,5);
```
(D8) $\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$

```
(C9) P(2,4,5);
```
(D10) $\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$

```
(C11) Q(3,2,5);
```
(D11) $\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$

```
(C12) R(3,2,4,5);
```
(D12) $\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 3 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$

```
(C13) determinant(P(2,4,5));
```
(D13) -1

```
(C14) determinant(Q(3,2,5));
```
(D14) 3

```
(C15) determinant(R(3,2,4,5));
```
(D15) 1

7.2. **行と列の操作について.** この小節では **maxima** で行列の行と列を扱う方法を述べよう。**maxima** にもともと備わっているコマンドに **row** 及び **col** がある。与えられた行列 A に対して、row(A, i); とタイプすると A の第 i 行が得られる、また、col(A, j); とすれ

ば A の第 j 列が得られる。厳密に言えば、row と col が返すものは、それぞれ $(m,1)$ 型および $(1,n)$ 型の行列である。

例 37. (C1) `A : matrix([a, b, c], [d, e, f], [g, h, i]);`

(D1) $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$

(C2) `col(A, 1);`

(D2) $\begin{pmatrix} a \\ d \\ g \end{pmatrix}$

(C3) `matrixp(col(A, 1));`

(D3) true

(C4) `listp(col(A, 1));`

(D4) false

(C5) `row(A, 1);`

(D5) $\begin{pmatrix} a & b & c \end{pmatrix}$

(C6) `matrixp(row(A, 1));`

(D6) true

(C7) `listp(row(A, 1));`

(D7) false

(C8) `col(A,1)[1];`

(D8) $[a]$

(C9) `col(A,1)[1][1];`

(D9) a

(C10) `col(A,1)[2][1];`

(D10) d

(C11) `col(A,1)[3][1];`

(D11) g

(C12) `row(A,1)[1];`

(D12) $[a, b, c]$

(C13) `row(A,1)[1][2];`

(D13) b

(C14) `row(A,1)[1][3];`

(D14) c

一方、A[i]; は行列 A の i 行を返すが、これは型としては 1 次元配列であり、行列ではない。row(A, i); と A[i]; の返す結果の見ための同一性には注意が必要である。コマンド **transpose** は行列の転置を返すものであるが、1 次元配列を引数として与えると $(m,1)$ 型の行列を返す。したがって、transpose(A[i]); は行列になる。

例 38. (C15) `A[1];`

(D15) $[a, b, c]$

(C16) `matrixp(A[1]);`

(D16) false

(C17) `listp(A[1]);`

(D17) true

(C18) `transpose(A[1]);`

(D18) $\begin{pmatrix} a \\ b \\ c \end{pmatrix}$

(C19) `matrixp(transpose(A[1]));`

(D19) true

col を使わないで A の第 j 列を取り出すには、transpose(transpose(A)[j]); とタイプすればよい。

例 39. (C20) `transpose(transpose(A)[1]);`

(D20) $\begin{pmatrix} a \\ d \\ g \end{pmatrix}$

以上で行、列を抽出することはできるようになった。それでは、指定の行、列に値を設定するにはどうしたらよいのか、また行基本変形についてはすでに前小節で触れたが、列基本変形についてはどうか、さらには、もっと一般的な操作、行や列の削除、追加といったことを **maxima** で実現するにはどうすればよいか？以下では、こういったことを考えていこう。

まず準備として、与えられた行列から指定したブロックを抽出するコマンド **matblock** と、指定したブロックにブロックデータを上書きするコマンド **matreplace** を新たに定義する。matblock(A, i1, i2, j1, j2); は、行列

$$A = \begin{pmatrix} a_{1,1} & \cdots & a_{1,j1} & \cdots & a_{1,j2} & \cdots & a_{1,n} \\ \vdots & & \vdots & & \vdots & & \vdots \\ a_{i1,1} & \cdots & a_{i1,j1} & \cdots & a_{i1,j2} & \cdots & a_{i1,n} \\ \vdots & & \vdots & & \vdots & & \vdots \\ a_{i2,1} & \cdots & a_{i2,j1} & \cdots & a_{i2,j2} & \cdots & a_{i2,n} \\ \vdots & & \vdots & & \vdots & & \vdots \\ a_{m,1} & \cdots & a_{m,j1} & \cdots & a_{m,j2} & \cdots & a_{m,n} \end{pmatrix}$$

から、行列ブロック

$$\begin{pmatrix} a_{i1,j1} & \cdots & a_{i1,j2} \\ \vdots & & \vdots \\ a_{i2,1} & \cdots & a_{i2,j1} \end{pmatrix}$$

を抽出するものだ。逆に、matreplace(A, B, i1, i2, j1, j2) は A の上の行列ブロック部分を、行列

$$B = \begin{pmatrix} b_{1,j2-j1+1} & \cdots & a_{1,j2-j1+1} \\ \vdots & & \vdots \\ a_{i2-i1+1,1} & \cdots & a_{i2-i1+1,j2-j1+1} \end{pmatrix}$$

で上書きするものだ。次のような matblock と matreplace の定義を書いたファイルを用意し、“matblock.mac” と名前をつけよう。

```
/* matblock.mac */
matblock(Mat,i1,i2,j1,j2) := block([_A, _m, _n, _i, _j],
```

```
    _m : i2 - i1, _n : j2 - j1,
    _A : zeromatrix(_m + 1, _n + 1),
    for _i : 1 thru _m + 1 do
        for _j : 1 thru _n + 1 do
            _A[_i][_j] : Mat[i1 + _i - 1][j1 + _j - 1],
    _A
)$

matreplace(Mat1,Mat2,i1,i2,j1,j2) := block([_A, _m, _n, _i, _j],
    _m : i2 - i1, _n : j2 - j1,
    _A : copymatrix(Mat1),
    for _i : 1 thru _m + 1 do
        for _j : 1 thru _n + 1 do
            _A[i1 + _i - 1][j1 + _j - 1] : Mat2[_i][_j],
    _A
)$
```

これらのコマンドを使うときには、事前に load(“matblock.mac”); とタイプして、定義ファイルを読み込んでおく。例をあげよう。

例 40. (C1) `A : matrix([q,w,e,r,t],[y,u,i,o,p],[a,s,d,f,g],[h,i,j,k,l],[z,x,c,v,b]);`

(D1) $$\begin{pmatrix} q & w & e & r & t \\ y & u & i & o & p \\ a & s & d & f & g \\ h & i & j & k & l \\ z & x & c & v & b \end{pmatrix}$$

(C2) `load("./matblock.mac");`

(D2) ./matblock.mac

(C3) `matblock(A,2,4,2,4);`

(D3) $$\begin{pmatrix} u & i & o \\ s & d & f \\ i & j & k \end{pmatrix}$$

(C4) `matblock(A,2,2,2,2);`

(D4) $\begin{pmatrix} u \end{pmatrix}$

(C5) `matblock(A,2,5,3,4);`

(D5) $$\begin{pmatrix} i & o \\ d & f \\ j & k \\ c & v \end{pmatrix}$$

(C6) `B : matrix([1,2,3],[4,5,6],[7,8,9]);`

(D6) $$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

(C7) `C : matrix([1,2],[3,4]);`

(D7) $$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

```
(C8) matreplace(A,B,2,4,2,4);
```
(D8)
$$\begin{pmatrix} q & w & e & r & t \\ y & 1 & 2 & 3 & p \\ a & 4 & 5 & 6 & g \\ h & 7 & 8 & 9 & l \\ z & x & c & v & b \end{pmatrix}$$
```
(C9) matreplace(A,C,1,2,1,2);
```
(D9)
$$\begin{pmatrix} 1 & 2 & e & r & t \\ 3 & 4 & i & o & p \\ a & s & d & f & g \\ h & i & j & k & l \\ z & x & c & v & b \end{pmatrix}$$

matreplace を使えば、適切な大きさの行、列ベクトルを用意しておいて、それを与えられた行列に張り付けることができることがわかるであろう。注意することは、与えられた行列は matreplace で変化しないと言うことである。あくまで、そのコピーの値を変えるだけである。

列基本変形には、前小節で定義した正則行列 P(i,j,n), Q(a,i,n), R(a,i,j,n) をそのまま使うことができる。(k,n) 型行列 M に対して、(1) "第 i 列と第 $1j$ 列を入れ換える" ことと P(i,j,n) を M に右から掛けることは同値である。(2) "第 i 列を a 倍する" ことと Q(a,i,n) を M に右から掛けることは同値である。(3) "第 i 列に第 j 列の a 倍を加える" ことと R(a,j,i,n) を M に右から掛けることは同値である。例をあげよう

例 41.
```
(C1) load("./em.mac");
(D1) ./em.mac
(C2) M : matrix([1,2,3],[4,5,6],[7,8,9]);
```
(D2)
$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$
```
(C3) M1 : M . R(-2, 1, 2, 3);
```
(D3)
$$\begin{pmatrix} 1 & 0 & 3 \\ 4 & -3 & 6 \\ 7 & -6 & 9 \end{pmatrix}$$
```
(C4) M2 : M1 . R(-3, 1, 3, 3);
```
(D4)
$$\begin{pmatrix} 1 & 0 & 0 \\ 4 & -3 & -6 \\ 7 & -6 & -12 \end{pmatrix}$$
```
(C5) M3 : M2 . R(-2, 2, 3, 3);
```
(D5)
$$\begin{pmatrix} 1 & 0 & 0 \\ 4 & -3 & 0 \\ 7 & -6 & 0 \end{pmatrix}$$
```
(C6) M4 : M3 . Q((-1/3), 2, 3);
```
(D6)
$$\begin{pmatrix} 1 & 0 & 0 \\ 4 & 1 & 0 \\ 7 & 2 & 0 \end{pmatrix}$$
```
(C7) M . P(1, 3, 3);
```

(D7) $\begin{pmatrix} 3 & 2 & 1 \\ 6 & 5 & 4 \\ 9 & 8 & 7 \end{pmatrix}$

行列に新たに行や列を付け加えるには、コマンド **addrow** と **addcol** を使う。**addrow** の書式は addrow(M, L1, L2, ...); である。ここで、M を (m, n) 型行列とするとき、L1, L2, ... は長さ n のリスト、あるいは (l, n) 型の行列で、順に M の最下行に加えられたものが addrow の返り値となる。**addcol** の書式は addcol(M, L1, L2, ...); である。ここで、M を上記の通りとするとき、L1, L2 , ... は長さ m のリスト、あるいは (m, l) 型の行列で、もしそれがリストのときは transpose(L1), transpose(L2), ... と変換されて、M の最右列に順次追加されたものが addcol の返り値となる。

例 42. (C1) `A : matrix([a,b,c],[d,e,f],[g,h,i],[j,k,l]);`

(D1) $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \end{pmatrix}$

(C2) `B : matrix([1,2,3],[4,5,6]);`

(D2) $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$

(C3) `C : matrix([1,2],[3,4],[5,6],[7,8]);`

(D3) $\begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \\ 7 & 8 \end{pmatrix}$

(C4) `addrow(A, C);`

```
Incompatible structure - ADDROW//ADDCOL
-- an error.  Quitting.  To debug this try DEBUGMODE(TRUE);)
```

(C6) `addrow(A, B);`

(D6) $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \\ 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$

(C7) `addcol(A, C);`

(D7) $\begin{pmatrix} a & b & c & 1 & 2 \\ d & e & f & 3 & 4 \\ g & h & i & 5 & 6 \\ j & k & l & 7 & 8 \end{pmatrix}$

(C8) `addrow(A, [1, 2, 3], [4, 5, 6]);`

(D8) $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \\ 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$

(C9) addcol(A, [1, 3, 5, 7], [2, 4, 6, 8]);

(D9) $\begin{pmatrix} a & b & c & 1 & 2 \\ d & e & f & 3 & 4 \\ g & h & i & 5 & 6 \\ j & k & l & 7 & 8 \end{pmatrix}$

matblock、**matreplace**、**addrow** と **addcol** を組み合わせて使えば、大抵の行列の切り貼りができることは理解されよう。個々の場合にどうすればいいかは読者への問題としておこう。

一つ応用を挙げておく。行列 A が

$$A = \begin{pmatrix} R & S \\ O & T \end{pmatrix}$$

の形をしているとしよう。ここで、A,R および T は正方行列とし、O の成分はすべてゼロとする。このとき、一般に

$$\det(A) = \det(R)\det(S)$$

がなり立つ。具体例で確認してみよう。

例 43. (C1) R : matrix([a, b], [c, d]);

(D1) $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

(C2) T : matrix([r, s, t], [u, v, w], [x, y, z]);

(D2) $\begin{pmatrix} r & s & t \\ u & v & w \\ x & y & z \end{pmatrix}$

(C3) S : matrix([h, i, j], [k, l, m]);

(D3) $\begin{pmatrix} h & i & j \\ k & l & m \end{pmatrix}$

(C4) O : zeromatrix(3, 2);

(D4) $\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$

(C5) A : addrow(addcol(R, S), addcol(O, T));

(D5) $\begin{pmatrix} a & b & h & i & j \\ c & d & k & l & m \\ 0 & 0 & r & s & t \\ 0 & 0 & u & v & w \\ 0 & 0 & x & y & z \end{pmatrix}$

(C6) determinant(R);

(D6) $ad - bc$

(C7) determinant(T);

(D7) $r\,(vz - wy) - s\,(uz - wx) + t\,(uy - vx)$

(C8) ratsimp(determinant(R)*determinant(T));

(D8) $((ad - bc)\,rv + (bc - ad)\,su)\,z + ((bc - ad)\,rw + (ad - bc)\,tu)\,y + ((ad - bc)\,sw + (bc - ad)\,tv)\,x$

(C9) determinant(A);

(D9) $ad\,(r\,(vz - wy) - s\,(uz - wx) + t\,(uy - vx)) - bc\,(r\,(vz - wy) - s\,(uz - wx) + t\,(uy - vx))$

(C10) ratsimp(D8 - D9);
(D10) 0

与えられた正方行列 A の (i,j) 成分 a_{ij} を含む行と列を取り除いてできる小行列 $A_{i,j}$ をもとめるには、**maxima** に用意されたコマンド **minor** をつかう。書式は minor(A,i,j); である。ちなみに、行列 $B=(b_{ij}),\quad b_{ij}:=(-1)^{i+j}\det(A_{ji}),\forall i,j$ を A の余因子行列と言うのであった。A の余因子行列を求めるには、コマンド **adjoint** を使って、adjoint(A); とタイプすればいい。

例 44. (C1) A : matrix([a,b,c,d],[e,f,g,h],[i,j,k,l],[m,n,o,p]);

(D1) $\begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{pmatrix}$

(C2) minor(A,1,1);

(D2) $\begin{pmatrix} f & g & h \\ j & k & l \\ n & o & p \end{pmatrix}$

(C3) minor(A,2,3);

(D3) $\begin{pmatrix} a & b & d \\ i & j & l \\ m & n & p \end{pmatrix}$

例 45. (C1) A : matrix([a, b], [c, d]);

(D1) $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

(C2) B : zeromatrix(2, 2);

(D2) $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$

(C3) B[1][1] : (-1)^(1+1)*determinant(minor(A,1,1));
(D3) d
(C4) B[1][2] : (-1)^(1+2)*determinant(minor(A,2,1));
(D4) $-b$
(C5) B[2][1] : (-1)^(2+1)*determinant(minor(A,1,2));
(D5) $-c$
(C6) B[2][2] : (-1)^(2+2)*determinant(minor(A,2,2));
(D6) a
(C7) B;

(D7) $\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$

(C8) adjoint(A);

(D8) $\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$

(C9) B - adjoint(A);

(D9) $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$

さて、理論的には

$$A^{-1} = \frac{A\text{ の余因子行列}}{A\text{ の行列式}}$$

あった。**maxima** の **adjoint** と **invert** は、Lisp ではなく、**maxima** 言語を用いて書かれたユーティリティである。そのソースコードは短くて分かりやすいので、ここに挙げておく。

例 46.
```
/* the following routines compute inverses and adjoints of matrices */
/* if ratmx is false [the default] then the elements will not be converted
to cre-form */

adjoint(mat):= block([adj,n], n:length(mat), adj:ident(n),
                     if n#1 then
                       for i thru n do for j thru n do
                           adj[i,j]:(-1)^(i+j)*determinant(minor(mat,j,i)),
                     adj)$

invert(mat):= block([adj,ans], adj:adjoint(mat),
                        ans:block([scalarmatrixp:true],
                               adj/(row(mat,1).col(adj,1))),
                 if scalarmatrixp=true and length(mat)=1
/* row(mat,1).col(adj,1) = determinant(mat) */
                         then ans[1,1] else ans)$
```

複数の行と列を一挙に削除する場合には、コマンド **submatrix** を使う。与えられた行列 M から 第 $m1, m2, \ldots$ 行と 第 $n1, n2, \ldots$ 列を削除した行列を得るには submatrix(m1, m2 , ..., M, n1, n2, ...); とタイプする。

例 47. `(C1) A : matrix([a,b,c,d,e],[f,g,h,i,j],[k,l,m,n,p],[p,q,r,s,t],[u,v,w,x,y]);`

(D1) $\begin{pmatrix} a & b & c & d & e \\ f & g & h & i & j \\ k & l & m & n & p \\ p & q & r & s & t \\ u & v & w & x & y \end{pmatrix}$

`(C2) B : submatrix(2,4,A,2,3);`

(D2) $\begin{pmatrix} a & d & e \\ k & n & p \\ u & x & y \end{pmatrix}$

7.3. **成分の操作について.** ここまでの知識から、与えられた行列 A の第 (i, j) 成分 a_{ij} を取り出すのは簡単で、A[i][j]; 、A[i,j]; を使えばいい。また、**row** と **col** を組み合わせて使い col(row(A, i), j); とすれば、a_{ij} をただ一つの成分とする $(1, 1)$ 型行列 (a_{ij}) が得られる。成分だけを取り出すには col(row(A, i), j)[1][1]; あるいは col(row(A, i), j)[1,1]; とすればよい。

例 48. `(C1) A : matrix([a, b, c, d], [e, f, g, h], [i, j, k, l]);`

(D1) $\begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{pmatrix}$

(C2) `A[2][3];`

(D2) g

(C3) `A[2,3];`

(D3) g

(C4) `col(row(A, 2), 3);`

(D4) $\begin{pmatrix} g \end{pmatrix}$

(C5) `col(row(A, 2), 3)[1][1];`

(D5) g

(C6) `col(row(A, 2), 3)[1,1];`

(D6) g

逆に、行列 A の第 (i, j) 成分 a_{ij} に値を設定するには、代入文 A[i][j] : a; 、A[i,j] : a; などを使えばいい。これらの方法では A そのものが変化してしまう。A そのものを変えたくないときは、A のコピーに代入文を使うか、あるいは **matreplace** を使って、D : matrix([a]); かつ matreplace(A, D, i, i, j, j); とすればよい。

例 49. (C1) `A : matrix([a, b, c, d], [e, f, g, h], [i, j, k, l]);`

(D1) $\begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{pmatrix}$

(C2) `A[2][3] : -3;`

(D2) -3

(C3) `A;`

(D3) $\begin{pmatrix} a & b & c & d \\ e & f & -3 & h \\ i & j & k & l \end{pmatrix}$

(C4) `B : A;`

(D4) $\begin{pmatrix} a & b & c & d \\ e & f & -3 & h \\ i & j & k & l \end{pmatrix}$

(C5) `B[2][3] : x;`

(D5) x

(C6) `B;`

(D6) $\begin{pmatrix} a & b & c & d \\ e & f & x & h \\ i & j & k & l \end{pmatrix}$

(C7) `A;`

(D7) $\begin{pmatrix} a & b & c & d \\ e & f & x & h \\ i & j & k & l \end{pmatrix}$

(C8) `C : copymatrix(A);`

(D8) $\begin{pmatrix} a & b & c & d \\ e & f & x & h \\ i & j & k & l \end{pmatrix}$
(C9) `C[2][3] : g;`
(D9) g
(C10) `C;`
(D10) $\begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{pmatrix}$
(C11) `A;`
(D11) $\begin{pmatrix} a & b & c & d \\ e & f & x & h \\ i & j & k & l \end{pmatrix}$
(C12) `load("./matblock.mac");`
(D12) ./matblock.mac
(C13) `D : matrix([g]);`
(D13) $\begin{pmatrix} g \end{pmatrix}$
(C14) `matreplace(A, D, 2, 2, 3, 3);`
(D14) $\begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{pmatrix}$
(C15) `A;`
(D15) $\begin{pmatrix} a & b & c & d \\ e & f & x & h \\ i & j & k & l \end{pmatrix}$

注意 6. 上の例の (C4) から (D7) を見よ。手続き型言語の代入文とは違って、B : A; は B が A と同じ値を持った A のコピーであることを意味しない。B の値が A である、あるいは A に B という別名をつけるという意味である。したがって、B の値をかえると A の値も変化する。コピーをつくるには、コマンド **copymatrix** を使う必要がある。

行列の特定の成分に値を設定するもう一つの方法は、コマンド **setelmx** を使うことである。行列 M に対して、setelmx(a, i, j, M); とタイプすると、M の第 (i, j) 成分の値が a に設定される。このコマンドは M を変化させる。返り値は M 自身である。

例 50. (C1) `M : matrix([a,b,c,d],[e,f,g,h],[i,j,k,l]);`
(D1) $\begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{pmatrix}$
(C2) `setelmx(x, 2, 3, M);`
(D2) $\begin{pmatrix} a & b & c & d \\ e & f & x & h \\ i & j & k & l \end{pmatrix}$

8. 線形写像と基底の変換

8.1. **線形写像、線形変換.** n 個の成分がスカラー体 $\mathbb{K}$ に属するような列ベクトル全体の集合を $\mathbb{K}^n$ と書く。$\mathbb{K}^n$ の各ベクトルに、$\mathbb{K}^n$ のベクトルを１つづつ対応させる対応規則が与えられたとしよう。数学の言葉を使うと、集合 $\mathbb{K}^n$ から $\mathbb{K}^m$ への一つの写像が与えられたとするのである。この対応規則で、$\mathbf{x} \in \mathbb{K}^n$ が $\mathbf{y} \in \mathbb{K}^m$ に対応しているとしよう。成分を使って、

$$\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}, \qquad \mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}$$

と表すとき、$\mathbf{y}$ の各成分が $\mathbf{x}$ の成分たちの同次一次式になっているとき、すなわち対応規則が

$$\begin{cases} y_1 = a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ y_2 = a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \qquad \cdots \\ y_m = a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \end{cases}$$

の形で書けていて、しかもこの形が特定の $\mathbf{x}, \mathbf{y}$ によらないとき、この対応を以前は**一次写像**と呼んでいた。いま、写像に f という名前をつけて、$\mathbf{y} = f(\mathbf{x})$ という記法を使うことにしよう。行列 $A \in \mathrm{Mat}(m, n; \mathbb{K})$($\mathbb{K}$ 上の $m \times n$ 行列全体の集合) を

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

により定めると、上の事は

$$\mathbf{y} = f(\mathbf{x}) = A\mathbf{x}$$

と表現できる。行列 A は一次写像に f によって一意的に定まり、逆に行列 $A \in \mathrm{Mat}(m, n; \mathbb{K})$ を任意に与えれば、上の式によって $\mathbb{K}^n$ から $\mathbb{K}^m$ への一次写像 f が一つ定義されたことになる。行列 A を一次写像 f の**表現行列**と言う。

上の一次写像の定式化は非常に強力なものなのだが、実際の問題に適用するには不十分であることがすぐにわかる。ある問題のモデルが $\mathbb{K}^n$ から $\mathbb{K}^m$ への一次写像としてとらえられるとしよう。すなわち、観測対象の集合として X と Y というものがあり、考えている問題が X から Y への写像 f によって記述されるとする。ここで、X は $\mathbb{K}^n$ に、Y は $\mathbb{K}^m$ にみたてられ、f は一次写像と考えられる。ここで問題になるのは、X あるいは Y を $\mathbb{K}^n$ あるいは $\mathbb{K}^m$ に同一視する方法が一つとは限らないということである。同一視の方法がちがえば、当然 f の表現行列は違うものになるが、現象を記述している f そのものは本質的に同じものでなくてはならない。必要なことは、個々の同一視を離れた視点と、２つの特別な同一視が与えられたとき、それらの間の関係を記述する方法である。そこで、数学の持つ抽象の力が発揮される。

まず $\mathbb{K}^n$ たちの持っていた性質を抽象して行くことにしよう。集合 V が $\mathbb{K}$ 上の**ベクトル空間**であると言うのは、V 上に２つの演算 **加法**" $+$ " と **スカラー倍** "$\cdot$" が定義されていて、以下の性質が成立することを言う。

(1) $\mathbf{v} + \mathbf{w} = \mathbf{w} + \mathbf{v}, \quad \forall \mathbf{v}, \forall \mathbf{w} \in V$
(2) $(\mathbf{u} + \mathbf{v}) + \mathbf{w} = \mathbf{u} + (\mathbf{v} + \mathbf{w}), \quad \forall \mathbf{u}, \forall \mathbf{v}, \forall \mathbf{w} \in V$

(3) $\exists_1 \mathbf{0}\colon \mathbf{v}+\mathbf{0}=\mathbf{0}+\mathbf{v}=\mathbf{v}, \quad \forall \mathbf{v}\in V$
(4) $\forall \mathbf{v}\in V, \exists_1 -\mathbf{v}\in V: \mathbf{v}+(-\mathbf{v})=(-\mathbf{v})+\mathbf{v}=\mathbf{0}$
(5) $\alpha\cdot(\beta\cdot\mathbf{v})=(\alpha\beta)\cdot\mathbf{v}, \quad \forall\alpha,\forall\beta\in\mathbb{K}, \quad \forall\mathbf{v}\in V$
(6) $(\alpha+\beta)\cdot\mathbf{v}=\alpha\cdot\mathbf{v}+\beta\cdot\mathbf{v}, \quad \forall\alpha,\forall\beta\in\mathbb{K}, \quad \forall\mathbf{v}\in V$
(7) $\alpha\cdot(\mathbf{v}+\mathbf{w})=\alpha\cdot\mathbf{v}+\alpha\cdot\mathbf{w}, \quad \forall\alpha\in\mathbb{K}, \quad \forall\mathbf{v},\forall\mathbf{w}\in V$
(8) $1\cdot\mathbf{v}=\mathbf{v}, \quad \forall\mathbf{v}\in V$
(9) $0\cdot\mathbf{v}=\mathbf{0}, \quad \forall\mathbf{v}\in V$

ここで、(3) の $\mathbf{0}$ を零ベクトル、(4) の $-\mathbf{v}$ を $\mathbf{v}$ の逆ベクトルと言う。$(-1)\mathbf{v}=-\mathbf{v}$ であることはすぐわかる。ベクトル空間の任意の要素をベクトルと言う。以下、"·" は省略することにしよう。

抽象的なベクトル空間だけでは色々な計算ができないので、これを列ベクトル全体の集合と同一視するための言葉を与えよう。ベクトル空間 V から有限個 (n 個) の順序のついたベクトルの組 $\mathfrak{A}=(\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_n)$ を選ぶことができて、これに対し任意の $\mathbf{v}\in V$ に対して

$$\exists_1 \mathbf{v}_{\mathfrak{A}} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix} \in \mathbb{K}^n \quad : \quad \mathbf{v}=v_1\mathbf{a}_1+v_2\mathbf{a}_2+\cdots+v_n\mathbf{a}_n$$

となるとき、すなわち行列の積を使った記法では、

$$\mathbf{v}=(\mathbf{a}_1,\ldots,\mathbf{a}_n)\begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix}=\mathfrak{A}\mathbf{v}_{\mathfrak{A}}$$

と書けるとき、ベクトル空間 V は n 次元であると言い、$\mathfrak{A}$ を V の**基底**と言う。また、$\mathbf{v}_{\mathfrak{A}}$ を $\mathbf{a}$ の $\mathfrak{A}$ に関する**成分**と言う。n 次元ベクトル空間の基底の定め方は無限の可能性があるが、各基底を構成するベクトルの個数は常に n 個であることが証明できる。このあたりの詳しいことは線形代数学の教科書に譲ろう。2 通りの基底があれば、各ベクトルを成分で表す方法も 2 通りあることになるが、それらの間に成り立つ関係については後の節で解説する。

特に、 $\mathbb{K}^n$ 自身、n 次元ベクトル空間である。それは、標準基底 $\mathfrak{E}=(\mathbf{e}_1,\mathbf{e}_2,\ldots,\mathbf{e}_n)$ により、任意の列ベクトル $\mathbf{v}\in\mathbb{K}^n$ が

$$\mathbf{v}=\begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix}=v_1\mathbf{e}_1+v_2\mathbf{e}_2+\cdots+v_n\mathbf{e}_n=\mathfrak{E}\mathbf{v}$$

と書けることからすぐわかる。さらに、$\mathbb{K}^n$ においては、

$$\mathbf{v}=\mathbf{v}_{\mathfrak{E}}$$

が成り立っている。

すべてのベクトル空間が有限次元であるというわけではない、有限次元でないベクトル空間は無限次元であるというが、この文書では扱わない。

つぎに、一次写像を抽象化しよう。抽象ベクトル空間の間の写像を考えるので、まずは座標によらない定義が必要になる。そこで、一次写像の持っていた幾何学的、あるいは抽象代数的側面に注目しよう。幾何学的に言うと一次写像は (複素) 直線を (複素) 直線ある

いは原点に移す写像である、これを代数の言葉で表現すると次のようになる。V を $\mathbb{K}$ 上の n 次元ベクトル空間、 W を $\mathbb{K}$ 上の m 次元ベクトル空間とする。f は V から W への写像であり、性質

$$f(\alpha\mathbf{v}+\beta\mathbf{w}) = \alpha f(\mathbf{v}) + \beta f(\mathbf{w}), \quad \forall\alpha, \forall\beta \in \mathbb{K}, \quad \forall\mathbf{v}, \forall\mathbf{w} \in V$$

を満たす。この性質を満たすような V から W への写像を **線形写像** と言う。線形写像という概念が一次写像を抽象化したものであることは、V と W に基底の組を入れてみれば容易に理解することができる。今、V の基底を $\mathfrak{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_n)$、$W$ の基底を $\mathfrak{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_m)$ としよう。線形写像 f は $(f(\mathbf{a}_1), \ldots, f(\mathbf{a}_n)) \in W^n$ で完全に決定される。なぜならば、任意の $\mathbf{v} \in V$ は $\mathfrak{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_n)$ たちによって、

$$\mathbf{v} = v_1\mathbf{a}_1 + \cdots v_n\mathbf{a}_n = \mathfrak{A}\mathbf{v}_{\mathfrak{A}}$$

と書け、f が線形写像であるから、

$$f(\mathbf{v}) = v_1 f(\mathbf{a}_1) + \cdots + v_n f(\mathbf{a}_n)$$

となるからである。記号の簡単のため、$f(\mathfrak{A}) = (f(\mathbf{a}_1), \ldots, f(\mathbf{a}_n))$ と書くことにすると、

$$f(\mathbf{v}) = f(\mathfrak{A})\mathbf{v}_{\mathfrak{A}}$$

と書ける。一方、各 $f(\mathbf{a}_i)$ たちは W のベクトルなので、$\mathfrak{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_m)$ によって、

$$\begin{cases} f(\mathbf{a}_1) = a_{11}\mathbf{a}_1 + a_{21}\mathbf{a}_2 \cdots + a_{m1}\mathbf{b}_m \\ f(\mathbf{a}_2) = a_{12}\mathbf{a}_1 + a_{22}\mathbf{a}_2 \cdots + a_{m2}\mathbf{b}_m \\ \qquad \cdots \\ f(\mathbf{a}_n) = a_{1n}\mathbf{a}_1 + a_{2n}\mathbf{a}_2 \cdots + a_{mn}\mathbf{b}_m \end{cases}$$

と書くことができる。ここで、行列 $A \in \mathrm{Mat}(m, n; \mathbb{K})$ を

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

と定める。この行列 A は写像 f と基底の組 $(\mathfrak{A}, \mathfrak{B})$ によって決まる。上のことを行列の積で表現すると

$$f(\mathfrak{A}) = \mathfrak{B}A$$

となる。したがって

$$f(\mathbf{v}) = \mathfrak{B}A\mathbf{v}_{\mathfrak{A}}$$

を得るが、この最後の式は $f(\mathbf{v})$ の $\mathfrak{B}$ に関する成分を $\mathbf{w}_{\mathfrak{B}}$ と書くことにすると、($f(v) = \mathfrak{B}\mathbf{w}_{\mathfrak{B}}$ より、)

$$\mathbf{w}_{\mathfrak{B}} = A\mathbf{v}_{\mathfrak{A}}$$

を意味することがわかる。このことは、V と $\mathbb{K}^n$ を、 W と $\mathbb{K}^m$ を基底対 $(\mathfrak{A}, \mathfrak{B})$ によって同一視したとき、線形写像 f が行列 A で定まる一次写像になっていることを意味する。つまり、線形写像はその具体的表現では一次写像になるのである。さて、ここで出てきた行列 A を線形写像 f の基底対 $(\mathfrak{A}, \mathfrak{B})$ に関する**表現行列**と言う。基底を取り替えたときに表現行列がどうなるかについては後の節で扱う。線形写像 f の性質は表現行列の基底の取り方に依らない性質の中に見出すことができる。

8.2. **核、像、次元.** ベクトル空間 V の部分集合 W がそれ自身ベクトル空間であるとき、W を V の**部分ベクトル空間** (以下、単に部分空間と言うことにする) と言う。これは、性質

$$\mathbf{v}, \mathbf{w} \in W \Rightarrow \alpha\mathbf{v} + \beta\mathbf{w} \in W, \quad \forall\alpha, \forall\beta \in \mathbb{K}$$

が満たされることと言ってもよい。(注意: 一般には $\alpha\mathbf{v} + \beta\mathbf{w} \in V$ である。) 言葉で言えば、加法とスカラー倍に対して閉じた部分集合を部分空間と言うのである。V 自身と、$\{\mathbf{0}\}$ は明らかに V の部分空間であるが、これらを自明な部分空間と言う。一般に、V の任意の部分集合 S に対して、S を含むような V の部分空間のうちで、包含関係について最小のものが存在する。それを、$\langle S\rangle$ と書いて、S で**生成される** V の部分空間という。集合論の記法を用いれば、

$$\langle S\rangle = \{\mathbf{v} \mid \mathbf{v} = \alpha_1\mathbf{v}_1 + \cdots + \alpha_k\mathbf{v}_k, \quad \alpha_1, \ldots, \alpha_k \in \mathbb{K}, \quad \mathbf{v}_1, \ldots, \mathbf{v}_k \in S, \quad k \in \mathbb{N}\}$$

である。つまり、S の有限個の要素たちの**一次結合**で書ける V の要素全体の集合を $\langle S\rangle$ と書くのである。S が最初から部分空間ならば $S = \langle S\rangle$ である。ベクトル空間 V が有限次元で基底 $\mathfrak{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_n)$ を持つならば、$V = \langle\{\mathbf{a}_1, \ldots, \mathbf{a}_n\}\rangle$ である。このときベクトル空間の次元は n であるのであるが、このことを記号で、$\dim V = n$ あるいは $\dim_{\mathbb{K}} V = n$ と書く。有限次元ベクトル空間 V の部分空間 W はやはり有限次元であるが、このとき $\dim W \leq \dim V$ が成り立つ。等号成立は $V = W$ を意味する。

この節で特に関心のあるのは、線形写像にともなう部分ベクトル空間たちである。今、V と W をともに有限次元ベクトル空間で、 $\dim V = n$、$\dim W = m$ とする。また、$f\colon V \to W$ を V から W への線形写像としよう。このとき、V の部分集合

$$\operatorname{Ker} f := \{\mathbf{v} \in V \mid f(\mathbf{v}) = \mathbf{0}\}$$

を f の**核** (Kernel) と言い、W の部分集合

$$\operatorname{Im} f := \{f(\mathbf{v}) \mid \mathbf{v} \in V\}$$

を f の**像** (Image) と言う。核 $\operatorname{Im} f$、像 $\operatorname{Im} f$ は f の線形性からそれぞれ V、 W の部分空間になる。実際、

$$\begin{aligned}\mathbf{v}, \mathbf{w} \in \operatorname{Ker} f &\Rightarrow f(\alpha\mathbf{v} + \beta\mathbf{w}) = \alpha f(\mathbf{v}) + \beta f(\mathbf{w}) = \mathbf{0}\\ &\Rightarrow \alpha\mathbf{v} + \beta\mathbf{w} \in \operatorname{Ker} f\end{aligned}$$

及び、

$$\alpha f(\mathbf{v}) + \beta f(\mathbf{w}) = f(\alpha\mathbf{v} + \beta\mathbf{w}) \in \operatorname{Im} f$$

が成り立つ。前小節で、線形写像を基底対のもとで列ベクトルの空間の間の一次写像と同一視することを説明したが、その状況の下で核と像はそれぞれ $\mathbb{K}^n$ と $\mathbb{K}^m$ の部分空間と同一視される。これらを具体的に見ていこう。$\mathfrak{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_n)$ を V の基底、$\mathfrak{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_m)$ を W の基底とし、$A \in \operatorname{Mat}(m, n; \mathbb{K})$ を f の基底対 $(\mathfrak{A}, \mathfrak{B})$ に関する表現行列としよう。

$$\mathbf{v} = \mathfrak{A}\mathbf{v}_{\mathfrak{A}}, \quad \mathbf{v}_{\mathfrak{A}} \in \mathbb{K}^n$$

が $v \in \operatorname{Ker} f$ であるということは、

$$A\mathbf{v}_{\mathfrak{A}} = \mathbf{0}$$

を意味する。すなわち、$\mathbf{v}$ の $\mathfrak{A}$ に関する成分 $\mathbf{v}_{\mathfrak{A}}$ は同次連立一次方程式 $A\mathbf{x} = \mathbf{0}$ の解である。この逆も真である。つまり、$\operatorname{Ker} f$ はその具体的表現としては、表現行列を係数行列とする同次連立一次方程式の解全体の空間 (以下、解空間という) として出現するのである。そこで、5 節の連立一次方程式の話しを全面的に適用することができる。

例 51. $V = \mathbb{K}^3, W = \mathbb{K}^2$ とする。どちらも標準基底で考えよう。線形写像 $f\colon V \to W$ が

$$f\begin{pmatrix}1\\0\\0\end{pmatrix} = \begin{pmatrix}2\\3\end{pmatrix}, \quad f\begin{pmatrix}0\\1\\0\end{pmatrix} = \begin{pmatrix}0\\-1\end{pmatrix}, \quad f\begin{pmatrix}0\\0\\1\end{pmatrix} = \begin{pmatrix}2\\-2\end{pmatrix}$$

で与えられるものとする。このとき、f の表現行列 A は

$$A = \begin{pmatrix}2 & 0 & 2\\3 & -1 & -2\end{pmatrix}$$

である。この例では

$$\operatorname{Ker} f = \{\mathbf{v} \in \mathbb{K}^3 | A\mathbf{v} = \mathbf{0}\}$$

である。**maxima** を使って $\operatorname{Ker} f$ を調べよう。

(C1) `A : matrix([2, 0, 2],[3, -1, -2]);`

(D1) $\begin{pmatrix}2 & 0 & 2\\3 & -1 & -2\end{pmatrix}$

(C2) `v : matrix([s],[t],[u]);`

(D2) $\begin{pmatrix}s\\t\\u\end{pmatrix}$

(C3) `o : matrix([0],[0]);`

(D3) $\begin{pmatrix}0\\0\end{pmatrix}$

(C4) `eq1 : (A . v)[1][1] = o[1][1];`

(D4) $2u + 2s = 0$

(C5) `eq2 : (A . v)[2][1] = o[2][1];`

(D5) $-2u - t + 3s = 0$

(C6) `solve([eq1, eq2],[s, t, u]);`

(D6) $[[s = -\%\mathrm{R1}, t = -5\%\mathrm{R1}, u = \%\mathrm{R1}]]$

この結果から、

$$\operatorname{Ker} f = \left\{ \begin{pmatrix}-r\\-5r\\r\end{pmatrix} \middle| \, r \in \mathbb{K} \right\} = \left\langle \left\{ \begin{pmatrix}-1\\-5\\1\end{pmatrix} \right\} \right\rangle$$

となる。したがって、$\dim \operatorname{Ker} f = 1$ であることもわかる。 □

像 $\operatorname{Im} f$ については、まず

$$\operatorname{Im} f = \langle \{ f(\mathbf{a}_1), \ldots, f(\mathbf{a}_n) \} \rangle$$

であることに注意する。各 $f(\mathbf{a}_i)$ の基底 $\mathfrak{B}$ に関する成分を $\mathbf{w}_{\mathfrak{B},i}$ と書くことにすれば、$\operatorname{Im} f$ と同一視されるべき $\mathbb{K}^m$ の部分空間は

$$\langle \{ \mathbf{w}_{\mathfrak{B},1}, \ldots, \mathbf{w}_{\mathfrak{B},n} \} \rangle$$

で与えられることは明らかであろう。一方、

$$\mathbf{a}_i = \mathfrak{A}\mathbf{a}_{\mathbf{A},i}, \quad \mathbf{a}_{\mathfrak{A},i} \in \mathbb{K}^n, \quad i \in \{1, \ldots, n\}$$

とするとき、

$$\mathbf{a}_{\mathfrak{A},i} = \mathbf{e}_i, \quad i \in \{1, \ldots, n\}$$

であり、

$$\mathbf{w}_{\mathfrak{B},i} = A\mathbf{a}_{\mathfrak{A},i} = A\mathbf{e}_i, \quad i \in \{1, \dots, n\}$$

となる。ここで、

$$\mathbf{f}_i = A\mathbf{e}_i$$

は行列 A の第 i 列を表すから、

$$A = (\mathbf{f}_1, \dots, \mathbf{f}_n) = (\mathbf{w}_{\mathfrak{B},1}, \dots, \mathbf{w}_{\mathfrak{B},n})$$

が成り立つ。したがって $\mathrm{Im}\, f$ の具体的実現は

$$\langle\{\mathbf{f}_1, \dots, \mathbf{f}_n\}\rangle$$

となる。つまり、$\mathrm{Im}\, f$ の情報はすべて表現行列 A に含まれているわけである。線形代数学によれば、行列としての階数 $\mathrm{rank}\, A$ は A の列ベクトルたちの中で一次独立なものの最大個数に等しいことが知られているが、それは正に上の列ベクトルたちで生成される部分空間の次元に等しい。すなわち、

$$\dim\ \mathrm{Im}\, f = \mathrm{rank}\, A$$

である。この数には重要な情報があるので、線形写像 f の 階数と言って $\mathrm{rank}\, f$ で表す。

$$\mathrm{rank}\, f := \dim\ \mathrm{Im}\, f = \mathrm{rank}\, A.$$

例 52. 今度の例では少し抽象的にしてみよう。V は 3 次元ベクトル空間で基底 $\mathfrak{A} = (\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3)$ が与えられているものとする。W は 2 次元ベクトル空間で基底 $\mathfrak{B} = (\mathbf{b}_1, \mathbf{b}_2)$ があてられているものとする。f は V から W への線形写像で

$$\begin{cases} f(\mathbf{a}_1) = 3\mathbf{b}_1 + 2\mathbf{b}_2 \\ f(\mathbf{a}_2) = -2\mathbf{b}_1 + \mathbf{b}_2 \\ f(\mathbf{a}_3) = -6\mathbf{b}_1 - 4\mathbf{b}_2 \end{cases}$$

であるとしよう。このとき、f の基底対 $(\mathfrak{A}, \mathfrak{B})$ に関する表現行列 A は

$$A = \begin{pmatrix} 3 & -2 & -6 \\ 2 & 1 & -4 \end{pmatrix}$$

で与えられる。f の 核、像を調べよう。

(C1) `A : matrix([3, -2, 6], [2, 1, -4]);`

(D1) $\begin{pmatrix} 3 & -2 & 6 \\ 2 & 1 & -4 \end{pmatrix}$

(C2) `eq1 : 3*s - 2*t + 6*v = 0;`

(D2) $6v - 2t + 3s = 0$

(C3) `eq2 : 2*s + t - 4*v = 0;`

(D3) $-4v + t + 2s = 0$

(C4) `solve([eq1, eq2], [s, t, v]);`

(D4) $[[s = \frac{2\%R1}{7}, t = \frac{24\%R1}{7}, v = \%R1]]$

(C5) `rank(A);`

(D5) 2

これから分かるのは

$$\mathrm{Ker}\, f = \left\{ \frac{2r}{7}\mathbf{a}_1 + \frac{24r}{7}\mathbf{a}_2 + r\mathbf{a}_3 \;\middle|\; r \in \mathbb{K} \right\} = \left\langle \left\{ \frac{2}{7}\mathbf{a}_1 + \frac{24r}{7}\mathbf{a}_2 + \mathbf{a}_3 \right\} \right\rangle$$

であり、$\dim \mathrm{Ker}\, f = 1$。また、$\dim \mathrm{Im}\, f = \mathrm{rank}\, f = \mathrm{rank}\, A = 2$、したがって、

$$\mathrm{Im}\, f = W$$

となることである。 □

勝手な $\mathbf{w} \in W$ が $\mathrm{Im}\, f$ に属すかどうかというのは、次のような意味がある。$\mathbf{w} \in \mathrm{Im}\, f$ ならば、ある $\mathbf{v} \in V$ があって、$\mathbf{w} = f(\mathbf{v})$ となることである。成分を使って表すと、$\mathbf{w}_{\mathbf{B}} = A\mathbf{v}_{\mathfrak{A}}$ となる。これは、非同次連立一次方程式

$$A\mathbf{x} = \mathbf{w}_{\mathbf{B}}$$

が解をもつと言うことと同値になる。一方、$\mathbf{w} \in \mathrm{Im}\, f$ ということは、上述より $\mathbf{w}$ が $f(\mathbf{a}_1), \ldots, f(\mathbf{a}_n)$ たちの一次結合で書けるということと同値である、すなわち、$\mathbf{w}_{\mathfrak{B}}$ が $\mathbf{f}_1, \ldots, \mathbf{f}_n$ の一次結合で書けることを意味する。これは、連立一次方程式 $A\mathbf{x} = \mathbf{w}_{\mathbf{B}}$ の係数行列 A と 拡大係数行列 $(A, \mathbf{w}_{\mathfrak{B}})$ の階数が同じであることと同値である。

$$\mathbf{w} \in \mathrm{Im}\, f \Leftrightarrow \mathrm{rank}\, A = \mathrm{rank}(A, \mathbf{w}_{\mathfrak{B}}).$$

例 53. $V = \mathbb{K}^3$、$W = \mathbb{K}^3$ としよう、簡単のためどちらも標準基底を入れて考えることにする。線形写像 $f\colon V \to W$ が行列

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

を使って、$f(\mathbf{v}) = A\mathbf{v}, \quad \forall \mathbf{v} \in V$ により定義されているとしよう。2つのベクトル

$$\mathbf{a} = \begin{pmatrix} -2 \\ 1 \\ 4 \end{pmatrix}, \qquad \mathbf{b} = \begin{pmatrix} -2 \\ 1 \\ 3 \end{pmatrix} \in W$$

について、それぞれ $\mathrm{Im}\, f$ に属すかどうかを調べよう。

(C1) `A : matrix([1, 2, 3], [4, 5, 6], [7, 8, 9]);`

(D1) $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$

(C2) `a : transpose(matrix([-2, 1, 4]));`

(D2) $\begin{pmatrix} -2 \\ 1 \\ 4 \end{pmatrix}$

(C3) `b : transpose(matrix([-2, 1, 3]));`

(D3) $\begin{pmatrix} -2 \\ 1 \\ 3 \end{pmatrix}$

(C4) `rank(A);`

(D4) 2

(C5) `rank(addcol(A,a));`

(D5) 2

(C6) `rank(addcol(A,b));`

(D6) 3

(C7) `addcol(A,a);`

(D7) $\begin{pmatrix} 1 & 2 & 3 & -2 \\ 4 & 5 & 6 & 1 \\ 7 & 8 & 9 & 4 \end{pmatrix}$

```
(C8) addcol(A,b);
```

(D8) $\begin{pmatrix} 1 & 2 & 3 & -2 \\ 4 & 5 & 6 & 1 \\ 7 & 8 & 9 & 3 \end{pmatrix}$

これによると、$\operatorname{rank} A = \operatorname{rank}(A, \mathbf{a})$ より $\mathbf{a} \in \operatorname{Im} f$ であり、$\operatorname{rank} A \neq \operatorname{rank}(A, \mathbf{b})$ より $\mathbf{b} \notin \operatorname{Im} f$ であることが分かる。 □

線形写像 $f\colon V \to W$ の研究で最も重要な公式は次の次元公式である。

$$\dim\ V = \dim\ \operatorname{Ker} f + \dim\ \operatorname{Im} f.$$

上述と合わせて $\operatorname{Ker} f$ の次元を

$$\dim\ \operatorname{Ker} f = \dim V - \operatorname{rank} f = \dim\ V - \operatorname{rank} A$$

で計算できる。$\operatorname{Ker} f$ は具体的表現においては 同次連立一次方程式の $A\mathbf{x} = \mathbf{0}$ の解空間と同一視されたから、これは解空間の次元公式でもある。

例 54. ここでも簡単のため、$V = \mathbb{K}^3, W = \mathbb{K}^3$ とし、どちらにも標準基底を入れて考えよう。線形写像 $f\colon V \to W$ は、行列

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

により、$f(\mathbf{v}) = A\mathbf{v}, \quad \forall \mathbf{v} \in V$ で定義されているものとする。

```
(C1) A : matrix([1, 2, 3], [4, 5, 6], [7, 8, 9]);
```

(D1) $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$

```
(C2) v : transpose(matrix([s, t, u]));
```

(D2) $\begin{pmatrix} s \\ t \\ u \end{pmatrix}$

```
(C3) AV : A . v;
```

(D3) $\begin{pmatrix} 3u + 2t + s \\ 6u + 5t + 4s \\ 9u + 8t + 7s \end{pmatrix}$

```
(C4) eq1 : AV[1][1];
```

(D4) $3u + 2t + s$

```
(C5) eq2 : AV[2][1];
```

(D5) $6u + 5t + 4s$

```
(C6) eq3 : AV[3][1];
```

(D6) $9u + 8t + 7s$

```
(C7) solve([eq1, eq2, eq3], [s, t, u]);
Dependent equations eliminated:  (3)
```

(D7) $[[s = \%\mathrm{R}1, t = -2\%\mathrm{R}1, u = \%\mathrm{R}1]]$

```
(C8)  rank(A);
(D8) 2
(C9)  A;
```

(D9) $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$

```
(C10)  load("./em.mac");
(D10) ./em.mac
(C11)  A1 : A . R(-2, 1, 2, 3);
```

(D11) $\begin{pmatrix} 1 & 0 & 3 \\ 4 & -3 & 6 \\ 7 & -6 & 9 \end{pmatrix}$

```
(C12)  A2 : A1 . R(-3, 1, 3, 3);
```

(D12) $\begin{pmatrix} 1 & 0 & 0 \\ 4 & -3 & -6 \\ 7 & -6 & -12 \end{pmatrix}$

```
(C13)  A3 : A2 . R(-2, 2, 3, 3);
```

(D13) $\begin{pmatrix} 1 & 0 & 0 \\ 4 & -3 & 0 \\ 7 & -6 & 0 \end{pmatrix}$

```
(C14)  A4 : A3 . Q((-1/3), 2, 3);
```

(D14) $\begin{pmatrix} 1 & 0 & 0 \\ 4 & 1 & 0 \\ 7 & 2 & 0 \end{pmatrix}$

この計算から、

$$\mathrm{Ker}\, f = \left\{ \begin{pmatrix} r \\ -2r \\ r \end{pmatrix} \,\middle|\, r \in \mathbb{R} \right\} = \left\langle \left\{ \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix} \right\} \right\rangle$$

であり、$\dim \mathrm{Ker}\, f = 1$ であることがわかる。さらに、$\dim \mathrm{Im}\, f = \mathrm{rank}\, f = \mathrm{rank}\, A = 2$ であり、

$$\mathrm{Im}\, f = \left\langle \left\{ \begin{pmatrix} 1 \\ 4 \\ 7 \end{pmatrix}, \begin{pmatrix} 2 \\ 5 \\ 8 \end{pmatrix}, \begin{pmatrix} 3 \\ 6 \\ 9 \end{pmatrix} \right\} \right\rangle = \left\langle \left\{ \begin{pmatrix} 1 \\ 4 \\ 7 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \\ 0 \end{pmatrix} \right\} \right\rangle$$

であることもわかる。$3 = 1 + 2$ より、次元公式

$$\dim V = \dim \mathrm{Ker}\, f + \dim \mathrm{Im}\, f$$

の成立がたしかめられた。 □

この節で見たように、線形写像の研究は具体的には、連立一次方程式の研究と同じである。

8.3. **基底と成分の変換.** 前小節までは、基底を固定して線形写像をその表現行列から研究した。この小節では、基底を取り替えてみて、ベクトルの成分や表現行列がどのように変わるのか、そして、それらが変化しても核や像の次元が変わることはないことを見ていく。V を n 次元のベクトル空間、W を m 次元のベクトル空間とする。また f を V から W への線形写像としよう。まず、V に 2 通りの基底 $\mathfrak{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_n)$ と $\mathfrak{A}' = (\mathbf{a}'_1, \ldots, \mathbf{a}'_n)$

が与えられたとしよう。基底 $\mathfrak{A}'$ を基準に考える。基底 $\mathfrak{A}$ を構成する各ベクトル $\mathbf{a}_i$ たちも V のベクトルであるから、$\mathbf{a}'_1, \ldots, \mathbf{a}'_n$ たちの一次結合で書ける。そこで、

$$\begin{cases} \mathbf{a}_1 = p_{11}\mathbf{a}'_1 + p_{21}\mathbf{a}'_2 + \cdots + p_{n1}\mathbf{a}'_n \\ \mathbf{a}_2 = p_{12}\mathbf{a}'_1 + p_{22}\mathbf{a}'_2 + \cdots + p_{n2}\mathbf{a}'_n \\ \qquad \cdots \\ \mathbf{a}_n = p_{1n}\mathbf{a}'_1 + p_{2n}\mathbf{a}'_2 + \cdots + p_{nn}\mathbf{a}'_n \end{cases}$$

としよう。ここで、行列 $P \in \mathrm{Mat}(n, n; \mathbb{K})$ を

$$P := \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{n1} & p_{n2} & \cdots & p_{nn} \end{pmatrix}$$

と置くと、上のことは

$$\mathfrak{A} = \mathfrak{A}'P$$

と行列の積で表現される。 この行列 P を基底の変換 $\mathfrak{A} \to \mathfrak{A}'$ の変換行列と言う。見方を変えると、これは恒等写像 $\iota\colon V \to V$ の基底対 $(\mathfrak{A}, \mathfrak{A}')$ に関する表現行列と言ってもよい。$\operatorname{rank}\iota = \operatorname{rank}P = n$ であるから、P は正則行列である。V の任意のベクトル $\mathbf{v}$ に対して、その 2 通りの基底についての成分の間の関係を調べよう。

$$\mathfrak{A}\mathbf{v}_{\mathfrak{A}} = \mathfrak{A}'\mathbf{v}_{\mathfrak{A}'} (= \mathbf{v})$$

より

$$\mathfrak{A}'P\mathbf{v}_{\mathbf{A}} = \mathfrak{A}'\mathbf{v}_{\mathfrak{A}'}$$

を得る。すなわち、

$$\mathbf{v}_{\mathfrak{A}'} = P\mathbf{v}_{\mathfrak{A}}$$

これが、基底の変換 $\mathfrak{A} \to \mathfrak{A}'$ にともなうベクトルの成分の変換則である。

$V = \mathbf{K}^n$ であるときには、基底を構成する各ベクトルも列ベクトルであるから、基底 $\mathfrak{A}$ と $\mathfrak{A}'$ はともに $n \times n$ 型の正則行列と見なすことができる。したがって、行列 P は、この場合

$$P = \mathfrak{A}'^{-1}\mathfrak{A}$$

により、計算することができる。

例 55. $V = \mathbb{R}^3$ とする。V の基底 $\mathfrak{A} = (\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_2)$ を

$$\mathbf{a}_1 = \begin{pmatrix} 2 \\ -2 \\ 1 \end{pmatrix}, \quad \mathbf{a}_2 = \begin{pmatrix} 3 \\ 0 \\ -1 \end{pmatrix}, \quad \mathbf{a}_1 = \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix}$$

とする。また、別の V の基底 $\mathfrak{A}' = (\mathbf{a}'_1, \mathbf{a}'_2, \mathbf{a}'_2)$ を

$$\mathbf{a}'_1 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \quad \mathbf{a}'_2 = \begin{pmatrix} 3 \\ 1 \\ 3 \end{pmatrix}, \quad \mathbf{a}'_1 = \begin{pmatrix} 1 \\ -2 \\ 4 \end{pmatrix}$$

としよう。このとき、行列として

$$\mathfrak{A} = \begin{pmatrix} 2 & 3 & 1 \\ -2 & 0 & 1 \\ 1 & -1 & 2 \end{pmatrix}, \quad \mathfrak{A}' = \begin{pmatrix} 1 & 3 & 1 \\ 1 & 1 & -2 \\ 0 & 3 & 4 \end{pmatrix}$$

であり、基底の変換 $\mathfrak{A} \to \mathfrak{A}'$ の変換行列 P は $P = (\mathfrak{A}')^{-1}\mathfrak{A}$ で与えられる。

(C1) `A1 : matrix([2, 3, 1], [-2, 0, 1], [1, -1, 2]);`

(D1) $\begin{pmatrix} 2 & 3 & 1 \\ -2 & 0 & 1 \\ 1 & -1 & 2 \end{pmatrix}$

(C2) `A2 : matrix([1, 3, 1], [1, 1, -2], [0, 3, 4]);`

(D2) $\begin{pmatrix} 1 & 3 & 1 \\ 1 & 1 & -2 \\ 0 & 3 & 4 \end{pmatrix}$

(C3) `P : A2^^(-1) . A1;`

(D3) $\begin{pmatrix} 31 & 37 & -13 \\ -13 & -15 & 6 \\ 10 & 11 & -4 \end{pmatrix}$

(C4) `A1 - (A2 . P);`

(D4) $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$

上の計算から

$$P = \begin{pmatrix} 31 & 37 & -13 \\ -13 & -15 & 6 \\ 10 & 11 & -4 \end{pmatrix}$$

であることがわかる。ここで、

$$\mathbf{v} = \begin{pmatrix} -1 \\ 3 \\ 6 \end{pmatrix}$$

の基底 $\mathfrak{A}$ 及び $\mathfrak{A}'$ それぞれについての成分を調べて、基底の変換にともなう成分の変換則の成立をチェックしておこう。$\mathbf{v}$ の成分 $\mathbf{v}_{\mathfrak{A}}$ と $\mathbf{v}_{\mathfrak{A}'}$ はそれぞれ、

$$\mathbf{v}_{\mathfrak{A}} = \mathfrak{A}^{-1}\mathbf{v}, \quad \mathbf{v}_{\mathfrak{A}'} = \mathfrak{A}'^{-1}\mathbf{v}$$

で計算できる。

(C5) `v : transpose(matrix([-1, 3, 6]));`

(D5) $\begin{pmatrix} -1 \\ 3 \\ 6 \end{pmatrix}$

(C6) `v1 : A1^^(-1) . v;`

(D6) $\begin{pmatrix} -\frac{4}{19} \\ -\frac{20}{19} \\ \frac{49}{19} \end{pmatrix}$

(C7) `v2 : A2^^(-1) . v;`

(D7) $\begin{pmatrix} -79 \\ 34 \\ -24 \end{pmatrix}$

```
(C8) P . v1;
```

(D8) $\begin{pmatrix} -79 \\ 34 \\ -24 \end{pmatrix}$

```
(C9) v2 - P . v1;
```

(D9) $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$

上の計算より、

$$\mathbf{v}_{\mathfrak{A}} = \begin{pmatrix} -\frac{4}{19} \\ -\frac{20}{19} \\ \frac{49}{19} \end{pmatrix}, \qquad \mathbf{v}_{\mathfrak{A}'} = \begin{pmatrix} -79 \\ 34 \\ -24 \end{pmatrix}$$

であり、

$$\mathbf{v}_{\mathfrak{A}'} = P\mathbf{v}_{\mathfrak{A}}$$

が成立していることが確認された。 □

W にも 2 通りの基底 $\mathfrak{B}$ と $\mathfrak{B}'$ をいれてみよう。基底の変換 $\mathfrak{B} \to \mathfrak{B}'$ の変換行列を $Q \in \mathrm{Mat}(n, n; \mathbf{K})$ とする。もちろん、

$$\mathfrak{B} = \mathfrak{B}'Q$$

であり、Q は正則行列である。線形写像 $f\colon V \to W$ の 2 通りの基底対 $(\mathfrak{A}, \mathfrak{B})$、$(\mathfrak{A}', \mathfrak{B}')$ についての表現行列を、それぞれ、A、A' としよう。このとき、

$$f(\mathfrak{A}) = \mathfrak{B}A, \quad f(\mathfrak{A}') = \mathfrak{B}'A'$$

である。f の線形性を使うと、

$$\begin{aligned} \mathfrak{B}'QA &= \mathfrak{B}A \\ &= f(\mathfrak{A}) \\ &= f(\mathfrak{A}'P) \\ &= f(\mathfrak{A}')P \\ &= \mathfrak{B}'A'P \end{aligned}$$

すなわち、

$$QA = A'P$$

あるいは、

$$A = Q^{-1}A'P, \qquad A' = QAP^{-1}$$

を得る。これが、表現行列の変換則である。前に正方行列のところで言及したように、行列の階数は正則行列を掛けても変わらないことから、

$$\mathrm{rank}\, A = \mathrm{rank}\, A'$$

が結論される。これにより、核や像の次元は、どんな基底対で計算しても同じになることが保証される。

例 56. 例 55 の状況に、さらに、$W = \mathbb{K}^2$ を考える。W にも 2 つの基底 $\mathfrak{B} = (\mathbf{b}_1, \mathbf{b}_2)$ と $\mathfrak{B}' = (\mathbf{b}'_1, \mathbf{b}'_2)$ を与えよう。ただし、

$$\mathbf{b}_1 = \begin{pmatrix} -1 \\ 2 \end{pmatrix}, \quad \mathbf{b}_2 = \begin{pmatrix} 3 \\ 1 \end{pmatrix}, \quad \mathbf{b}'_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \mathbf{b}'_2 = \begin{pmatrix} 2 \\ -2 \end{pmatrix}$$

とする。このとき、$\mathfrak{B}$ と $\mathfrak{B}'$ は行列として、

$$\mathfrak{B} = \begin{pmatrix} -1 & 3 \\ 2 & 1 \end{pmatrix}, \quad \mathfrak{B}' = \begin{pmatrix} 1 & 2 \\ 1 & -2 \end{pmatrix}$$

と書け、基底の変換 $\mathfrak{B} \to \mathfrak{B}'$ の変換行列 Q は $Q = \mathfrak{B}'^{-1}\mathfrak{B}$ で与えられる。今、線形写像 $f\colon V \to W$ の基底対 $(\mathfrak{A}, \mathfrak{B})$ に関する表現行列が

$$F = \begin{pmatrix} 2 & 0 & -1 \\ 4 & 1 & -5 \end{pmatrix}$$

で与えられているとして、f の基底対 $(\mathfrak{A}', \mathfrak{B}')$ に関する表現行列 F' を求めてみよう。

(C1) `A1 : matrix([2, 3, 1], [-2, 0, 1], [1, -1, 2]);`

(D1) $$\begin{pmatrix} 2 & 3 & 1 \\ -2 & 0 & 1 \\ 1 & -1 & 2 \end{pmatrix}$$

(C2) `A2 :matrix([1, 3, 1], [1, 1, -2], [0, 3, 4]);`

(D2) $$\begin{pmatrix} 1 & 3 & 1 \\ 1 & 1 & -2 \\ 0 & 3 & 4 \end{pmatrix}$$

(C3) `P : A2^^(-1) . A1;`

(D3) $$\begin{pmatrix} 31 & 37 & -13 \\ -13 & -15 & 6 \\ 10 & 11 & -4 \end{pmatrix}$$

(C4) `B1 : matrix([-1, 3], [2, 1]);`

(D4) $$\begin{pmatrix} -1 & 3 \\ 2 & 1 \end{pmatrix}$$

(C5) `B2 : matrix([1, 2], [1, -2]);`

(D5) $$\begin{pmatrix} 1 & 2 \\ 1 & -2 \end{pmatrix}$$

(C6) `Q : B2^^(-1) . B1;`

(D6) $$\begin{pmatrix} \frac{1}{2} & 2 \\ -\frac{3}{4} & \frac{1}{2} \end{pmatrix}$$

(C7) `F1 : matrix([2, 0, -1], [4, 1, -5]);`

(D7) $$\begin{pmatrix} 2 & 0 & -1 \\ 4 & 1 & -5 \end{pmatrix}$$

(C8) `F2 : Q . F1 . (P^^(-1));`

(D8) $$\begin{pmatrix} -\frac{223}{38} & -\frac{495}{38} & \frac{41}{19} \\ -\frac{45}{76} & -\frac{181}{76} & -\frac{23}{19} \end{pmatrix}$$

(C9) `Q^^(-1) . F2 . P;`

(D9) $$\begin{pmatrix} 2 & 0 & -1 \\ 4 & 1 & -5 \end{pmatrix}$$

上の計算より、

$$Q = \begin{pmatrix} \frac{1}{2} & 2 \\ -\frac{3}{4} & \frac{1}{2} \end{pmatrix}$$

及び、

$$F' = QFP^{-1} = \begin{pmatrix} -\frac{223}{38} & -\frac{495}{38} & \frac{41}{19} \\ -\frac{45}{76} & -\frac{181}{76} & -\frac{23}{19} \end{pmatrix}$$

を得る。 □

9. ベクトルの内積

9.1. **ベクトルの内積について.** ベクトルの内積とは何だろうか？ 高校で習った知識によると、それは2つの平面あるいは空間ベクトル $\mathbf{a}$ と $\mathbf{b}$ に対して、一つの実数 $\mathbf{a} \bullet \mathbf{b}$ が決まり、これを $\mathbf{a}$ と $\mathbf{b}$ の内積と言ったのであった。そして、その定義は θ を $\mathbf{a}$ と $\mathbf{b}$ の間の角とするとき、

$$\mathbf{a} \bullet \mathbf{b} := |\mathbf{a}||\mathbf{b}| \cos\theta$$

で与えられるものだった。あるいは、例えば、成分で $\mathbf{a} = (a_1, a_2, a_3)$、$\mathbf{b} = (b_1, b_2, b_3)$ と与えられているならば、

$$\mathbf{a} \bullet \mathbf{b} = a_1 b_1 + a_2 b_2 + a_3 c_3$$

としてよいことも習ったであろう。さて、内積を一般のベクトル空間でも定義するためには、すこし反省してみることが必要である。勝手なベクトル空間におけるベクトルの長さとか、2つのベクトルの間の角とか、そんなものがあるのかどうかわからないし、成分にしたって何を基準に決めてよいかわからないからである。

そこで、平面あるいは空間ベクトルの内積の持つ形式的な性質を抽象して、一般のベクトル空間における内積を定義することにしよう。与えられたベクトル空間 V 上の内積とは、直積空間 $V \times V$ から、スカラー体 $\mathbb{K}$ への写像 g であって、任意の $\mathbf{a}_1, \mathbf{a}_2, \mathbf{b}_1, \mathbf{b}_2 \in V$ と、任意のスカラー $\alpha_1, \alpha_2, \beta_1, \beta_2 \in \mathbb{K}$ に対して、

$$g(\alpha_1\mathbf{a}_1+\alpha_2\mathbf{a}_2, \beta_1\mathbf{b}_1+\beta_2\mathbf{b}_2) = \overline{\alpha_1}\beta_1 g(\mathbf{a}_1, \mathbf{b}_1)+\overline{\alpha_1}\beta_2 g(\mathbf{a}_1, \mathbf{b}_2)+\overline{\alpha_2}\beta_1 g(\mathbf{a}_2, \mathbf{b}_1)+\overline{\alpha_2}\beta_2 g(\mathbf{a}_2, \mathbf{b}_2)$$

及び、任意の $\mathbf{a}, \mathbf{b} \in V$ に対して

$$g(\mathbf{a}, \mathbf{b}) = \overline{g(\mathbf{b}, \mathbf{a})}$$

を満たすものである。一般には、さらに次の性質を附化して考えることが普通である。

$$g(\mathbf{a}, \mathbf{a}) \geq 0$$

であり、等号成立は $\mathbf{a} = \mathbf{0}$ のときに限る。この最後の性質を満たす内積を特に**正定値内積**というが、単に内積と言ってしまうことが多い。それで、普通困らないのであるが、例えば相対性理論などでは、正定値でない内積を扱うのでそういう場合は意識的にこの性質を除いて考える必要がある。

さて、この文書では **maxima** で実際に計算できる話しに注意を向けているので、考えるベクトル空間はすべて有限次元とし、スカラー体 $\mathbb{K}$ は複素数体 $\mathbb{C}$ または 実数体 $\mathbb{R}$ とする。ベクトル空間 V が n 次元であるとすると、V の基底 $\mathfrak{A} = \{\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_n\}$ を一つ決めることができる。前節のように、任意のベクトル $\mathbf{a} \in V$ は、この基底を用いて

$$\mathbf{a} = \alpha_1\mathbf{a}_1 + \alpha_2\mathbf{a}_2 + \cdots + \alpha_n\mathbf{a}_n$$

と書け、このとき

$$\mathbf{a}_{\mathfrak{A}} = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{pmatrix} \in \mathbb{K}$$

を $\mathbf{a}$ の $\mathfrak{A}$ に関する成分という。V の任意のベクトル $\mathbf{a}$、$\mathbf{b}\in V$ を考え、それらの $\mathfrak{A}$ に関する成分がそれぞれ

$$\mathbf{a}_{\mathfrak{A}}=\begin{pmatrix}\alpha_1\\ \alpha_2\\ \vdots\\ \alpha_n\end{pmatrix},\qquad \mathbf{b}_{\mathfrak{A}}=\begin{pmatrix}\beta_1\\ \beta_2\\ \vdots\\ \beta_n\end{pmatrix}$$

であるとする。このとき、$g(\mathbf{a},\mathbf{b})$ は内積の性質により、

$$\begin{aligned}g(\mathbf{a},\mathbf{b})&=\overline{\alpha_1}\beta_1 g(\mathbf{a}_1,\mathbf{a}_1)+\overline{\alpha_1}\beta_2 g(\mathbf{a}_1,\mathbf{a}_2)+\cdots+\overline{\alpha_n}\beta_n g(\mathbf{a}_n,\mathbf{a}_n)\\&=\sum_{i=1}^{n}\sum_{j=1}^{n}\overline{\alpha_i}\beta_j g(\mathbf{a}_i,\mathbf{a}_j)\end{aligned}$$

となる。ここで、簡単のため

$$g_{ij}:=g(\mathbf{a}_i,\mathbf{a}_j),\quad \forall i,\forall j\in\{1,2,\ldots,n\}$$

と書くことにする。この $g_{i,j}$ たちを行列のかたちに並べて

$$G:=\begin{pmatrix}g_{11}&g_{12}&\cdots&g_{1n}\\ g_{21}&g_{22}&\cdots&g_{2n}\\ \vdots&\vdots&\ddots&\vdots\\ g_{n1}&g_{n2}&\cdots&g_{nn}\end{pmatrix}$$

と書くことにする。内積 g の性質は、基底 $\mathfrak{A}$ を定めれば、この行列 G によって完全に決まってしまう。G を g の $\mathfrak{A}$ に関する**計量行列**という。g を G を使って表現すると次のようになる。

$$\begin{aligned}g(\mathbf{a},\mathbf{b})&=\overline{(\alpha_1,\alpha_2,\ldots,\alpha_n)}\begin{pmatrix}g_{11}&g_{12}&\cdots&g_{1n}\\ g_{21}&g_{22}&\cdots&g_{2n}\\ \vdots&\vdots&\ddots&\vdots\\ g_{n1}&g_{n2}&\cdots&g_{nn}\end{pmatrix}\begin{pmatrix}\beta_1\\ \beta_2\\ \vdots\\ \beta_n\end{pmatrix}\\&=\overline{{}^t\mathbf{a}_{\mathfrak{A}}}G\mathbf{b}_{\mathfrak{A}}.\end{aligned}$$

ところで、$g(\mathbf{a},\mathbf{b})=\overline{g(\mathbf{b},\mathbf{a})},\forall\mathbf{a},\forall\mathbf{b}\in V$ であるから、

$$g_{ij}=\overline{g_{ji}},\quad \forall i,\forall j\in\{1,2,\ldots,n\}$$

が成り立つ。すなわち、

$$G=\overline{{}^tG}$$

であり、$\mathbb{K}=\mathbb{C}$ ならば G はエルミート行列、$\mathbb{K}=\mathbb{R}$ ならば G は対称行列である。

話しを少し簡単にして、$V=\mathbb{K}^n$ の場合を考えよう。V の基底として、標準基底 $\mathfrak{E}=(\mathbf{e}_1,\mathbf{e}_2,\ldots,\mathbf{e}_n)$ を使うことにすれば、任意の $\mathbf{a}\in V$ に対して、$\mathbf{a}=\mathbf{a}_{\mathfrak{E}}$ となるから、ベクトルとその $\mathfrak{E}$ に関する成分を区別しないで用いることにする。任意に n 次エルミート行列 G を与えると

$$g(\mathbf{a},\mathbf{b}):=\overline{{}^t\mathbf{a}}G\mathbf{b},\quad \forall\mathbf{a},\forall\mathbf{b}\in V$$

により、(正定値とは限らない) 内積 $g\colon V\times V\longrightarrow\mathbb{K}$ が一つ定まる。このことを使って、与えられた計量行列 G から内積 g を作る **maxima** のコマンドを作ってみよう。以下の内容を “metrics.mac” と名付けてファイルに保存する。

```
/* metrics.mac */

load(eigen)$

geom(a, b, G) := block([ta],
    ta : conjugate(transpose(a)),
    ratsimp(ta . G . b)
)$
```

コマンド **geom** に引数として与える行列 G はエルミート行列あるいは対称行列でないと内積にはならない。早速テストしてみよう。

例 57. (C1) `load("metrics.mac");`

Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS

(D1) metrics.mac

(C2) `G : matrix([1, 2 - %i], [2 + %i, 3]);`

(D2) $\begin{pmatrix} 1 & 2-i \\ i+2 & 3 \end{pmatrix}$

(C3) `a : matrix([s], [t]);`

(D4) $\begin{pmatrix} s \\ t \end{pmatrix}$

(C5) `b : matrix([x], [y]);`

(D5) $\begin{pmatrix} x \\ y \end{pmatrix}$

(C6) `geom(a, b, G);`

(D6) $(3t + (2-i)\,s)\,y + ((i+2)\,t + s)\,x$

(C7) `geom(b, a, G);`

(D7) $(3t + (i+2)\,s)\,y + ((2-i)\,t + s)\,x$

(C8) `geom(a, a, G);`

(D8) $3t^2 + 4st + s^2$

(C9) `geom(b, b, G);`

(D9) $3y^2 + 4xy + x^2$

(C10) `kill(a,b);`

(D10) DONE

(C11) `a : matrix([1], [2]);`

(D11) $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$

(C12) `b : matrix([2 - 3 * %i], [3 + %i]);`

(D12) $\begin{pmatrix} 2-3i \\ i+3 \end{pmatrix}$

(C13) `geom(a, b, G);`

(D13) $41 - 6i$

(C14) `geom(b, a, G);`

(D14) $6i + 41$

(C15) `geom(a, a, G);`

(D15) 21

```
(C16) geom(b, b, G);
(D16) 77
```

さて、計量行列 G が単位行列 I である場合:

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

すなわち、

$$g_{ij} = \delta_{ij} = \begin{cases} 1 & (i = j) \\ 0 & (i \neq j) \end{cases}$$

であるとき、これに対応する内積 g を $\mathbb{K}^n$ の**標準内積**という。特に $\mathbb{K} = \mathbb{C}$ のとき $\mathbb{C}^n$ の標準内積をエルミート内積、$\mathbb{K} = \mathbb{R}$ のとき $\mathbb{R}^n$ の標準内積をユークリッド内積ともいう。**maxima** には、標準内積を計算するコマンド **innerproduct** が用意されている。また省略形 **inprod** も使える。

例 58. (C1) load("metrics.mac")

```
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
(D1) metrics.mac
(C2) a : transpose(matrix([s, t, u]));
```

(D2) $\begin{pmatrix} s \\ t \\ u \end{pmatrix}$

(C3) b : transpose(matrix([x, y, z]));

(D5) $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$

(C6) inprod(a, b);

(D6) $uz + ty + sx$

(C7) I : ident(3);

(D7) $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

(C8) geom(a, b, I);

(D8) $uz + ty + sx$

(C9) kill(a, b)

(D9) DONE

(C10) a : matrix([s, t, u]);

(D10) $\begin{pmatrix} s & t & u \end{pmatrix}$

(C11) b : matrix([x, y, z]);

(D11) $\begin{pmatrix} x & y & z \end{pmatrix}$

```
(C12) inprod(a, b);
(D12) uz + ty + sx
(C13) geom(a, b, I);
(D13) uz + ty + sx
(C14) kill(a, b);
(D14) DONE
(C15) a : [s, t, u];
(D18) [s, t, u]
(C19) b : [x, y, z];
(D16) [x, y, z]
(C17) inprod(a, b);
(D19) uz + ty + sx
(C20) geom(a, b, I);
(D20) uz + ty + sx
```

inprod を使う前には "load(eigen)" でライブラリを読み込んでおく必要がある。標準内積は正定値内積である。標準内積 g に対し、値 $g(\mathbf{a}, \mathbf{b})$ を

$$\langle \mathbf{a}, \mathbf{b} \rangle := g(\mathbf{a}, \mathbf{b})$$

と表すことにする。$\mathbb{K}^n$ の任意のベクトル $\mathbf{a}$ に対して、$\mathbf{a}$ の絶対値 (大きさ、長さ) $||\mathbf{a}||$ を

$$||\mathbf{a}||^2 := \langle \mathbf{a}, \mathbf{a} \rangle$$

により定義する。また、$\mathbb{K} = \mathbb{R}$ のとき、$\mathbb{R}^n$ の任意のゼロベクトルでない 2 つのベクトル $\mathbf{a}, \mathbf{b}$ に対して、$\mathbf{a}$ と $\mathbf{b}$ の間の角 $\theta \quad (0 \leq \theta \leq \pi)$ を

$$\cos\theta := \frac{\langle \mathbf{a}, \mathbf{b} \rangle}{||\mathbf{a}||\,||\mathbf{b}||}$$

で定義する。

以上、標準計量をもとにして、ベクトルの絶対値、間の角を定義した。話しの順序は逆であるが、これで標準内積が高校で学んだ平面と空間ベクトルの内積の一般化になっていることがわかった。

一般のベクトル空間 V の話しに戻る。V と V 上の内積 g が与えられれば、標準内積の場合のアナロジーとして、

$$||\mathbf{a}||^2 := g(\mathbf{a}, \mathbf{b}),$$
$$\cos\theta := \frac{g(\mathbf{a}, \mathbf{b})}{||\mathbf{a}||\,||\mathbf{b}||}$$

により、g に関するベクトルの絶対値、2 つのベクトルの間の角を定義できることに気付くだろう。$V = \mathbb{K}^n$ の場合に、与えられた計量行列 G から決まる内積 g に関するベクトルの絶対値を求めるコマンド **absolute** と、2 つのベクトルの間の角の cosine を求めるコマンド **anglcos** を定義しよう。以下のプログラムを "metrics.mac" に追加しよう。

```
absolute(a, G) := ratsimp(sqrt(geom(a, a, G)))$

anglcos(a, b, G) := block([denom],
    denom : absolute(a, G) * absolute(a, G),
```

```
    ratsimp(geom(a, b, G) / denom)
)$
```

これを、使った計算例を見てみよう。

例 59. (C1) `load("metrics.mac");`

```
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
```

(D1) metrics.mac

(C2) `I : ident(4);`

(D2) $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

(C3) `len(a) := absolute(a, I);`

(D3) $\mathrm{len}\,(a) := \mathrm{absolute}\,(a, I)$

(C4) `ancos(a, b) := anglcos(a, b, I);`

(D4) $\mathrm{ancos}\,(a, b) := \mathrm{anglcos}\,(a, b, I)$

(C5) `a : transpose(matrix([1, -2, 0, -1]));`

(D5) $\begin{pmatrix} 1 \\ -2 \\ 0 \\ -1 \end{pmatrix}$

(C6) `b : transpose(matrix([2, 1, 3, -1]));`

(D6) $\begin{pmatrix} 2 \\ 1 \\ 3 \\ -1 \end{pmatrix}$

(C7) `len(a);`

(D7) $\sqrt{6}$

(C8) `len(b);`

(D8) $\sqrt{15}$

(C9) `ancos(a, b);`

(D9) $\frac{1}{\sqrt{6}\sqrt{15}}$

(C10) `inprod(a, b)/ (len(a) * len(b));`

(D10) $\frac{1}{\sqrt{6}\sqrt{15}}$

(C11) `kill(a,b);`

(D11) DONE

(C12) `a : transpose(matrix([s, t, u, v]));`

(D12) $\begin{pmatrix} s \\ t \\ u \\ v \end{pmatrix}$

(C13) `b : transpose(matrix([w, x, y, z]));`

(D13) $\begin{pmatrix} w \\ x \\ y \\ z \end{pmatrix}$

(C14) `len(a);`

(D14) $\sqrt{v^2+u^2+t^2+s^2}$

(C15) `len(b);`

(D15) $\sqrt{z^2+y^2+x^2+w^2}$

(C16) `inprod(a, b);`

(D16) $vz+uy+tx+sw$

(C17) `ancos(a, b);`

(D17) $\frac{vz+uy+tx+sw}{\sqrt{v^2+u^2+t^2+s^2}\sqrt{z^2+y^2+x^2+w^2}}$

注意しておきたいことは、内積も絶対値も角も、ベクトル空間 V の特別な基底の取り方に依存しない概念であるということである。($\mathbb{K}^n$ の標準計量ですら、標準基底によらずに定まるものである。) 基底の取り方を変えたときに変わるのは計量行列の成分だけである。V に 2 通りの基底 $\mathfrak{A}$ と $\mathfrak{B}$ を与える。2 つのベクトル $\mathbf{a}$ と $\mathbf{b}$ の $\mathfrak{A}$ に関する成分をそれぞれ $\mathbf{a}_{\mathfrak{A}}$、$\mathbf{b}_{\mathfrak{A}}$ とし、 $\mathbf{a}$ と $\mathbf{b}$ の $\mathfrak{B}$ に関する成分をそれぞれ $\mathbf{a}_{\mathfrak{B}}$、$\mathbf{b}_{\mathfrak{B}}$ とする。また、内積 g の $\mathfrak{A}$ に関する計量行列を $G_{\mathfrak{A}}$、$\mathfrak{B}$ に関する計量行列を $G_{\mathfrak{B}}$ とする。内積 $g(\mathbf{a},\mathbf{b})$ そのものは基底の取り方に依らないのであるから、

$$\overline{{}^t\mathbf{a}_{\mathfrak{A}}}G_{\mathfrak{A}}\mathbf{b}_{\mathfrak{A}} = \overline{{}^t\mathbf{a}_{\mathfrak{B}}}G_{\mathfrak{B}}\mathbf{b}_{\mathfrak{B}} \quad (= g(\mathbf{a},\mathbf{b}))$$

が成立しているはずである。今、基底の変換 $\mathfrak{A} \to \mathfrak{B}$ の変換行列を P とすると、

$$\mathfrak{A} = \mathfrak{B}P$$

であり、

$$\mathbf{a}_{\mathfrak{B}} = P\mathbf{a}_{\mathfrak{A}}, \quad \mathbf{b}_{\mathfrak{B}} = P\mathbf{b}_{\mathfrak{A}}$$

である。それゆえ、

$$\overline{{}^t\mathbf{a}_{\mathfrak{B}}}G_{\mathfrak{B}}\mathbf{b}_{\mathfrak{B}} = \overline{{}^t\mathbf{a}_{\mathfrak{A}}{}^tP}G_{\mathfrak{B}}P\mathbf{b}_{\mathfrak{A}} = \overline{{}^t\mathbf{a}_{\mathfrak{A}}}G_{\mathfrak{A}}\mathbf{b}_{\mathfrak{A}}$$

を得る。$\mathbf{a}$ および $\mathbf{b}$ は任意であるから、

$$\overline{{}^tP}G_{\mathfrak{B}}P = G_{\mathfrak{A}}$$

あるいは、

$$\overline{{}^tP^{-1}}G_{\mathfrak{A}}P^{-1} = G_{\mathfrak{B}}$$

を得る。これが、基底の変換にともなう計量行列の成分の変換則である。

ベクトル空間 V に内積 g が与えられているとき、V と g をセットにして考え、(V,g) を**内積空間**あるいは、**計量ベクトル空間という。**特に、$(\mathbb{C}^n,\langle,\rangle)$ をエルミート空間、$(\mathbb{R}^n,\langle,\rangle)$ をユークリッド空間という。

正値でない内積の例も扱っておこう。$\mathbb{R}^n$ の標準基底 $\mathfrak{E}$ に関して、計量行列

$$M = \begin{pmatrix} 1 & & & O \\ & -1 & & \\ & & \ddots & \\ O & & & -1 \end{pmatrix}$$

により定まる $\mathbb{R}^n$ の内積 η を $(1, n-1)$ 型のローレンツ内積という。特に、$\mathbb{R}^4$ のとき $(1,3)$ 型のローレンツ内積をミンコフスキー内積という。そして、内積空間 $(\mathbb{R}^4, \eta)$ を $\mathbb{R}^{1,3}$ と書いてミンコフスキー空間という。これは、アインシュタインの特殊相対性理論を数学的に定式化するときに出てくるものである。任意のベクトル $\mathbf{a} \in \mathbb{R}^{1,3}$ に対して、η に関する自己内積 $\eta(\mathbf{a}, \mathbf{a})$ はもはや非負とは限らない。$\eta(\mathbf{a}, \mathbf{a}) < 0$ のとき、$\mathbf{a}$ を空間的ベクトル、$\eta(\mathbf{a}, \mathbf{a}) = 0$ のとき、$\mathbf{a}$ を光的ベクトル、そして、$\eta(\mathbf{a}, \mathbf{a}) > 0$ のとき、$\mathbf{a}$ を時間的ベクトルという。普通の意味では絶対値を定義することはできないので、かわりにミンコフスキーノルムというものを考える (記号は絶対値と同じ記号を使う)。 その定義は、

$$||\mathbf{a}|| := \begin{cases} \sqrt{-\eta(\mathbf{a}, \mathbf{a})} & \text{空間的な場合} \\ 0 & \text{光的な場合} \\ \sqrt{\eta(\mathbf{a}, \mathbf{a})} & \text{時間的な場合} \end{cases}$$

である。ベクトル $\mathbf{a}$ の第 1 成分が正のとき、ベクトル $\mathbf{a}$ は未来的であると言うことにしよう。2 つの時間的ベクトル $\mathbf{a}, \mathbf{b} \in \mathbb{R}^{1,3}$ に対し、$\mathbf{b} - \mathbf{a}$ が未来的時間的ベクトルであるとき、$\mathbf{a}$ から $\mathbf{b}$ をみた**双曲角** μ を

$$\cosh \mu := \frac{\eta(\mathbf{a}, \mathbf{b})}{||\mathbf{a}||\,||\mathbf{b}||}$$

で定義する。

maxima でこれらを計算させるコマンドを作ってみよう。**MinkMatrix** はミンコフスキー内積の計量行列をあたえる。**mink** はミンコフスキー内積、**tminknorm**、**sminknorm** はそれぞれ時間的ベクトル、空間的ベクトルのミンコフスキーノルムを計算する。**minknorm** は引数のベクトルが数値として具体的に与えられている場合に、そのミンコフスキーノルムを計算する。ただし、文字式をいれるとエラーになる。2 つの時間的ベクトルの間の双曲角の hyperbolic cosine を計算するには **hypangcos** を用いる。これらを、以下のようにコーディングしよう。

```
MinkMatrix : ematrix(4, 4, 2, 1, 1) - ident(4)$

mink(a, b) := geom(a, b, MinkMatrix)$

tminknorm(a) := sqrt(mink(a,a))$

sminknorm(a) := sqrt(-mink(a,a))$

minknorm(a) := block(
    if mink(a, a) < 0 then
        sqrt(-mink(a, a))
    else
        sqrt(mink(a, a))
)$

hypangcos(a, b) := ratsimp(mink(a, b) / (tminknorm(a) * tminknorm(b)))$
```

再び、”metrics.mac” に追加しておこう。計算例は以下の通りである。

例 60. (C1) load("metrics.mac");

```
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
```

(D1) metrics.mac

(C2) MinkMatrix;

(D2) $$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

(C3) a : transpose(matrix([s, t, u, v]));

(D3) $$\begin{pmatrix} s \\ t \\ u \\ v \end{pmatrix}$$

(C4) b : transpose(matrix([w, x, y, z]));

(D4) $$\begin{pmatrix} w \\ x \\ y \\ z \end{pmatrix}$$

(C5) mink(a, b);

(D5) $- vz - uy - tx + sw$

(C6) tminknorm(a);

(D6) $\sqrt{-v^2 - u^2 - t^2 + s^2}$

(C7) sminknorm(a);

(D7) $\sqrt{v^2 + u^2 + t^2 - s^2}$

(C8) hypangcos(a, b);

(D8) $-\frac{vz+uy+tx-sw}{\sqrt{-v^2-u^2-t^2+s^2}\sqrt{-z^2-y^2-x^2+w^2}}$

(C9) minknorm(a);

```
MACSYMA was unable to evaluate the predicate:
```

ERREXP 1

```
# 0:  minknorm(a=MATRIX([s],[t],[u],[v]))(metrics.mac line 10)
-- an error.  Quitting.  To debug this try DEBUGMODE(TRUE);)
```

(C10) kill(a, b);

(D10) DONE

(C11) a : transpose(matrix([5, 1, 0, -2]));

(D11) $$\begin{pmatrix} 5 \\ 1 \\ 0 \\ -2 \end{pmatrix}$$

(C12) b : transpose(matrix([6, 2, -1, 2]));

(D12) $$\begin{pmatrix} 6 \\ 2 \\ -1 \\ 2 \end{pmatrix}$$

```
(C13) mink(a, b);
(D13) 32
(C14) mink(a, a);
(D14) 20
(C15) mink(b, b);
(D15) 27
(C16) minknorm(a);
```
(D16) $2\sqrt{5}$
```
(C17) minknorm(b);
```
(D17) $3\sqrt{3}$
```
(C18) hypangcos(a, b);
```
(D18) $\frac{16}{3\sqrt{3}\sqrt{5}}$
```
(C19) c : transpose(matrix([1, 2, -1, 3]));
```
(D19) $\begin{pmatrix} 1 \\ 2 \\ -1 \\ 3 \end{pmatrix}$
```
(C20) mink(c, c);
```
(D20) -13
```
(C21) sminknorm(c);
```
(D21) $\sqrt{13}$
```
(C22) minknorm(c);
```
(D22) $\sqrt{13}$

9.2. **Gram-Schmidt の直交化.** 内積空間 (V, g) において、与えられた有限個のベクトルの順序列 $(\mathbf{a}_1, \dots, \mathbf{a}_k)$ に適当な操作を施して、**直交系** $(\mathbf{a}'_1, \dots, \mathbf{a}'_k)$ を作る方法が、**Gram-Schmidt の直交化法**と言う名で知られている。ここで、$(\mathbf{a}'_1, \dots, \mathbf{a}'_k)$ が直交系であると言うのは、

$$\begin{cases} g(\mathbf{a}'_i, \mathbf{a}'_i) \neq 0 & \forall i \in \{1, \dots, k\} \\ g(\mathbf{a}'_i, \mathbf{a}'_j) = 0 & \forall i \neq \forall j \in \{1, \dots, k\} \end{cases}$$

が成り立つことを意味する。条件 $g(\mathbf{v}, \mathbf{w}) = 0$ が成り立つとき、$\mathbf{v}$ と $\mathbf{w}$ は (g に関して) 直交すると言い $\mathbf{v} \perp \mathbf{w}$ と表すこともある。$(\mathbf{a}'_1, \dots, \mathbf{a}'_k)$ が直交系となるためには、$(\mathbf{a}_1, \dots, \mathbf{a}_k)$ も一定の条件を満たす必要がある。その条件は

$$\langle \{\mathbf{a}_1, \dots, \mathbf{a}_k\} \rangle = \langle \{\mathbf{a}'_1, \dots, \mathbf{a}'_k\} \rangle$$

である。これは、適当な操作が構成ベクトルの一つに他の構成ベクトルのスカラー倍を加えるという、可逆な操作を使うことによる要請である。ここで注意したいのは、$\{\mathbf{a}'_1, \dots, \mathbf{a}'_k\}$ は**線形独立**になることで、これは次のことが成り立つことを意味する。

$$\alpha_1 \mathbf{a}'_1 + \cdots + \alpha_k \mathbf{a}'_k = \mathbf{0}$$

であるとする。$\forall i \in \{1, \dots, k\}$ について、上の式の両辺と $\mathbf{a}'_i$ との内積を取ってみると、内積 g の性質より、

$$\alpha_i g(\mathbf{a}'_i, \mathbf{a}'_i) = 0$$

を得るが、$g(\mathbf{a}'_i,\mathbf{a}'_i)\neq 0$ であったから、$\alpha_i=0,\quad \forall i$ でなければならない。このことは、さらに $(\mathbf{a}'_1,\dots,\mathbf{a}'_k)$ が $\langle\{\mathbf{a}'_1,\dots,\mathbf{a}'_k\}\rangle$ の基底であることも示している。個数が等しいので、$\{\mathbf{a}_1,\dots,\mathbf{a}_k\}$ は線形独立でなければならない。以上から、議論の出発点を正確に述べることができるようになった。

改めて、(V,g) を n 次元内積空間とし、$(\mathbf{a}_1,\dots,\mathbf{a}_k)$ $(k\leq n)$ を V の k 個の線形独立なベクトルよりなる順序列とする。ただし、各ベクトルはヌルベクトルでは無い $(g(\mathbf{a}_i,\mathbf{a}_i)\neq 0,\forall i)$ とする。これに対し、順序列 $(\mathbf{a}'_1,\dots,\mathbf{a}'_k)$ を以下のように帰納的に構成する。まず、

$$\mathbf{a}'_1:=\mathbf{a}_1$$

とする。$(\mathbf{a}'_1,\dots,\mathbf{a}'_l)$ $(l\leq k-1)$ が直交系であるように構成できたとして、$\mathbf{a}_{l+1}{}'$ を

$$\mathbf{a}_{l+1}{}':=\mathbf{a}_{l+1}-\frac{g(\mathbf{a}_{l+1},\mathbf{a}'_1)}{g(\mathbf{a}'_1,\mathbf{a}'_1)}\mathbf{a}'_1-\cdots-\frac{g(\mathbf{a}_{l+1},\mathbf{a}'_l)}{g(\mathbf{a}'_l,\mathbf{a}'_l)}\mathbf{a}'_l$$

とする。このとき、$\forall i\in\{1,\dots,l\}$ に対して、

$$\begin{aligned}g(\mathbf{a}_{l+1}{}',\mathbf{a}'_i)&=g(\mathbf{a}_{l+1},\mathbf{a}'_i)-\frac{g(\mathbf{a}_{l+1},\mathbf{a}'_i)}{g(\mathbf{a}'_i,\mathbf{a}'_i)}g(\mathbf{a}'_i,\mathbf{a}'_i)\\&=0\end{aligned}$$

を得る。また、$\{\mathbf{a}'_1,\dots,\mathbf{a}'_l,\mathbf{a}_{l+1}\}$ の一次独立性より、

$$g(\mathbf{a}_{l+1}{}',\mathbf{a}_{l+1}{}')\neq 0$$

を得る。したがって、$(\mathbf{a}'_1,\dots,\mathbf{a}_{l+1}{}')$ はやはり直交系になる。この操作を繰り返し、最終的に、直交系 $(\mathbf{a}'_1,\dots,\mathbf{a}'_k)$ を得る。以上のようにして、$(\mathbf{a}_1,\dots,\mathbf{a}_k)$ から $(\mathbf{a}'_1,\dots,\mathbf{a}'_k)$ を作る方法を Gram-Schmidt の直交化法と言う。

さらに、内積 g が正定値であるときには、

$$\mathbf{a}''_i:=\frac{1}{\sqrt{g(\mathbf{a}'_i,\mathbf{a}'_i)}}\mathbf{a}'_i,\quad \forall i\in\{1,\dots,k\}$$

とすれば、$(\mathbf{a}''_1,\dots,\mathbf{a}''_k)$ は、いわゆる**正規直交系**と呼ばれるものになる。すなわち、

$$\begin{cases}g(\mathbf{a}''_i,\mathbf{a}''_i)=1 & \forall i\in\{1,\dots,k\}\\ g(\mathbf{a}''_i,\mathbf{a}''_j)=0 & \forall i\neq\forall j\in\{1,\dots,k\}\end{cases}$$

を満たす。$(\mathbf{a}_1,\dots,\mathbf{a}_k)$ から $(\mathbf{a}''_1,\dots,\mathbf{a}''_k)$ を求めるこの方法のことを **Gram-Schmidt の正規直交化法**と言う。

さて、大学初年級においては実用上は $\mathbb{R}^n$ あるいは $\mathbb{C}^n$ に標準内積を入れたもの、すなわちユークリッド空間あるいはエルミート空間の場合に限定して考察すれば十分である。以下では、標準内積の場合に限定して、Gram-Schmidt 法を **maxima** で実行する手順について説明しよう。

maxima のライブラリコマンド **gramschmidt** は、引数として、リストのリストをとり、値としてやはりリストのリストを返す。すなわち、各ベクトルをリストとして表現し、それらの並びでベクトルの順序列を表すのである。**maxima** においては、行列はリストのリストとしてもあつかえるので、引数に行列を与えることも可能である。この場合、行列の各行ベクトルが並んだ順序列と解釈される。gramschmidt を使うには、事前に “load(eigeni);” とタイプして、ライブラリを読み込んでおかねばならない。

例 61.

```
(C1) load(eigeni);
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
(D1) /usr/local/share/maxima/5.9.0/share/matrix/eigeni.mac
(C2) gramschmidt([[2,3],[1,-2]]);
```

(D2) $\left[[2,3], \left[\frac{37}{13}, -\frac{27}{13}\right]\right]$

```
(C3) a1 : [1, 2, 3];
```

(D3) $[1,2,3]$

```
(C4) a2 : [4, 5, 6];
```

(D4) $[4,5,6]$

```
(C5) a3 : [7, 8, -1];
```

(D5) $[7,8,-1]$

```
(C6) gramschmidt([a1, a2, a3]);
```

(D6) $\left[[1,2,3], \left[\frac{2^2 3}{7}, \frac{3}{7}, -\frac{23}{7}\right], \left[-\frac{5}{3}, \frac{25}{3}, -\frac{5}{3}\right]\right]$

```
(C7) A : matrix([1, 2, 3], [4, 5, 6], [7, 8, -1]);
```

(D7) $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & -1 \end{pmatrix}$

```
(C8) gramschmidt(A);
```

(D8) $\left[[1,2,3], \left[\frac{2^2 3}{7}, \frac{3}{7}, -\frac{23}{7}\right], \left[-\frac{5}{3}, \frac{25}{3}, -\frac{5}{3}\right]\right]$

ベクトルの順序列を、列ベクトルの並びとして、あるいはそれを行列として表現したもので与えたい場合がよく起こる。あるいは、正規直交化した結果が欲しい場合もあるだろう。**maxima** の標準ライブラリコマンドには直接そのような取り扱いのできるコマンドがないので、新たなコマンドとして自作することにしよう。以下のプログラムを “gson.mac” と言うファイルとして保存しよう。

```
/* gson.mac */

load(eigeni)$

gso(A) := block([m,n,i,B,C,D],
  m : length(A),
  B : transpose(A),
  n : length(B),
  C : gramschmidt(B),
  D : zeromatrix(n, m),
  for i:1 thru n do
    D[i] : C[i],
  transpose(D)
)$

gson(A) := block([m,n,i,B,C,D],
  m : length(A),
```

```
  B : transpose(A),
  n : length(B),
  C : gramschmidt(B),
  D : zeromatrix(n, m),
  for i:1 thru n do
    D[i] : unitvector(C[i]),
  transpose(D)
)$
```

ここで定義したコマンド **gso** と **gson** はともに行列を引数として、行列を値として返す。gso は引数の行列を列ベクトルの順序列と解釈して Gram-Schmidt 法を適用したものを返す。gson は Gram-Schmidt の正規直交化法を適用したものを返す。コマンド定義に使われている **maxima** のコマンド **unitvector** は引数にリストを取り、それをベクトルと思って正規化 (単位ベクトル化) したものを値として返すものである。

例 62. (C1) `load("gson.mac");`

```
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
```

(D1) gson.mac

(C2) `A : transpose(matrix([2, -1], [-1, 2]));`

(D2) $\begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}$

(C3) `ratsimp(gso(A));`

(D3) $\begin{pmatrix} 2 & \frac{3}{5} \\ -1 & \frac{6}{5} \end{pmatrix}$

(C4) `ratsimp(gson(A));`

(D4) $\begin{pmatrix} \frac{2}{\sqrt{5}} & \frac{\sqrt{5}}{5} \\ -\frac{1}{\sqrt{5}} & \frac{2\sqrt{5}}{5} \end{pmatrix}$

(C5) `a[1] : transpose(D4)[1];`

(D5) $\left[\frac{2}{\sqrt{5}}, -\frac{1}{\sqrt{5}}\right]$

(C6) `a[2] : transpose(D4)[2];`

(D6) $\left[\frac{\sqrt{5}}{5}, \frac{2\sqrt{5}}{5}\right]$

(C7) `inprod(a[1], a[2]);`

(D7) 0

(C8) `A : transpose(matrix([1, 2, 3], [4, 5, 6], [7, 8, -1]));`

(D8) $\begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & -1 \end{pmatrix}$

(C9) `ratsimp(gso(A));`

(D9) $\begin{pmatrix} 1 & \frac{12}{7} & -\frac{5}{3} \\ 2 & \frac{3}{7} & \frac{10}{3} \\ 3 & -\frac{6}{7} & -\frac{5}{3} \end{pmatrix}$

(C10) `ratsimp(gson(A));`

(D10) $$\begin{pmatrix} \frac{1}{\sqrt{14}} & \frac{4\sqrt{7}}{7\sqrt{3}} & -\frac{\sqrt{3}}{3\sqrt{2}} \\ \frac{2}{\sqrt{14}} & \frac{\sqrt{3}\sqrt{7}}{21} & \frac{\sqrt{2}\sqrt{3}}{3} \\ \frac{3}{\sqrt{14}} & -\frac{2\sqrt{3}\sqrt{7}}{21} & -\frac{\sqrt{3}}{3\sqrt{2}} \end{pmatrix}$$

(C11) `B : transpose(matrix([1, 0, -2, 1], [2, 3, 3, -1], [0, 1, -1, -1]));`

(D11) $$\begin{pmatrix} 1 & 2 & 0 \\ 0 & 3 & 1 \\ -2 & 3 & -1 \\ 1 & -1 & -1 \end{pmatrix}$$

(C12) `rank(B);`

(D12) 3

(C13) `ratsimp(gso(B));`

(D13) $$\begin{pmatrix} 1 & \frac{17}{6} & -\frac{50}{113} \\ 0 & 3 & \frac{80}{113} \\ -2 & \frac{4}{3} & -\frac{90}{113} \\ 1 & -\frac{1}{6} & -\frac{130}{113} \end{pmatrix}$$

(C14) `ratsimp(gson(B));`

(D14) $$\begin{pmatrix} \frac{1}{\sqrt{6}} & \frac{17\sqrt{6}}{6\sqrt{113}} & -\frac{5\sqrt{113}}{113\sqrt{3}} \\ 0 & \frac{3\sqrt{6}}{\sqrt{113}} & \frac{8\sqrt{113}}{113\sqrt{3}} \\ -\frac{2}{\sqrt{6}} & \frac{4\sqrt{6}}{3\sqrt{113}} & -\frac{3\sqrt{3}\sqrt{113}}{113} \\ \frac{1}{\sqrt{6}} & -\frac{\sqrt{6}}{6\sqrt{113}} & -\frac{13\sqrt{113}}{113\sqrt{3}} \end{pmatrix}$$

引数の行列が正則行列である場合、コマンド **gson** の返り値は、次の小節で議論する直交行列、あるいはユニタリ行列になる。$A = (\mathbf{a}_1, \dots, \mathbf{a}_n)$ が正則行列であるとする。このとき、$A'' = (\mathbf{a}''_1, \dots, \mathbf{a}''_n) = \mathrm{gson}(A)$ を正規直交化された行列とすると、

$$\overline{{}^tA'} = \begin{pmatrix} \overline{{}^t\mathbf{a}''_1} \\ \vdots \\ \overline{{}^t\mathbf{a}''_n} \end{pmatrix}$$

であり、

$$\overline{{}^tA'}A' = \begin{pmatrix} \overline{{}^t\mathbf{a}''_1}\mathbf{a}''_1 & \overline{{}^t\mathbf{a}''_1}\mathbf{a}''_2 & \cdots & \overline{{}^t\mathbf{a}''_1}\mathbf{a}''_n \\ \overline{{}^t\mathbf{a}''_2}\mathbf{a}''_1 & \overline{{}^t\mathbf{a}''_2}\mathbf{a}''_2 & \cdots & \overline{{}^t\mathbf{a}''_2}\mathbf{a}''_n \\ \vdots & \vdots & \ddots & \vdots \\ \overline{{}^t\mathbf{a}''_n}\mathbf{a}''_1 & \overline{{}^t\mathbf{a}''_n}\mathbf{a}''_2 & \cdots & \overline{{}^t\mathbf{a}''_n}\mathbf{a}''_n \end{pmatrix}$$

となる。ここで、$\forall i, \forall j \in \{1, \dots, n\}$ に対して、

$$\overline{{}^t\mathbf{a}''_i}\mathbf{a}''_j = \langle \mathbf{a}''_i, \mathbf{a}''_j \rangle = \begin{cases} 1 & (i = j) \\ 0 & (i \neq j) \end{cases}$$

であることに注意すれば、

$$\overline{{}^tA''}A'' = I_n$$

を得る。したがって、

$$\overline{{}^tA''} = (A'')^{-1}$$

であり、$\mathbb{K} = \mathbb{R}$ ならば A'' は直交行列、$\mathbb{K} = \mathbb{C}$ ならば A'' はユニタリ行列である。

例 63. (C1) load("gson.mac");

Warning - you are redefining the MACSYMA function EIGENVALUES

Warning - you are redefining the MACSYMA function EIGENVECTORS

(D1) gson.mac

(C2) A : matrix([1, 2], [3, 4]);

(D2) $\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$

(C3) AA : ratsimp(gson(A));

(D3) $\begin{pmatrix} \frac{1}{\sqrt{10}} & \frac{3\sqrt{5}}{5\sqrt{2}} \\ \frac{3}{\sqrt{10}} & -\frac{\sqrt{5}}{5\sqrt{2}} \end{pmatrix}$

(C4) transpose(AA);

(D4) $\begin{pmatrix} \frac{1}{\sqrt{10}} & \frac{3}{\sqrt{10}} \\ \frac{3\sqrt{5}}{5\sqrt{2}} & -\frac{\sqrt{5}}{5\sqrt{2}} \end{pmatrix}$

(C5) AA^^(-1);

(D5) $\begin{pmatrix} \frac{\sqrt{10}}{10} & \frac{3\sqrt{10}}{10} \\ \frac{3\sqrt{2}}{2\sqrt{5}} & -\frac{\sqrt{2}}{2\sqrt{5}} \end{pmatrix}$

(C6) ratsimp(AA^^(-1) - transpose(AA));

(D6) $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$

(C7) ratsimp(transpose(AA) . AA);

(D7) $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

(C8) B : matrix([1+%i, 2, -%i], [0, 2-%i, 1-2*%i], [0, 0, 2+3*%i]);

(D8) $\begin{pmatrix} i+1 & 2 & -i \\ 0 & 2-i & 1-2i \\ 0 & 0 & 3i+2 \end{pmatrix}$

(C9) rank(B);

(D9) 3

(C10) BB : ratsimp(gson(B));

(D10) $\begin{pmatrix} \frac{i+1}{\sqrt{2}} & 0 & 0 \\ 0 & -\frac{i-2}{\sqrt{5}} & 0 \\ 0 & 0 & \frac{3i+2}{\sqrt{13}} \end{pmatrix}$

(C11) conjugate(transpose(BB));

(D11) $\begin{pmatrix} \frac{1-i}{\sqrt{2}} & 0 & 0 \\ 0 & -\frac{-i-2}{\sqrt{5}} & 0 \\ 0 & 0 & \frac{2-3i}{\sqrt{13}} \end{pmatrix}$

(C12) BB^^(-1);

(D12) $\begin{pmatrix} \frac{\sqrt{2}}{i+1} & 0 & 0 \\ 0 & -\frac{\sqrt{5}}{i-2} & 0 \\ 0 & 0 & \frac{\sqrt{13}}{3i+2} \end{pmatrix}$

(C13) ratsimp(conjugate(transpose(BB)) - BB^^(-1));

(D13) $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$

```
(C14) ratsimp(conjugate(transpose(BB)) . BB);
```

(D14) $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

```
(C15) C : matrix([1+%i, 2-%i], [3+2*%i, 4*%i]);
```

(D15) $\begin{pmatrix} i+1 & 2-i \\ 2i+3 & 4i \end{pmatrix}$

```
(C16) rank(C);
```

(D16) 2

```
(C17) CC : ratsimp(gson(C));
```

(D17) $\begin{pmatrix} \frac{i+1}{\sqrt{15}} & -\frac{11\sqrt{5}i-10\sqrt{5}}{5\sqrt{51}} \\ \frac{2i+3}{\sqrt{15}} & \frac{5\sqrt{5}i-3\sqrt{5}}{5\sqrt{51}} \end{pmatrix}$

```
(C18) ratsimp(conjugate(transpose(CC)) - CC^^(-1));
```

(D18) $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$

```
(C19) ratsimp(conjugate(transpose(CC)) . CC);
```

(D19) $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

9.3. **直交行列とユニタリ行列.** ベクトル空間 V から V 自身への線形写像のことを特に V 上の**線形変換**と言う。V に内積 g が与えられているとしよう。すなわち、(V,g) は内積空間とする。V 上の線形変換の中でも、内積を保つものは興味深い研究対象になる。

そこで、**内積を保つ線形変換**という概念を正確に記述しておこう。改めて (V,g) を内積空間とし、$f\colon V \to V$ を V 上の線形変換とする。f が内積 g を保つとは、条件

$$g(f(\mathbf{a}), f(\mathbf{b})) = g(\mathbf{a}, \mathbf{b}), \quad \forall \mathbf{a}, \forall \mathbf{b} \in V$$

が成り立つことを言う。

f が内積 g を保つ線形変換であるとき、任意の V の基底 $\mathfrak{A}$ に関して、f の基底対 $(\mathfrak{A},\mathfrak{A})$ に関する表現行列 F と g の $\mathfrak{A}$ に関する計量行列 G の間には何らかの関係があるのだろうか。以下では、それを考えてみる。V の任意のベクトル $\mathbf{a}$ と $\mathbf{b}$ の基底 $\mathfrak{A}$ に関する成分をそれぞれ $\mathbf{a}_{\mathfrak{A}}, \mathbf{b}_{\mathfrak{A}} \in \mathbb{K}^n$ としよう。ただし、$n = \dim V$ である。このとき、f が g を保つという条件を $\mathfrak{A}$ に関する成分表示で表すと、

$$\overline{{}^t(F\mathbf{a}_{\mathfrak{A}})}G(Ab_{\mathfrak{A}}) = \overline{{}^t\mathbf{a}_{\mathfrak{A}}}(\overline{{}^tF}GA)\mathfrak{b}_{\mathfrak{A}} = \overline{{}^t\mathbf{a}_{\mathfrak{A}}}G\mathbf{b}_{\mathfrak{A}}$$

となる。$\mathbf{a}_{\mathfrak{A}}$ および $\mathbf{b}_{\mathfrak{A}}$ は任意であるから、欲しかった条件は簡潔に

$$\overline{{}^tF}GF = G$$

と表現できる。この関係式を満たす行列 F を**計量行列** G **に関する直交行列**と言う。

ここで、特別な場合について考えておこう。$V = \mathbb{K}^n$ とし、内積として標準内積 $\langle \bullet, \bullet \rangle$ を考える。基底として標準基底 $\mathfrak{E}$ を考えよう。標準内積の標準基底に関する計量行列は

単位行列 I_n であった。そこで、標準内積を保つ任意の線形変換 f の基底対 $(\mathfrak{E},\mathfrak{E})$ に関する表現行列 F は I_n に関する直交行列となることが分かる。すなわち、

$$\overline{{}^tF}I_nF = \overline{{}^tF}F = I_n$$

が成り立つ。この条件を満たす行列は特に重要なので特別に名前が与えられている。$\mathbb{K}=\mathbb{R}$ のとき、

$$ {}^tFF = I_n$$

を満たす行列 F を単に**直交行列**と言う。$\mathbb{K}=\mathbb{C}$ のとき、

$$\overline{{}^tF}F = I_n$$

を満たす行列 F を**ユニタリ行列**と言う。また、これに対応して $\mathbb{K}^n$ 上の標準内積を保つ線形変換 f を $\mathbb{K}=\mathbb{R}$ のとき**直交変換**、$\mathbb{K}=\mathbb{C}$ のとき**ユニタリ変換**と言う。

再び、一般の正定値内積空間 (V,g) で話しをする。V の基底 $\mathfrak{A}=(\mathbf{a}_1,\dots,\mathbf{a}_n)$ が**正規直交基底**であるということを、条件

$$g(\mathbf{a}_i,\mathbf{a}_j) = \begin{cases} 1 & (i=j) \\ 0 & (i\neq j) \end{cases}$$

が満たされることと定義する。V の別の基底 $\mathfrak{B}=(\mathbf{b}_1,\dots\mathbf{b}_n)$ もまた正規直交基底であるとしよう。このとき、任意のベクトル $\mathbf{v}\in V$ の $\mathfrak{A}$、$\mathfrak{B}$ それぞれに関する成分 $\mathbf{v}_{\mathfrak{A}}$、$\mathbf{v}_{\mathfrak{B}}$ を考える。前節で述べたように、基底の変換 $\mathfrak{A}\to\mathfrak{B}$ の変換行列を P とすれば、

$$\mathbf{v}_{\mathfrak{B}} = P\mathbf{v}_{\mathfrak{A}}$$

がなりたつ。このとき、行列 P は係数体 $\mathbb{K}$ が $\mathbb{R}$ ならば直交行列であり、$\mathbb{C}$ ならばユニタリ行列である。なぜならば、

$$\langle\mathbf{e}_i,\mathbf{e}_j\rangle = g(\mathbf{a}_i,\mathbf{a}_j) = g(\mathbf{b}_i,\mathbf{b}_j),\quad \forall i,\forall j$$

であり、それゆえ

$$\langle P\mathbf{v}_{\mathfrak{A}}, P\mathbf{w}_{\mathfrak{A}}\rangle = \langle\mathbf{v}_{\mathfrak{B}},\mathbf{w}_{\mathfrak{B}}\rangle = g(\mathbf{v},\mathbf{w}) = \langle\mathbf{v}_{\mathfrak{A}},\mathbf{w}_{\mathfrak{A}}\rangle$$

が任意の $\mathbf{v},\mathbf{w}\in V$ に対して成立するから、上述のように

$$\overline{{}^tP}P = I_n$$

を得るからである。

例 64.
```
(C1) load("gson.mac");
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
(D1) gson.mac
(C2) A : ratsimp(gson(matrix([1,2,3],[4,5,6],[7,8,0])));
```

(D2) $$\begin{pmatrix} \frac{1}{\sqrt{66}} & \frac{3\sqrt{11}}{11} & -\frac{\sqrt{2}}{2\sqrt{3}} \\ \frac{4}{\sqrt{66}} & \frac{\sqrt{11}}{11} & \frac{\sqrt{2}}{\sqrt{3}} \\ \frac{7}{\sqrt{66}} & -\frac{\sqrt{11}}{11} & -\frac{\sqrt{2}}{2\sqrt{3}} \end{pmatrix}$$

```
(C3) B : ratsimp(gson(matrix([1,0,1],[0,1,2],[1,3,-1])));
```

(D3) $$\begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{3\sqrt{2}}{2\sqrt{11}} & \frac{\sqrt{11}}{11} \\ 0 & \frac{\sqrt{2}}{\sqrt{11}} & \frac{3\sqrt{11}}{11} \\ \frac{1}{\sqrt{2}} & \frac{3\sqrt{2}}{2\sqrt{11}} & -\frac{\sqrt{11}}{11} \end{pmatrix}$$

(C4) `P : A^^(-1) . B;`

(D4) $$\begin{pmatrix} \frac{4\sqrt{66}}{33\sqrt{2}} & \frac{13\sqrt{2}\sqrt{66}}{66\sqrt{11}} & \frac{\sqrt{11}\sqrt{66}}{121} \\ \frac{2}{\sqrt{2}\sqrt{11}} & -\frac{5\sqrt{2}}{11} & \frac{7}{11} \\ -\frac{\sqrt{3}}{3} & \frac{2\sqrt{3}}{3\sqrt{11}} & \frac{2\sqrt{3}\sqrt{11}}{11\sqrt{2}} \end{pmatrix}$$

(C5) `ratsimp(transpose(P) . P);`

(D5) $$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(C6) `ratsimp(transpose(A) . A);`

(D6) $$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(C7) `ratsimp(transpose(B) . B);`

(D7) $$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(C8) `C : ratsimp(gson(matrix([1+%i, 2], [2-%i, 3+2*%i])));`

(D8) $$\begin{pmatrix} \frac{i+1}{\sqrt{7}} & -\frac{11\sqrt{7}i-13\sqrt{7}}{7\sqrt{58}} \\ -\frac{i-2}{\sqrt{7}} & \frac{10\sqrt{7}i+4\sqrt{7}}{7\sqrt{58}} \end{pmatrix}$$

(C9) `D : ratsimp(gson(matrix([1, %i], [1-%i, 2-%i])));`

(D9) $$\begin{pmatrix} \frac{1}{\sqrt{3}} & \frac{\sqrt{3}i-3\sqrt{3}}{3\sqrt{5}} \\ -\frac{i-1}{\sqrt{3}} & -\frac{2\sqrt{3}i-\sqrt{3}}{3\sqrt{5}} \end{pmatrix}$$

(C10) `Q : ratsimp(C . D);`

(D10) $$\begin{pmatrix} \frac{\left(\sqrt{3}\sqrt{58}-24\sqrt{3}\right)i+\sqrt{3}\sqrt{58}+2\sqrt{3}}{3\sqrt{7}\sqrt{58}} & -\frac{\left(2\sqrt{3}\sqrt{5}\sqrt{58}+37\sqrt{3}\sqrt{5}\right)i+4\sqrt{3}\sqrt{5}\sqrt{58}+9\sqrt{3}\sqrt{5}}{15\sqrt{7}\sqrt{58}} \\ -\frac{\left(\sqrt{3}\sqrt{58}-6\sqrt{3}\right)i-2\sqrt{3}\sqrt{58}-14\sqrt{3}}{3\sqrt{7}\sqrt{58}} & \frac{\left(5\sqrt{3}\sqrt{5}\sqrt{58}+2\sqrt{3}\sqrt{5}\right)i-5\sqrt{3}\sqrt{5}\sqrt{58}+24\sqrt{3}\sqrt{5}}{15\sqrt{7}\sqrt{58}} \end{pmatrix}$$

(C11) `ratsimp(conjugate(transpose(Q)) . Q);`

(D11) $$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

(C12) `ratsimp(conjugate(transpose(C)) . C);`

(D12) $$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

(C13) `ratsimp(conjugate(transpose(D)) . D);`

(D13) $$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

この例では A, B, P が直交行列、C, D, Q がユニタリ行列である。 □

ユニタリ行列あるいは直交行列の行列式については次が成り立つ。A をユニタリ行列とすると、

$$|{}^t\overline{A}||A| = |{}^t\overline{A}A| = |I_n| = 1.$$

したがって、$|A|$ は絶対値が 1 の複素数であることがわかる。また、A が直交行列ならば、

$$|A|^2 = |{}^tA||A| = |{}^tAA| = |I_n| = 1$$

より、$|A| = \pm 1$ であることがわかる。行列式が 1 であるユニタリ行列、直交行列を、それぞれ**特殊ユニタリ行列**、**特殊直交行列**と言う。

例 65. (C1) load("gson.mac");

```
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
(D1) gson.mac
(C2) A : ratsimp(gson(matrix([1,2,3],[4,5,6],[7,8,0])));
```

(D2) $\begin{pmatrix} \frac{1}{\sqrt{66}} & \frac{3\sqrt{11}}{11} & -\frac{\sqrt{2}}{2\sqrt{3}} \\ \frac{4}{\sqrt{66}} & \frac{\sqrt{11}}{11} & \frac{\sqrt{2}}{\sqrt{3}} \\ \frac{7}{\sqrt{66}} & -\frac{\sqrt{11}}{11} & -\frac{\sqrt{2}}{2\sqrt{3}} \end{pmatrix}$

```
(C3) ratsimp(determinant(A)^2);
(D3) 1
(C4) B : ratsimp(gson(matrix([1,%i,2-%i],[3+%i, 4*%i, 2],[-1+%i, 2-3*%i, %i])));
```

(D4) $\begin{pmatrix} \frac{1}{\sqrt{13}} & -\frac{\sqrt{13}i-\sqrt{13}}{13\sqrt{193}} & -\frac{67\sqrt{193}i-125\sqrt{193}}{193\sqrt{113}} \\ \frac{i+3}{\sqrt{13}} & \frac{11\sqrt{13}i+17\sqrt{13}}{13\sqrt{193}} & \frac{5\sqrt{193}i-38\sqrt{193}}{193\sqrt{113}} \\ \frac{i-1}{\sqrt{13}} & -\frac{24\sqrt{13}i-39\sqrt{13}}{13\sqrt{193}} & -\frac{15\sqrt{193}i-\sqrt{193}}{193\sqrt{113}} \end{pmatrix}$

```
(C5) detb : ratsimp(determinant(B));
```

(D5) $-\frac{7i-8}{\sqrt{113}}$

```
(C6) ratsimp(conjugate(detb)*detb);
(D6) 1
```

2 次の正方行列が直交行列になる条件を考えよう。

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

が直交行列であるとすると、${}^tAA = I_2$ より、

$$\begin{pmatrix} a & c \\ b & d \end{pmatrix}\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} a^2+c^2 & ab+cd \\ ba+dc & b^2+d^2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

を得る。この条件を読みとくと、

$$A = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \quad \text{または} \quad \begin{pmatrix} \cos\eta & \sin\eta \\ \sin\eta & -\cos\eta \end{pmatrix}$$

のいずれかの形に書けることがわかる。それぞれが、行列式 1 および -1 に対応している。幾何学的には前者は平面の角 θ の回転を表す。後者は 平面の反転をあらわす行列

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

と前者のタイプの行列の積として表すことができる。

例 66. (C1) `A : matrix([cos(th), -sin(th)], [sin(th), cos(th)]);`

(D1) $\begin{pmatrix} \cos \mathrm{th} & -\sin \mathrm{th} \\ \sin \mathrm{th} & \cos \mathrm{th} \end{pmatrix}$

(C2) `transpose(A) . A;`

(D2) $\begin{pmatrix} \sin^2 \mathrm{th} + \cos^2 \mathrm{th} & 0 \\ 0 & \sin^2 \mathrm{th} + \cos^2 \mathrm{th} \end{pmatrix}$

(C3) `trigsimp(D2);`

(D3) $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

(C4) `trigsimp(determinant(A));`

(D4) 1

(C5) `B : matrix([cos(phi), sin(phi)], [sin(phi), -cos(phi)]);`

(D5) $\begin{pmatrix} \cos \varphi & \sin \varphi \\ \sin \varphi & -\cos \varphi \end{pmatrix}$

(C6) `transpose(B) . B;`

(D6) $\begin{pmatrix} \sin^2 \varphi + \cos^2 \varphi & 0 \\ 0 & \sin^2 \varphi + \cos^2 \varphi \end{pmatrix}$

(C7) `trigsimp(D6);`

(D7) $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

(C8) `trigsimp(determinant(B));`

(D8) -1

(C9) `T : matrix([0, 1], [1, 0]);`

(D9) $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

(C10) `C : B . T;`

(D10) $\begin{pmatrix} \sin \varphi & \cos \varphi \\ -\cos \varphi & \sin \varphi \end{pmatrix}$

(C11) `trigsimp(transpose(C) . C);`

(D11) $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

(C12) `trigsimp(determinant(C));`

(D12) 1

上の計算例で使った **maxima** のコマンド **trigsimp** は、三角関数を含む式の表現を簡約化するコマンドである。 □

正定値でない内積を持つ内積空間の例として、$\mathbb{R}^n$ にローレンツ内積 η を入れたもの、すなわち**ローレンツ空間** $\mathbb{R}^{1,n-1} = (\mathbb{R}^n, \eta)$ を考えよう。($n = 4$ のときはミンコフスキー空間。) ローレンツ内積 η の標準基底 $\mathfrak{E}$ に関する計量行列 J は、

$$J = \begin{pmatrix} 1 & & & 0 \\ & -1 & & \\ & & \ddots & \\ 0 & & & -1 \end{pmatrix}$$

であった。$\mathbb{R}^{1,n-1}$ 上の線形変換 f でローレンツ内積 η を保つものを考える。そのような f の基底対 $(\mathfrak{E}, \mathfrak{E})$ に関する表現行列を F とすると、先に述べた議論から F は条件

$$ {}^tFJF = J $$

を満足する。一般にこの条件を満たす行列を**ローレンツ行列**と言う。基底対 $(\mathfrak{E}, \mathfrak{E})$ に関する表現行列がローレンツ行列であるような $\mathbb{R}^{1,n-1}$ 上の線形変換をローレンツ変換と言う。

$\mathbb{R}^{1,n-1}$ の基底 $\mathfrak{A} = (\mathbf{a}_1, \dots, \mathbf{a}_n)$ が条件

$$ \eta(\mathbf{a}_i, \mathbf{a}_j) = \begin{cases} 1 & (i = j = 1) \\ -1 & (i = j \neq 1) \\ 0 & (i \neq j) \end{cases} $$

を満たすとき、$\mathfrak{A}$ を $\mathbb{R}^{1,n-1}$ の**ローレンツ基底**と言う。これは、正定値内積空間の正規直交基底に対応する概念である。いま $\mathbb{R}^{1,n-1}$ の 2 通りの基底 $\mathfrak{A} = (\mathbf{a}_1, \dots, \mathbf{a}_n)$、$\mathfrak{B} = (\mathbf{b}_1, \dots, \mathbf{b}_n)$ がともにローレンツ基底であるとしよう。このとき、基底の変換 $\mathfrak{A} \to \mathfrak{B}$ にともなう変換行列を P とすれば、直交行列のときと同様の議論で、条件

$$ {}^tPJP = J $$

が成り立つことがわかる。すなわち P はローレンツ行列になる。この基底の変換にともなう座標の変換公式

$$ \mathbf{v}_{\mathfrak{B}} = P\mathbf{v}_{\mathfrak{A}} $$

を**ローレンツ座標変換**と言う。F を任意のローレンツ行列としよう。定義より、

$$ |F|^2|J| = |{}^tF||J||F| = |{}^tFJF| = |J| $$

であり、$|J| \neq 0$ であるから、

$$ |F|^2 = 1 $$

すなわち、$|F| = \pm 1$ を得る。

2 次のローレンツ行列を考えよう。

$$ A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} $$

がローレンツ行列であるとすると、定義から

$$ \begin{pmatrix} a & c \\ b & d \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} $$

が成り立つ。すなわち、

$$ \begin{pmatrix} a^2 - c^2 & ab - cd \\ ab - cd & b^2 - d^2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} $$

が成り立つ。これより、A は

$$ \begin{pmatrix} \cosh\mu & \sinh\mu \\ \sinh\mu & \cosh\mu \end{pmatrix}, \quad \begin{pmatrix} \cosh\mu & -\sinh\mu \\ \sinh\mu & -\cosh\mu \end{pmatrix}, \quad \begin{pmatrix} -\cosh\mu & -\sinh\mu \\ \sinh\mu & \cosh\mu \end{pmatrix}, \quad \begin{pmatrix} -\cosh\mu & \sinh\mu \\ \sinh\mu & -\cosh\mu \end{pmatrix} $$

のいずれかの形に書けることがわかる。$v = \tanh\mu$ とすると、

$$ \cosh\mu = \frac{1}{\sqrt{1 - v^2}}, \quad \sinh\mu = \frac{v}{\sqrt{1 - v^2}} $$

であるから、$v \quad (|v|<1)$ を任意パラメータとして、

$$\begin{pmatrix} \frac{1}{\sqrt{1-v^2}} & \frac{v}{\sqrt{1-v^2}} \\ \frac{v}{\sqrt{1-v^2}} & \frac{1}{\sqrt{1-v^2}} \end{pmatrix}, \quad \begin{pmatrix} \frac{1}{\sqrt{1-v^2}} & \frac{-v}{\sqrt{1-v^2}} \\ \frac{v}{\sqrt{1-v^2}} & \frac{-1}{\sqrt{1-v^2}} \end{pmatrix}, \quad \begin{pmatrix} \frac{-1}{\sqrt{1-v^2}} & \frac{-v}{\sqrt{1-v^2}} \\ \frac{v}{\sqrt{1-v^2}} & \frac{1}{\sqrt{1-v^2}} \end{pmatrix}, \quad \begin{pmatrix} \frac{-1}{\sqrt{1-v^2}} & \frac{v}{\sqrt{1-v^2}} \\ \frac{v}{\sqrt{1-v^2}} & \frac{-1}{\sqrt{1-v^2}} \end{pmatrix}$$

のいずれかの形と言ってもいい。

例 67. (C1) `load("metrics.mac");`

```
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
```

(D1) metrics.mac

(C2) `A : matrix([cosh(mu), sinh(mu)], [sinh(mu), cosh(mu)]);`

(D2) $\begin{pmatrix} \cosh\mu & \sinh\mu \\ \sinh\mu & \cosh\mu \end{pmatrix}$

(C3) `J : matrix([1, 0], [0, -1]);`

(D3) $\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

(C4) `transpose(A) . J . A;`

(D4) $\begin{pmatrix} \cosh^2\mu - \sinh^2\mu & 0 \\ 0 & \sinh^2\mu - \cosh^2\mu \end{pmatrix}$

(C5) `trigsimp(D4);`

(D5) $\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

(C6) `v0 : transpose(matrix([t, x]));`

(D6) $\begin{pmatrix} t \\ x \end{pmatrix}$

(C7) `v1 : A . v0;`

(D7) $\begin{pmatrix} \sinh\mu x + \cosh\mu t \\ \cosh\mu x + \sinh\mu t \end{pmatrix}$

(C8) `geom(v0, v0, J);`

(D8) $t^2 - x^2$

(C9) `geom(v1, v1, J);`

(D9) $\left(\sinh^2\mu - \cosh^2\mu\right) x^2 + \left(\cosh^2\mu - \sinh^2\mu\right) t^2$

(C10) `trigsimp(D9);`

(D10) $t^2 - x^2$

(C11) `B : matrix([1/sqrt(1-v^2), v/sqrt(1-v^2)],[v/sqrt(1-v^2), 1/sqrt(1-v^2)]);`

(D11) $\begin{pmatrix} \frac{1}{\sqrt{1-v^2}} & \frac{v}{\sqrt{1-v^2}} \\ \frac{v}{\sqrt{1-v^2}} & \frac{1}{\sqrt{1-v^2}} \end{pmatrix}$

(C12) `transpose(B) . J . B;`

(D12) $\begin{pmatrix} \frac{1}{1-v^2} - \frac{v^2}{1-v^2} & 0 \\ 0 & \frac{v^2}{1-v^2} - \frac{1}{1-v^2} \end{pmatrix}$

(C13) `ratsimp(D12);`

(D13) $\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

(C14) `v2 : B . v0;`

(D14) $\begin{pmatrix} \frac{vx}{\sqrt{1-v^2}} + \frac{t}{\sqrt{1-v^2}} \\ \frac{x}{\sqrt{1-v^2}} + \frac{tv}{\sqrt{1-v^2}} \end{pmatrix}$

```
(C15) geom(v2 , v2, J);
```

(D15) $t^2 - x^2$

行列 F がユニタリ行列、(直交行列、ローレンツ行列) ならば、その逆行列 $F^{-1} = \overline{{}^tF}$ もユニタリ行列 (直交行列、ローレンツ行列) である。また、F, G がともに同じ型のユニタリ行列 (直交行列、ローレンツ行列) ならば、積 FG もまたユニタリ行列 (直交行列、ローレンツ行列) である。このことを、n 次ユニタリ行列 (直交行列、ローレンツ行列) 全体の集合は群をなすと言う。n 次ユニタリ行列全体のなす群を $U(n)$ と書いて、n 次ユニタリ群と言う。同様に、n 次の直交行列、ローレンツ行列全体のなす群をそれぞれ $O(n)$、$O(1, n-1)$ と書いて n 次直交群、ローレンツ群と言う。特に、特殊ユニタリ行列全体、および特殊直交行列全体の集合も群をなすが、それらをそれぞれ $SU(n)$、$SO(n)$ と書いて、n 次特殊ユニタリ群、特殊直交群と言う。

10. 固有値と固有ベクトル

10.1. **固有値と固有方程式.** V を $\mathbb{K}$ 上のベクトル空間、f を V 上の線形変換とする。V のあるベクトル $\mathbf{v}(\neq \mathbf{0})$ に対して、あるスカラー $\lambda \in \mathbb{K}$ が存在して、

$$f(\mathbf{v}) = \lambda\mathbf{v}$$

となるとき、λ を f の**固有値**と言い、また $\mathbf{v}$ を λ に属する**固有ベクトル**と言う。

V の部分集合 V_λ を

$$V_\lambda := \{\mathbf{v} \in V \mid f(\mathbf{v}) = \lambda\mathbf{v}\}$$

により定義する。このとき、

$$\begin{aligned}
&\mathbf{v}, \mathbf{w} \in V_\lambda, \alpha, \beta \in \mathbb{K} \\
&\Rightarrow \quad f(\alpha\mathbf{v} + \beta\mathbf{w}) = \alpha f(\mathbf{v}) + \beta f(\mathbf{w}) = \alpha(\lambda\mathbf{v}) + \beta(\lambda\mathbf{w}) = \lambda(\alpha\mathbf{v} + \beta\mathbf{w}) \\
&\Rightarrow \alpha\mathbf{v} + \beta\mathbf{w} \in V_\lambda
\end{aligned}$$

となるから、V_λ は V の部分ベクトル空間である。V を λ に属する**固有空間**と言う。

V は有限次元 $(n = \dim V)$ であるとし、基底 $\mathfrak{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_n)$ が与えられているとする。このとき、$f(\mathbf{v}) = \lambda\mathbf{v}$ という条件式を、基底 $\mathfrak{A}$ に関する成分を使って表現しよう。それは、f の基底対 $(\mathfrak{A}, \mathfrak{A})$ に関する表現行列を A とすると、

$$A\mathbf{v}_{\mathfrak{A}} = \lambda\mathbf{v}_{\mathfrak{A}}$$

と書ける。ここで、$\mathbf{v}_{\mathfrak{A}} \in \mathbb{K}^n$ は $\mathbf{v}$ の $\mathfrak{A}$ に関する成分である。

$$\langle\lambda\rangle := \{\mathbf{a} \in \mathbb{K}^n \mid A\mathbf{a} = \lambda\mathbf{a}\}$$

とすると、これは V_λ に 1 対 1 に対応する $\mathbb{K}^n$ の部分ベクトル空間であり、V_λ を基底 $\mathfrak{A}$ に関する成分で表現したものになっている。$\langle\lambda\rangle$ を λ に属する $\mathbb{K}^n$ の固有空間と言う。

$\langle\lambda\rangle \neq \{\mathbf{0}\}$ (すなわち、$V_\lambda \neq \{\mathbf{0}\}$) という条件は、$\lambda$ に属する固有ベクトルが少なくとも一つ存在するということを意味し、 が固有値であるということを意味する。これは、$\mathbf{a} \in \mathbb{K}^n$ についての同次連立一次方程式

$$(A - \lambda I_n)\mathbf{a} = \mathbf{0}$$

が非自明解を持つという条件に他ならない。同次連立一次方程式の理論から、この方程式が非自明解をもつための必要十分条件は $\mathrm{rank}(A - \lambda I_n) < n$、あるいは、行列式の言葉で、

$$|A - \lambda I_n| = 0 \tag{5}$$

と表現される。逆に、これを λ に関する条件式と見ると、λ がこの方程式の解ならば、λ は f の固有値であり、λ に属する固有ベクトルが存在することになる。方程式 (5) を行列 A の**固有方程式**、あるいは、線形変換 f の (基底 $\mathfrak{A}$ に関する) 固有方程式と言う。ここで、次のことを注意しておく。f の固有値は f のみによって決まり、基底 $\mathfrak{A}$、したがって表現行列 A によらない。

与えられた正方行列 A に対して、その固有方程式を求める **maxima** のコマンドは **charpoly** である。行列 A の変数 x に関する固有方程式を “charpoly(A, x);” とタイプして求めることができる。

例 68. (C1) `A : matrix([2, -1, 0], [3, 1, -2], [0, 1, 1]);`

(D1) $\begin{pmatrix} 2 & -1 & 0 \\ 3 & 1 & -2 \\ 0 & 1 & 1 \end{pmatrix}$

(C2) `charpoly(A , x);`
(D2) $\left((1-x)^2+2\right)(2-x)+3(1-x)$
(C3) `expand(D2);`
(D3) $-x^3+4x^2-10x+9$

固有値を求めるには、もちろん、charpoly で求めた固有方程式を **solve** で解いてもよいのであるが、**maxima** にはそれ用の専用コマンド **eigenvalues** が用意されている。eigenvalues は、正方行列を引数として取り、リストのリストを値として返す。返り値の最初の部分リストは固有値のリストで、次の部分リストは各固有値に対応する固有方程式の解としての重複度のリストである。

例 69. (C1) `A : matrix([2, 1, -1], [1, 1, 0], [-1, 0, 1]);`
(D1) $\begin{pmatrix} 2 & 1 & -1 \\ 1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}$
(C2) `eq1 : charpoly(A, x);`
(D2) $2x+(1-x)^2(2-x)-2$
(C3) `solve([eq1], [x]);`
(D3) $[x=3, x=1, x=0]$
(C4) `eigenvalues(A);`

```
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
```

(D4) $[[3,1,0],[1,1,1]]$
この例では、$3,1,0$ が行列 A の固有値であり、それぞれの重複度が $1,1,1$ であることを示している。 □

固有値が求まったら、次は各固有値に属する固有ベクトルを求めることになるが、それはもちろん、同次連立一次方程式

$$(A-\lambda I_n)\mathbf{a}=\mathbf{0}$$

を解くことにより求まる。その解は固有空間 $\langle\lambda\rangle$ の次元に相当する個数の任意パラメータを含む形でもとまる。

例 70. 例 69 の行列 A の固有値 $3,1,0$ に属する固有ベクトルをそれぞれ求めてみよう。
(C1) `A : matrix([2, 1, -1], [1, 1, 0], [-1, 0, 1]);`
(D1) $\begin{pmatrix} 2 & 1 & -1 \\ 1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}$
(C2) `xx : transpose(matrix([x, y, z]));`
(D2) $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$
(C3) `eq3 : (A - 3*ident(3)) . xx;`
(D3) $\begin{pmatrix} -z+y-x \\ x-2y \\ -2z-x \end{pmatrix}$
(C4) `solve([eq3[1][1], eq3[2][1], eq3[3][1]], [x, y, z]);`

```
Dependent equations eliminated:  (3)
```

(D4) $[[x = -2\%\mathrm{R1}, y = -\%\mathrm{R1}, z = \%\mathrm{R1}]]$

```
(C5) eq1 : (A - 1*ident(3)) . xx;
```

(D5) $$\begin{pmatrix} -z + y + x \\ x \\ -x \end{pmatrix}$$

```
(C6) solve([eq1[1][1], eq1[2][1], eq1[3][1]], [x, y, z]);
Dependent equations eliminated:  (3)
```

(D6) $[[x = 0, y = \%\mathrm{R2}, z = \%\mathrm{R2}]]$

```
(C7) eq0 : A . xx;
```

(D7) $$\begin{pmatrix} -z + y + 2x \\ y + x \\ z - x \end{pmatrix}$$

```
(C8) solve([eq0[1][1], eq0[2][1], eq0[3][1]], [x, y, z]);
Dependent equations eliminated:  (3)
```

(D8) $[[x = \%\mathrm{R3}, y = -\%\mathrm{R3}, z = \%\mathrm{R3}]]$

```
(C9) xx3 : transpose(matrix([-2*r, -r, r]));
```

(D9) $$\begin{pmatrix} -2r \\ -r \\ r \end{pmatrix}$$

```
(C10) A . xx3 - 3 * xx3;
```

(D10) $$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

```
(C11) xx1 : transpose(matrix([0, s, s]));
```

(D11) $$\begin{pmatrix} 0 \\ s \\ s \end{pmatrix}$$

```
(C12) A . xx1 - 1 * xx1;
```

(D12) $$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

```
(C13) xx0 : transpose(matrix([t, -t, t]));
```

(D13) $$\begin{pmatrix} t \\ -t \\ t \end{pmatrix}$$

```
(C14) A . xx0 - 0 * xx0;
```

(D14) $$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

実は **maxima** には、直接固有ベクトルを求めてくれるコマンドが用意されている。コマンド **eigenvactors** は正方行列を引数に取り、リストのリストを値として返す。返り値の最初の部分リストは eigenvalues の返り値と同じもので、固有値のリストとその重複度のリストからなるリストのりすとである。第 2 以降の部分リストが各固有値に対応する固有ベクトルを表す。固有ベクトルは任意パラメータに適当な値を代入して得られる数ベク

トルとして返される。eigenvectors を使用するには事前に “load(eigen)” によりライブラリを読み込んでおく必要がある。

例 71. (C1) load(eigen)

```
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
(D1) /usr/local/share/maxima/5.9.0/share/matrix/eigen.mac
(C2) A : matrix([2, 1, -1], [1, 1, 0], [-1, 0, 1]);
```

(D2) $\begin{pmatrix} 2 & 1 & -1 \\ 1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}$

(C3) eigenvectors(A);

(D3) $\left[\left[[3,1,0],[1,1,1]\right],\left[1,\frac{1}{2},-\frac{1}{2}\right],[0,1,1],[1,-1,1]\right]$

(C4) eigenvalues(A);

(D4) $[[3,1,0],[1,1,1]]$

ここで、最後の式は

$$\begin{pmatrix} 1 \\ \frac{1}{2} \\ -\frac{1}{2} \end{pmatrix}, \quad \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}, \quad \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix},$$

が、それぞれ、固有値 $3,1,0$ に対応する固有ベクトルであることを示している。 □

目的によっては、eigenvector の返り値であるリスト形式ではなくて、別の形式で答が欲しい場合があるだろう。特に、次小節で使うために、固有ベクトルを列ベクトルで表し、それを行列の形に並べたものを返すようなコマンドが欲しい。そのために、新しいコマンドを自作することにしよう。次のプログラムを “gson.mac” に追加しよう。

```
eigmat(A) := block([i,n,B,C],
  n : length(A),
  B : eigenvectors(A),
  C : zeromatrix(n, n),
  for i:1 thru n do
    C[i] : B[i+1],
  transpose(C)
)$
```

コマンド **eigmat** は引数の正方行列の固有ベクトルを eigenvectors で求めたものを列ベクトルにして、行列に並べたものを返す。

例 72. (C1) load("gson.mac");

```
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
(D1) gson.mac
(C2) A : matrix([2, 1, -1], [1, 1, 0], [-1, 0, 1]);
```

(D2) $\begin{pmatrix} 2 & 1 & -1 \\ 1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}$

```
(C3) eigmat(A);
```

(D3) $\begin{pmatrix} 1 & 0 & 1 \\ \frac{1}{2} & 1 & -1 \\ -\frac{1}{2} & 1 & 1 \end{pmatrix}$

一般に、n 次正方行列の固有方程式は n 次の多項式であるから、$n \geq 5$ ならば当然解けない可能性が大きい。

10.2. **対称行列の対角化.** 与えられた正方行列 A に対して、適当な正則行列 P を選んで、

$$P^{-1}AP = \begin{pmatrix} \lambda_1 & & & 0 \\ & \lambda_1 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{pmatrix}$$

の形にできるとき、行列 A は P により**対角化可能**であると言う。

上の状況が成り立っているとき、$P = (\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n)$ を P の列ベクトルへの分解とすれば、上の式の両辺に P を左から掛けて、

$$AP = P \begin{pmatrix} \lambda_1 & & & 0 \\ & \lambda_1 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{pmatrix}$$

を得る。すなわち、

$$(A\mathbf{p}_1, A\mathbf{p}_2, \ldots, A\mathbf{p}_n) = (\lambda_1\mathbf{p}_1, \lambda_2\mathbf{p}_2, \ldots, \lambda_n\mathbf{p}_n)$$

であるが、これは、各 $i \in \{1, \ldots, n\}$ に対して、

$$A\mathbf{p}_i = \lambda_i\mathbf{p}_i$$

が成立することを意味する。したがって、$\lambda_1, \ldots, \lambda_n$ はすべて A の固有値であり、$\mathbf{p}_1, \ldots, \mathbf{p}_n$ はそれぞれの固有値に属する A の固有ベクトルとなる。P が正則であると言う条件から、$(\mathbf{p}_1, \ldots, \mathbf{p}_n)$ は $\mathbb{K}^n$ の基底である。

V を $\mathbb{K}$ 上の n 次元ベクトル空間とし、$f\colon V \to V$ を V 上の線形変換とする。V の一つの基底 $\mathfrak{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_n)$ が与えられており、f の基底対 $(\mathfrak{A}, \mathfrak{A})$ に対する表現行列が A であるとしよう。いま、上のように A はある正則行列 P により対角化可能であるとする。基底 $\mathfrak{A}$ による V と $\mathbb{K}^n$ の同一視は、各 $\mathbf{a}_i \in V$ に $\mathbf{e}_i \in \mathbb{K}^n$ を対応させることにより得られている。この対応の逆対応により、各ベクトル $\mathbf{p}_i \in \mathbb{K}^n$ に対応するベクトル $\mathbf{v}_i \in V$ を考えよう。$(\mathbf{p}_1, \ldots, \mathbf{p}_n)$ が $\mathbb{K}^n$ の基底であったから、$\mathfrak{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_n)$ は、それに対応する V の基底となる。V における基底の変換 $\mathfrak{A} \to \mathfrak{V}$ は、$\mathbb{K}^n$ における基底

の変換 $\mathfrak{E} = (\mathbf{e}_1, \ldots, \mathbf{e}_n) \to P = (\mathbf{p}_1, \ldots, \mathbf{p}_n)$ に対応しており、その変換行列はともに、$P^{-1}\mathfrak{E} = P^{-1}I_n = P^{-1}$ で与えられることがわかう。このことから、f の基底対 $(\mathfrak{V}, \mathfrak{V})$ に関する表現行列は、変換公式にしたがい、

$$P^{-1}A(P^{-1})^{-1} = P^{-1}AP = \begin{pmatrix} \lambda_1 & & & 0 \\ & \lambda_1 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{pmatrix}$$

で与えられることがわかる。つまり、線形変換 f のある基底対 $(\mathfrak{A}, \mathfrak{A})$ に関する表現行列 A が**対角化可能**であるということは、f の固有ベクトル $\mathbf{v}_1, \ldots, \mathbf{v}_n$ から構成される基底 $\mathfrak{V}$ が存在するということを意味する。逆に、そのような基底 $\mathfrak{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_n)$ が存在すれば、

$$f(\mathbf{v}_i) = \lambda_i \mathbf{v}_i, \quad \forall i \in \{1, \ldots, n\}$$

より、 f の 基底対 $(\mathfrak{V}, \mathfrak{V})$ に関する表現行列は

$$\begin{pmatrix} \lambda_1 & & & 0 \\ & \lambda_1 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{pmatrix}$$

で与えられることがわかる。

そこで、f が対角化可能であるということを、固有ベクトルからなる基底が存在することと定義する。これは、特別な基底の取り方にはよらない、f のみに依存する性質である。

さて、(V, g) を $\mathbb{R}$ 上の正定値内積空間としよう。V 上の線形変換 $f\colon V \to V$ が**対称変換**であるとは、条件

$$g(f(\mathbf{v}), \mathbf{w}) = g(\mathbf{v}, f(\mathbf{w})), \forall \mathbf{v}, \forall \mathbf{w} \in V$$

成り立つことである。今、$\mathfrak{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_n)$ を V の正規直交基底とし、 A を f の 基底対 $(\mathfrak{A}, \mathfrak{A})$ に関する表現行列としよう。すると上式より、

$$g(f(\mathbf{a}_i), \mathbf{a}_j) = g(\mathbf{a}_i, f(\mathbf{a}_j)), \quad \forall i, \forall j$$

となる。これを成分レベルで表現すると

$$\langle A\mathbf{e}_i, \mathbf{e}_j \rangle = \langle \mathbf{e}_i, A\mathbf{e}_j \rangle, \quad \forall i, \forall j$$

となる。一方、常に

$$\langle A\mathbf{e}_i, \mathbf{e}_j \rangle = \langle \mathbf{e}_i, {}^tA\mathbf{e}_j \rangle, \quad \forall i, \forall j$$

であり、上と比較することにより、

$$A = {}^tA$$

を得る。すなわち、対称変換 f の任意の正規直交基底に関する表現行列は**対称行列**である。λ をこの対称行列 A の固有方程式の解の一つとしよう。λ は一般には複素数値と考えられるのであるが、f が対称変換、すなわち A が対称行列ならば固有方程式の解は必ず実数になる。なぜならば、$\mathbf{v}$ を $\mathbb{C}^n$ における A の λ に属する固有ベクトルとすると、

$$\overline{\lambda}|\mathbf{v}|^2 = \langle \lambda\mathbf{v}, \mathbf{v}\rangle = \langle A\mathbf{v}, \mathbf{v}\rangle = \langle \mathbf{v}, A\mathbf{v}\rangle = \lambda\langle \mathbf{v}, \mathbf{v}\rangle = \lambda|\mathbf{v}|^2$$

となって、$\overline{\lambda} = \lambda$ を得るからである。また、$\lambda \neq \mu$ を共に f の固有値とし、$\mathbf{v}, \mathbf{w}$ をそれぞれ λ, μ に属する固有ベクトルとすると、

$$\lambda g(\mathbf{v}, \mathbf{w}) = g(\lambda\mathbf{v}, \mathbf{w}) = g(f(\mathbf{v}), \mathbf{w}) = g(\mathbf{v}, f(\mathbf{w})) = g(\mathbf{v}, \mu\mathbf{w}) = \mu g(\mathbf{v}, \mathbf{w})$$

より、$g(\mathbf{v}, \mathbf{w}) = 0$ を得る。すなわち、f の異なる固有値に属する固有ベクトルは互いに直交する。
さて、この小節の目標は対称変換 (行列) の対角化可能性を示すことであるが、上の下線部で述べたことの他に、もう一つ必要な事柄がある。それは少し難しいので、詳細については線形代数学の本に譲ることにして、ここでは、事実として認めることにしよう。f の固有値 λ の重複度が k であるとき、λ に属する固有空間の次元も k である。($\dim V_\lambda = \dim\langle\lambda\rangle = k$.)

以上の準備の下に、具体的な f の、すなわち対称行列 A の対角化の手順を考えよう。
(STEP1)　まず、A の固有方程式 $|A - \lambda I_n| = 0$ を解き、相異なる固有値 $\lambda_1, \ldots, \lambda_s$ とそれら各々の重複度 $m_1, \ldots, m_s$ を求める。ただしここで、$m_1 + \cdots + m_s = n$ である。
(STEP2)　各固有値 $\lambda_i, i \in \{1, \ldots, s\}$ に対して、λ_i に属する A の固有空間 $\langle\lambda_i\rangle$ の正規直交基底を必要ならば Gram-Schmidt の正規直交化法などを使って作り、それを $\mathbf{v}_i^{(1)}, \ldots, \mathbf{v}_i^{(m_i)}$ とする。
(STEP3)　STEP2 で求めた固有ベクトルを

$$P = (\mathbf{v}_1^{(1)}, \ldots, \mathbf{v}_1^{(m_1)}, \mathbf{v}_2^{(1)}, \ldots, \mathbf{v}_2^{(m_2)}, \ldots, \mathbf{v}_s^{(1)}, \ldots, \mathbf{v}_s^{(m_s)})$$

の順に並べる。すると、上述より P は $\mathbb{R}^n$ の正規直交基底である。各 $\mathbf{v}_i^{(j)}, i \in \{1, \ldots, s\}, j \in \{1, \ldots, m_i\}$ は A の λ_i に属する固有ベクトルであるから、最初に議論したように、

$$P^{-1}AP = \begin{pmatrix} \lambda_1 & & & & & & & & 0 \\ & \ddots & & & & & & & \\ & & \lambda_1 & & & & & & \\ & & & \lambda_2 & & & & & \\ & & & & \ddots & & & & \\ & & & & & \lambda_2 & & & \\ & & & & & & \ddots & & \\ & & & & & & & \lambda_s & \\ & & & & & & & & \ddots \\ 0 & & & & & & & & \lambda_s \end{pmatrix}$$

となり、A は P によって対角化されることが結論される。また、P は正規直交基底であるから、行列としては直交行列である。

以上をまとめよう。任意の対称行列 A は、ある直交行列 P により対角化される。出てくる対角行列の各対角成分は A の固有値である。また直交行列 P は A の固有ベクトルからなる正規直交基底を行列と見なしたものである。

注意 7. 対称変換 f を対角化するということは、上の直交行列 P を求めることである。本質的なのは A ではなく、P と対角行列 $\mathrm{diag}(\lambda_1, \ldots, \lambda_s)$ の方である。

さて、次に係数体を $\mathbb{C}$ にして考える。$\mathbb{C}$ 上の正定値内積空間 (V, g) と、V 上の線形変換 $f\colon V \to V$ についても $\mathbb{R}$ 上での場合とほぼ同様の議論ができる。f が**エルミート変換**であるとは、条件

$$g(f(\mathbf{v}), \mathbf{w}) = g(\mathbf{v}, f(\mathbf{w})), \quad \forall \mathbf{v}, \forall \mathbf{w} \in V$$

が成り立つことである。このとき、V の任意の正規直交基底 $\mathfrak{A}$ に対して、f の基底対 $(\mathfrak{A}, \mathfrak{A})$ に関する表現行列が、条件

$$A = \overline{{}^tA}$$

を満たすことは容易にチェックできる。すなわち、エルミート変換の任意の正規直交基底に関する表現行列はエルミート行列である。この場合の、上述に対応する主張とは次の通りである。エルミート変換 f (エルミート行列 A) は、あるユニタリ行列 P によって対角化される。すなわち、

$$P^{-1}AP = \begin{pmatrix} \lambda_1 & & & 0 \\ & \lambda_1 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{pmatrix}$$

が成り立つ。この場合も、固有値 $\lambda_1, \ldots, \lambda_n$ はすべて実数である。詳細については、最早ここでは述べない。線形代数学の本を参照して欲しい。以下の例としては取り上げることにしよう。

例 73. 実対称行列

$$A = \begin{pmatrix} -\frac{21}{22} & \frac{2}{11} & \frac{7}{22} \\ \frac{2}{11} & -\frac{3}{11} & \frac{14}{11} \\ \frac{7}{22} & \frac{14}{11} & \frac{27}{22} \end{pmatrix}$$

を上の手順に従って対角化しよう。

```
(C1) load("gson.mac");
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
(D1) gson.mac
(C2) A : matrix([-21/22, 2/11, 7/22], [2/11, -3/11, 14/11],[7/22, 14/11, 27/22]);
```

(D2) $\begin{pmatrix} -\frac{21}{22} & \frac{2}{11} & \frac{7}{22} \\ \frac{2}{11} & -\frac{3}{11} & \frac{14}{11} \\ \frac{7}{22} & \frac{14}{11} & \frac{27}{22} \end{pmatrix}$

```
(C3) I : ident(3);
```

(D3) $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

(C4) `ratsimp(determinant(A - x * I));`
(D4) $-x^3+3x+2$
(C5) `chply : ratsimp(expand(charpoly(A, x)));`
(D5) $-x^3+3x+2$
(C6) `solve([chply], [x]);`
(D6) $[x=2, x=-1]$
(C7) `eigenvalues(A);`
(D7) $[[2,-1],[1,2]]$
(C8) `xx : transpose(matrix([s, t, u]));`
(D8) $\begin{pmatrix} s \\ t \\ u \end{pmatrix}$
(C9) `eq1 : (A - 2*I) . xx;`
(D9) $\begin{pmatrix} \frac{7u}{22}+\frac{2t}{11}-\frac{65s}{22} \\ \frac{14u}{11}-\frac{25t}{11}+\frac{2s}{11} \\ -\frac{17u}{22}+\frac{14t}{11}+\frac{7s}{22} \end{pmatrix}$
(C10) `solve([eq1[1][1], eq1[2][1], eq1[3][1]], [s, t, u]);`
Dependent equations eliminated: (3)
(D10) $\left[\left[s=\frac{\%R1}{7}, t=\frac{4\%R1}{7}, u=\%R1\right]\right]$
(C11) `eq2 : (A - (-1)*I) . xx;`
(D11) $\begin{pmatrix} \frac{7u}{22}+\frac{2t}{11}+\frac{s}{22} \\ \frac{14u}{11}+\frac{8t}{11}+\frac{2s}{11} \\ \frac{49u}{22}+\frac{14t}{11}+\frac{7s}{22} \end{pmatrix}$
(C12) `solve([eq2[1][1], eq2[2][1], eq2[3][1]], [s, t, u]);`
Dependent equations eliminated: (2 3)
(D12) $[[s=-4\%R3-7\%R2, t=\%R3, u=\%R2]]$
(C13) `eigenvectors(A);`
(D13) $\left[[[2,-1],[1,2]],[1,4,7],\left[1,0,-\frac{1}{7}\right],\left[0,1,-\frac{4}{7}\right]\right]$
(C14) `PP : eigmat(A);`
(D14) $\begin{pmatrix} 1 & 1 & 0 \\ 4 & 0 & 1 \\ 7 & -\frac{1}{7} & -\frac{4}{7} \end{pmatrix}$
(C15) `P : gson(PP);`
(D15) $\begin{pmatrix} \frac{1}{\sqrt{66}} & \frac{7}{5\sqrt{2}} & -\frac{52}{5^2\sqrt{33}} \\ \frac{4}{\sqrt{66}} & 0 & \frac{5}{\sqrt{33}} \\ \frac{7}{\sqrt{66}} & -\frac{7}{57\sqrt{2}} & -\frac{527}{5^2\sqrt{33}} \end{pmatrix}$
(C16) `D : ratsimp(P^^(-1) . A . P);`
(D16) $\begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$

この例では、固有値 -1 は重解であり、それに属する固有空間は 2 次元である。行列 A は直交行列 P、すなわち単位ベクトルからなる固有ベクトルたちを並べたもので対角化される。 □

上のようにすると、途中の経過がわかるが、手っ取り早く結果のみ欲しい場合のために、専用のコマンドを作成しよう。次のプログラムを “gson.mac” に追加する。

```
dzmat(A) := ratsimp(gson(eigmat(A)))$

diagonalize(A) := block([B, C],
  B : dzmat(A),
  C : B^^(-1) . A . B,
  ratsimp(C)
)$
```

ここで、定義したコマンド **diagonalize** は引数の行列を対角化した行列を返す。すなわち、引数の行列の固有値を対角成分に持つ対角行列を値として返す。また、コマンド **dzmat** は引数の行列を対角化するのに使う直交行列を値として返す。

例 74.

(C1) load("gson.mac");
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
(D1) gson.mac
(C2) A : matrix([2, -1, 1], [-1, 3, 2], [1, 2, -1]);

(D2) $$\begin{pmatrix} 2 & -1 & 1 \\ -1 & 3 & 2 \\ 1 & 2 & -1 \end{pmatrix}$$

(C3) P : dzmat(A);

(D3) $$\begin{pmatrix} \frac{2}{\sqrt{9\sqrt{5}+25}} & \frac{2}{\sqrt{25-9\sqrt{5}}} & \frac{1}{\sqrt{11}} \\ \frac{\sqrt{5}+3}{2\sqrt{9\sqrt{5}+25}} & -\frac{\sqrt{5}-3}{2\sqrt{25-9\sqrt{5}}} & -\frac{3}{\sqrt{11}} \\ -\frac{3\sqrt{5}+5}{2\sqrt{9\sqrt{5}+25}} & \frac{3\sqrt{5}-5}{2\sqrt{25-9\sqrt{5}}} & -\frac{1}{\sqrt{11}} \end{pmatrix}$$

(C4) diagonalize(A);

(D4) $$\begin{pmatrix} -\frac{5}{\sqrt{5}} & 0 & 0 \\ 0 & \frac{5}{\sqrt{5}} & 0 \\ 0 & 0 & 4 \end{pmatrix}$$

(C15) ratsimp(P^^(-1) . A . P);

(D5) $$\begin{pmatrix} -\frac{5}{\sqrt{5}} & 0 & 0 \\ 0 & \frac{5}{\sqrt{5}} & 0 \\ 0 & 0 & 4 \end{pmatrix}$$

エルミート行列の例についても見ておこう。

例 75.

(C1) load("gson.mac");
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
(D1) gson.mac
(C2) A : matrix([2, 1+3*%i], [1-3*%i, -1]);

(D2) $\begin{pmatrix} 2 & 3i+1 \\ 1-3i & -1 \end{pmatrix}$

(C3) `chply : ratsimp(charpoly(A , x));`

(D3) $x^2 - x - 12$

(C4) `solve([chply],[x]);`

(D4) $[x = -3, x = 4]$

(C5) `I : ident(2);`

(D5) $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

(C6) `xx : transpose(matrix([s, t]));`

(D6) $\begin{pmatrix} s \\ t \end{pmatrix}$

(C7) `eq1 : (A - (-3)*I) . xx;`

(D7) $\begin{pmatrix} (3i+1)\,t + 5s \\ 2t + (1-3i)\,s \end{pmatrix}$

(C8) `solve([eq1[1][1], eq1[2][1]], [s, t]);`

`Dependent equations eliminated:  (2)`

(D8) $\left[\left[s = -\frac{3i\%\mathrm{R1}+\%\mathrm{R1}}{5}, t = \%\mathrm{R1}\right]\right]$

(C9) `eq2 : (A - 4*I) . xx;`

(D9) $\begin{pmatrix} (3i+1)\,t - 2s \\ (1-3i)\,s - 5t \end{pmatrix}$

(C10) `solve([eq2[1][1], eq2[2][1]], [s, t]);`

`Dependent equations eliminated:  (2)`

(D10) $\left[\left[s = \frac{3i\%\mathrm{R2}+\%\mathrm{R2}}{2}, t = \%\mathrm{R2}\right]\right]$

(C11) `eigenvalues(A);`

(D11) $[[-3, 4], [1, 1]]$

(C12) `eigenvectors(A);`

(D12) $\left[[[-3, 4], [1, 1]], \left[1, \frac{3i-1}{2}\right], \left[1, -\frac{3i-1}{5}\right]\right]$

(C13) `eigmat(A);`

(D13) $\begin{pmatrix} 1 & 1 \\ \frac{3i-1}{2} & -\frac{3i-1}{5} \end{pmatrix}$

(C14) `dzmat(A);`

(D14) $\begin{pmatrix} \frac{\sqrt{2}}{\sqrt{7}} & \frac{\sqrt{5}}{\sqrt{7}} \\ \frac{3\sqrt{2}i-\sqrt{2}}{2\sqrt{7}} & -\frac{3\sqrt{5}i-\sqrt{5}}{5\sqrt{7}} \end{pmatrix}$

(C15) `diagonalize(A);`

(D15) $\begin{pmatrix} -3 & 0 \\ 0 & 4 \end{pmatrix}$

10.3. **行列の巾乗再論.** 行列の対角化を利用して、対角化可能な行列の巾乗を簡単に計算することができる。正方行列 A が、正則行列 P によって、

$$P^{-1}AP = \begin{pmatrix} a_1 & & & 0 \\ & a_2 & & \\ & & \ddots & \\ 0 & & & a_n \end{pmatrix}$$

の形に対角化されたとしよう。このとき、

$$\begin{aligned} (P^{-1}AP)^k &= (P^{-1}AP)(P^{-1}AP)\cdots(P^{-1}AP) \\ &= P^{-1}A^kP \\ &= \begin{pmatrix} a_1^k & & & 0 \\ & a_2^k & & \\ & & \ddots & \\ 0 & & & a_k \end{pmatrix} \end{aligned}$$

を得る。したがって、

$$\begin{aligned} A^k &= P(P^{-1}A^kP)P^{-1} \\ &= P\begin{pmatrix} a_1^k & & & 0 \\ & a_2^k & & \\ & & \ddots & \\ 0 & & & a_k \end{pmatrix}P^{-1} \end{aligned}$$

を得る。これは、定義通りに巾乗を計算するのに比べて、k が大きいときは計算量を大幅に節約することができる計算法である。また、これにより任意の k に対する公式を与えていることも注目に値する。前小節で扱ったように、実対称行列あるいはエルミート行列は直交行列あるいはユニタリ行列で対角化可能であったから、これらの行列に対してはこの方法で巾乗を計算することができる。

例 76.

```
(C1) load("gson.mac");
Warning - you are redefining the MACSYMA function EIGENVALUES
Warning - you are redefining the MACSYMA function EIGENVECTORS
(D1) gson.mac
(C2) A : matrix([2, -1, 1], [-1, 3, 2], [1, 2, -1]);
```

(D2) $$\begin{pmatrix} 2 & -1 & 1 \\ -1 & 3 & 2 \\ 1 & 2 & -1 \end{pmatrix}$$

```
(C3) P : dzmat(A);
```

(D3) $$\begin{pmatrix} \frac{2}{\sqrt{9\sqrt{5}+25}} & \frac{2}{\sqrt{25-9\sqrt{5}}} & \frac{1}{\sqrt{11}} \\ \frac{\sqrt{5}+3}{2\sqrt{9\sqrt{5}+25}} & -\frac{\sqrt{5}-3}{2\sqrt{25-9\sqrt{5}}} & -\frac{3}{\sqrt{11}} \\ -\frac{3\sqrt{5}+5}{2\sqrt{9\sqrt{5}+25}} & \frac{3\sqrt{5}-5}{2\sqrt{25-9\sqrt{5}}} & -\frac{1}{\sqrt{11}} \end{pmatrix}$$

```
(C4) B : diagonalize(A);
```

(D4) $$\begin{pmatrix} -\frac{5}{\sqrt{5}} & 0 & 0 \\ 0 & \frac{5}{\sqrt{5}} & 0 \\ 0 & 0 & 4 \end{pmatrix}$$

```
(C5) B5 : B^^5;
```

(D5) $$\begin{pmatrix} -\frac{125}{\sqrt{5}} & 0 & 0 \\ 0 & \frac{125}{\sqrt{5}} & 0 \\ 0 & 0 & 1024 \end{pmatrix}$$

```
(C7) A5 : ratsimp(P . B5 . P^^(-1))
(C8) A5;
```

(D8) $$\begin{pmatrix} 134 & -277 & -59 \\ -277 & 831 & 302 \\ -59 & 302 & 59 \end{pmatrix}$$

11. ベクトル解析

11.1. **外積.** この節は付録的な節である。本格的にベクトル解析を **maxima** で扱おうと言う意図は全くない。ベクトル解析の初歩のところで出てくる空間ベクトルの**内積**, **外積**と、３種類の一階微分作用素 **grad**、**div**、**rot** に限って、これらを **maxima** で電卓的に扱う方法 (の一つを) 解説する。これらは、本格的な使用には耐えないかもしれないが、ちょっとした計算には活用できると期待する。

この節では、空間ベクトルを 3 次の実列ベクトルとして扱う。まず、この小節では２つの空間ベクトルの内積と外積を、２項演算子として扱う。f が、ある集合 U 上の２項演算であるというのは、f が $U \times U$ から 値の集合 V への写像であることに他ならない。しかし、(中置きの) 演算という言葉を使う場合は、値を通常の用に $f(a, b)$ のように書くのではなくて、afb のように表す。例えば、加法 “+” なども２項演算の例である。

空間ベクトルの内積は、すでに扱った 3 次元列ベクトル空間 $\mathbf{R}^3$ の標準内積と同じものであるが、ここでは関数 (**maxima** では コマンド **inprod** で計算できた) ではなく、２項演算として実現しよう。**maxima** で新たに中置きの２項演算子を定義するには、コマンド **infix** を使う。新たに f という名の演算子を定義したい場合には、“infix(f)” で f が新しい演算子であることを宣言し、その後 “$a\ f\ b$:= 関数定義式” で実体を定義する。ここで、a と b は仮引数である。それでは、空間ベクトルの内積を @∗ という記号で表すことにして、**maxima** で使うために定義しよう。次のプログラムを “vectanal.mac” と言うファイルに保存する。

```
infix("@*")$

v @* w := block([val],
  val : v[1][1] * w[1][1] + v[2][1] * w[2][1] + v[3][1] * w[3][1] ,
  return(val)
)$
```

早速、実験してみよう。

例 77.

(C1) `load("vectanal.mac");`

(D1) vectanal.mac

(C2) `v : matrix([a],[b],[c]);`

(D2) $\begin{pmatrix} a \\ b \\ c \end{pmatrix}$

(C3) `w : matrix([s],[t],[u]);`

(D3) $\begin{pmatrix} s \\ t \\ u \end{pmatrix}$

(C4) `v @* w;`

(D4) $cu + bt + as$

(C5) `v1 : transpose(matrix([1, -2, 3]));`

(D5) $\begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix}$

(C6) `w1 : transpose(matrix([4, 1, -2]));`

(D6) $\begin{pmatrix} 4 \\ 1 \\ -2 \end{pmatrix}$

(C7) `v1 @* w1;`

(D7) -4

内積というのは 2 つの空間ベクトルに一つの実数を対応させる 2 項演算子であるが、3 次元、したがって空間ベクトルの場合には、外積と言うもう一つの有用な 2 項演算子がある。外積は 2 つの空間ベクトルに一つの空間ベクトルを対応させるもので、その定義は座標を使って次のように定義される。

$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ b3 \end{pmatrix} = \begin{pmatrix} a_2b_3 - b_2a_3 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{pmatrix}.$$

ここで、“×” が数学で普通外積を表すのに使われる記号である。この記号は、キーボード上に存在しないので **maxima** で外積を表す記号としては、“@x” を使うことにしよう。内積のときのように定義する。以下のプログラムをファイル “vectanal.mac” に追加する。

```
infix("@x")$

v @x w := block([valvec],
  valvec : zeromatrix(3,1),
  valvec[1][1] : v[2][1] * w[3][1] - v[3][1] * w[2][1] ,
  valvec[2][1] : v[3][1] * w[1][1] - v[1][1] * w[3][1] ,
  valvec[3][1] : v[1][1] * w[2][1] - v[2][1] * w[1][1] ,
  return(valvec)
)$
```

これで、**maxima** で外積を使う準備ができた。

例 78.

(C1) `load("vectanal.mac");`

(D1) `vectanal.mac`

(C2) `v : transpose(matrix([a, b, c]));`

(D2) $\begin{pmatrix} a \\ b \\ c \end{pmatrix}$

(C3) `w : transpose(matrix([s, t, u]));`

(D3) $\begin{pmatrix} s \\ t \\ u \end{pmatrix}$

(C4) `v @x w;`

(D4) $\begin{pmatrix} bu-ct \\ cs-au \\ at-bs \end{pmatrix}$

(C5) `v1 : transpose(matrix([1, -2, 3]));`

(D5) $\begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix}$

(C6) `w1 : transpose(matrix([4, 1, -2]));`

(D6) $\begin{pmatrix} 4 \\ 1 \\ -2 \end{pmatrix}$

(C7) `v1 @x w1;`

(D7) $\begin{pmatrix} 1 \\ 14 \\ 9 \end{pmatrix}$

外積の幾何学的意味は次の通りである。$\mathbf{v}$、$\mathbf{w}$ を共にゼロベクトルではなく、互いに平行ではない 2 つの空間ベクトルとする。このとき $\mathbf{v}$ と $\mathbf{w}$ の外積 $\mathbf{v}\times\mathbf{w}$ は、$\mathbf{v}$ と $\mathbf{w}$ の両方に対して垂直な空間ベクトルで、その長さは $\mathbf{v}$ と $\mathbf{w}$ の張る平行四辺形の面積 $|\mathbf{v}||\mathbf{w}||\sin\theta|$ に等しい。ここで、θ は $\mathbf{v}$ と $\mathbf{w}$ の間の角とする。これらのことを **maxima** でチェックしてみよう。確かめればいいのは、$\mathbf{v}\bullet(\mathbf{v}\times\mathbf{w}) = \mathbf{w}\bullet(\mathbf{v}\times\mathbf{w}) = 0$ と、

$$\begin{aligned}
|\mathbf{v}\times\mathbf{w}|^2 &= |\mathbf{v}|^2|\mathbf{w}|^2\sin^2\theta \\
&= |\mathbf{v}|^2|\mathbf{w}|^2(1-\cos^2\theta) \\
&= |\mathbf{v}|^2|\mathbf{w}|^2\frac{|\mathbf{v}|^2|\mathbf{w}|^2-(\mathbf{v}\bullet\mathbf{w})^2}{|\mathbf{v}|^2|\mathbf{w}|^2} \\
&= |\mathbf{v}|^2|\mathbf{w}|^2-(\mathbf{v}\bullet\mathbf{w})^2
\end{aligned}$$

である。

例 79.

(C1) `load("vectanal.mac");`

(D1) vectanal.mac

(C2) `v : transpose(matrix([a, b, c]));`

(D2) $\begin{pmatrix} a \\ b \\ c \end{pmatrix}$

(C3) `w : transpose(matrix([s, t, u]));`

(D3) $\begin{pmatrix} s \\ t \\ u \end{pmatrix}$

(C4) `v @* (v @x w);`

(D4) $a\,(bu-ct)+b\,(cs-au)+c\,(at-bs)$

(C5) `ratsimp(D4);`

(D5) 0

(C6) `w @* (v @x w);`

(D6) $s\,(bu - ct) + t\,(cs - au) + (at - bs)\,u$
(C7) `ratsimp(D6);`
(D7) 0
(C8) `ratsimp((v @x w) @* (v @x w) - ((v @* v) * (w @* w) - (v @* w)^2));`
(D8) 0

外積というのは、非可換な積である。一般に $\mathbf{v}\times\mathbf{w} = -\mathbf{w}\times\mathbf{v}$ が成り立つ、すなわち、積の順序を入れ換えると計算結果が逆向きのベクトルになる。したがって、特に $\mathbf{v}\times\mathbf{v} = \mathbf{0}$ となる。外積と内積を使って計算する量 $(\mathbf{u}\times\mathbf{v}) \bullet \mathbf{w}$ は、$\mathbf{u}$、$\mathbf{v}$、$\mathbf{w}$ がこの順序で張る平行六面体の符合つき体積に等しく、それは行列 $A = (\mathbf{u}, \mathbf{v}, \mathbf{w})$ の行列式 $|A|$ に等しい。これらのこともチェックしてみよう。

例 80.

(C1) `load("vectanal.mac");`
(D1) `vectanal.mac`
(C2) `v : transpose(matrix([a, b, c]));`
(D2) $\begin{pmatrix} a \\ b \\ c \end{pmatrix}$
(C3) `w : transpose(matrix([e, f, g]));`
(D3) $\begin{pmatrix} e \\ f \\ g \end{pmatrix}$
(C4) `v @x w;`
(D4) $\begin{pmatrix} bg - cf \\ ce - ag \\ af - be \end{pmatrix}$
(C5) `- (w @x v);`
(D5) $\begin{pmatrix} bg - cf \\ ce - ag \\ af - be \end{pmatrix}$
(C6) `ratsimp((v @x w) + (w @x v));`
(D6) $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$
(C7) `v @x v;`
(D7) $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$
(C8) `w @x w;`
(D8) $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$
(C9) `kill(v, w);`
(D9) DONE
(C10) `u : transpose(matrix([a, b, c]));`

(D10) $\begin{pmatrix} a \\ b \\ c \end{pmatrix}$

(C11) `v : transpose(matrix([d, e, f]));`

(D11) $\begin{pmatrix} d \\ e \\ f \end{pmatrix}$

(C12) `w : transpose(matrix([g, h, i]));`

(D12) $\begin{pmatrix} g \\ h \\ i \end{pmatrix}$

(C13) `vol1 : ratsimp((u @x v) @* w);`

(D13) $(ae - bd)\, i + (cd - af)\, h + (bf - ce)\, g$

(C14) `B : zeromatrix(3,3)`

(C15) `B[1] : transpose(u)[1];`

(D15) $[a, b, c]$

(C16) `B[2] : transpose(v)[1];`

(D16) $[d, e, f]$

(C17) `B[3] : transpose(w)[1];`

(D17) $[g, h, i]$

(C18) `A : transpose(B);`

(D18) $\begin{pmatrix} a & d & g \\ b & e & h \\ c & f & i \end{pmatrix}$

(C19) `vol2 : ratsimp(determinant(A));`

(D19) $(ae - bd)\, i + (cd - af)\, h + (bf - ce)\, g$

(C20) `ratsimp(vol1 - vol2);`

(D20) 0

11.2. **微分作用素.** 3次元空間 $\mathbb{R}^3$ 上のベクトル場というのは、空間の各点にひとつのベクトルを対応させる規則のことである。$\mathbb{R}^3$ から、あるベクトル空間 V への写像のことと言ってもいい。特に、$\mathbb{R}^3$ の各点 $\mathbf{x}$ に $\mathbf{x}$ を起点とする有向線分の一つを対応させるものは $\mathbb{R}^3$ 上の接ベクトル場と呼ばれるが、これは一つの写像 $f\colon \mathbb{R}^3 \to \mathbb{R}^3$ と思っていい。この小節では、3次元空間 $\mathbf{R}^3$ に正規直交系として標準基底 $\mathfrak{E} = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$ を入れて考え、空間内の点を $\mathfrak{E}$ に関する座標 $\mathbf{x} = {}^t(x, y, z)$ をつかって表す。通常の関数と区別するため、ベクトル場は太字 $\mathbf{f}$ などを使って表すことにしよう。$\mathbf{f}$ を接ベクトル場とする。$\mathbf{f}$ の点 $\mathbf{x} = {}^t(x, y, z)$ に置ける値は $\mathbf{R}^3$ の一つのベクトルであり、

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} f(x, y, z) \\ g(x, y, z) \\ h(x, y, z) \end{pmatrix}$$

の形に書ける。ここで、f, g, h は各座標関数 x, y, z を変数とする3変数の関数である。$\mathbf{f}$ が (x, y, z) に関して連続であるとき、$\mathbf{f}$ は連続であると言う。また、各成分 f, g, h が

x, y, z の各々について偏微分可能であるとき、$\mathbf{f}$ はその座標関数について偏微分可能であると言う。以下では、各成分が座標関数について適当な微分可能性を持っているようなベクトル場を扱う。

一階線形微分作用素 **grad** は、$\mathbb{R}^3$ 上の関数 f に作用して、$\mathbb{R}^3$ 上の接ベクトル場 $\operatorname{grad} f$ を返す作用素である。ここで、f は各座標関数について偏微分可能な関数で、$\operatorname{grad} f$ は

$$(\operatorname{grad} f)(\mathbf{x}) = \begin{pmatrix} \frac{\partial f}{\partial x}(\mathbf{x}) \\ \frac{\partial f}{\partial y}(\mathbf{x}) \\ \frac{\partial f}{\partial z}(\mathbf{x}) \end{pmatrix}$$

により定義される接ベクトル場である。逆に、一階線形微分作用素 **div** は、各座標関数について偏微分可能な接ベクトル場

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} f(\mathbf{x}) \\ g(\mathbf{x}) \\ h(\mathbf{x}) \end{pmatrix}$$

に対して、関数 div を対応させる作用素である。ここで、 div は

$$(\operatorname{div} \mathbf{f})(\mathbf{x}) = \frac{\partial f}{\partial x}(\mathbf{x}) + \frac{\partial g}{\partial y}(\mathbf{x}) + \frac{\partial h}{\partial z}(\mathbf{x})$$

で定義される。もう一つの一階線形微分作用素 **rot** は、各座標関数について偏微分可能な接ベクトル場

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} f(\mathbf{x}) \\ g(\mathbf{x}) \\ h(\mathbf{x}) \end{pmatrix}$$

に対して、接ベクトル場 $\operatorname{rot} \mathbf{f}$ を対応させる作用素である。ここで、$\operatorname{rot} \mathbf{f}$ は

$$(\operatorname{rot} \mathbf{f})(\mathbf{x}) = \begin{pmatrix} \left(\frac{\partial h}{\partial y} - \frac{\partial g}{\partial z}\right)(\mathbf{x}) \\ \left(\frac{\partial f}{\partial z} - \frac{\partial h}{\partial x}\right)(\mathbf{x}) \\ \left(\frac{\partial g}{\partial x} - \frac{\partial f}{\partial y}\right)(\mathbf{x}) \end{pmatrix}$$

で定義される。$\operatorname{grad} f$ は関数 f の勾配ベクトル場、$\operatorname{div} \mathbf{f}$ はベクトル場 $\mathbf{f}$ の発散、$\operatorname{rot} \mathbf{f}$ はベクトル場 $\mathbf{f}$ の回転という。

これらの微分作用素は、古典電磁気学の完成形である Maxwell の理論の記述に使われることからもわかるように、極めて基本的で重要な微分作用素と考えられる。詳しい数学的性質や、その物理現象への応用などは相応の専門書に譲ることにして、ここでは、**maxima** でこれらの作用素を、前置演算子として定義し、いくつかの計算例を示すことで満足することにしよう。

maxima で前置演算子を定義するのには、コマンド **prefix** を使う。prefix の詳細はマニュアルを見てもらうことにして、大体のところは infix と同様につかえると思えば良い。**maxima** で関数の微分を扱うコマンドは **diff** である。diff(f, x); とタイプすると、関数 f を x に関し (偏) 微分したものを返す。ただし、f は具体的に x を含む形で定義された関数か、あるいは **depends**(f, x) というコマンドで、x に依存することを宣言された形式的な関数である必要がある。それ以外のときは、x に関しては定数とみなされ、0 を返す。prefix と diff を使って、コマンド **grad**、**div** および **rot** を定義しよう。次のプログラムをファイル “vectanal.mac” に追加する。

```
prefix("Grad", 142, expr, expr)$
prefix("Div", 142, expr, expr)$
prefix("Rot", 142 , expr, expr)$

"Grad"(f) := block([gv],
  gv : zeromatrix(3,1),
  gv[1][1] : diff(f, x),
  gv[2][1] : diff(f, y),
  gv[3][1] : diff(f, z),
  return(gv)
)$

"Div"(v) := block([f],
  f : diff(v[1][1], x) + diff(v[2][1], y) + diff(v[3][1], z),
  return(f)
)$

"Rot"(v) := block([w],
  w : zeromatrix(3,1),
  w[1][1] : diff(v[3][1], y) - diff(v[2][1], z),
  w[2][1] : diff(v[1][1], z) - diff(v[3][1], x),
  w[3][1] : diff(v[2][1], x) - diff(v[1][1], y),
  return(w)
)$

Laplacian(f) := div (grad (f))$
```

いくつかの使用例を見てみよう。

例 81. noindent

(C1) `load("vectanal.mac");`

(D1) vectanal.mac

(C2) `diff(f, x);`

(D2) 0

(C3) `depends(f , x);`

(D3) $[f(x)]$

(C4) `diff(f, x);`

(D4) $\frac{d}{dx}f$

(C5) `depends(f, [x, y, z]);`

(D5) $[f(x, y, z)]$

(C6) `grad f;`

(D6) $\begin{pmatrix} \frac{d}{dx}f \\ \frac{d}{dy}f \\ \frac{d}{dz}f \end{pmatrix}$

(C7) `F : transpose(matrix([o, p, q]));`

(D7) $\begin{pmatrix} o \\ p \\ q \end{pmatrix}$

(C8) `div F;`

(D8) 0

(C9) `rot F;`

(D9) $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$

(C10) `depends([o, p, q], [x, y, z]);`

(D10) $[o(x,y,z), p(x,y,z), q(x,y,z)]$

(C11) `div F;`

(D11) $\frac{d}{dz}q + \frac{d}{dy}p + \frac{d}{dx}o$

(C12) `rot F;`

(D12) $\begin{pmatrix} \frac{d}{dy}q - \frac{d}{dz}p \\ \frac{d}{dz}o - \frac{d}{dx}q \\ \frac{d}{dx}p - \frac{d}{dy}o \end{pmatrix}$

(C13) `div (grad f);`

(D13) $\frac{d^2}{dz^2}f + \frac{d^2}{dy^2}f + \frac{d^2}{dx^2}f$

(C14) `grad (div F);`

(D14) $\begin{pmatrix} \frac{d^2}{dx\,dz}q + \frac{d^2}{dx\,dy}p + \frac{d^2}{dx^2}o \\ \frac{d^2}{dy\,dz}q + \frac{d^2}{dy^2}p + \frac{d^2}{dx\,dy}o \\ \frac{d^2}{dz^2}q + \frac{d^2}{dy\,dz}p + \frac{d^2}{dx\,dz}o \end{pmatrix}$

(C15) `rot (grad f);`

(D15) $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$

(C16) `div (rot F);`

(D16) 0

(C17) `Laplacian(f);`

(D17) $\frac{d^2}{dz^2}f + \frac{d^2}{dy^2}f + \frac{d^2}{dx^2}f$

$\mathrm{div}(\mathrm{grad} f)$ は通常 $\Delta(f)$ あるいは Δf などと書かれ、f のラプラシアンと呼ばれる。ラプラス作用素 Δ は 2 階の楕円型作用素の一つである。**maxima** のコマンドとしては **Laplacian** として使える。

これらの微分作用素が実際にどのように使用されるかについては、ここでは全く解説しないので、適当な物理の本を読んでいただきたい。何を計算しているかについては説明しないが、最後にもうちょっとだけ、計算例を付け加えよう。

例 82.

(C1) `load("vectanal.mac");`

(D1) vectanal.mac

(C2) `g(x,y,z) := -1/sqrt(x^2 + y^2 + z^2);`

(D2) $g(x,y,z) := \frac{-1}{\sqrt{x^2+y^2+z^2}}$

```
(C3) grad g(x,y,z);
```

(D3) $\begin{pmatrix} \frac{x}{(z^2+y^2+x^2)^{\frac{3}{2}}} \\ \frac{y}{(z^2+y^2+x^2)^{\frac{3}{2}}} \\ \frac{z}{(z^2+y^2+x^2)^{\frac{3}{2}}} \end{pmatrix}$

```
(C4) ratsimp(sqrt(D3 @* D3));
```

(D4) $\frac{1}{\sqrt{z^4+(2y^2+2x^2)z^2+y^4+2x^2y^2+x^4}}$

```
(C5) factor(z^4 + (2*y^2 + 2*x^2)*z^2 + y^4 + 2*x^2*y^2 + x^4);
```

(D5) $(z^2+y^2+x^2)^2$

```
(C6) ratsimp(D4^2 - 1 / D5);
```

(D6) 0

```
(C7) r : sqrt(x^2 + y^2 + z^2);
```

(D7) $\sqrt{z^2+y^2+x^2}$

```
(C9) rot (grad g(x, y, z));
```

(D9) $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$

```
(C10) div (grad g(x,y,z));
```

(D10) $\frac{3}{(z^2+y^2+x^2)^{\frac{3}{2}}} - \frac{3z^2}{(z^2+y^2+x^2)^{\frac{5}{2}}} - \frac{3y^2}{(z^2+y^2+x^2)^{\frac{5}{2}}} - \frac{3x^2}{(z^2+y^2+x^2)^{\frac{5}{2}}}$

```
(C11) ratsimp(D10);
```

(D11) 0

```
(C12) F : transpose(matrix([sin(x*y), cos(x*y), z^2 - x*y]));
```

(D12) $\begin{pmatrix} \sin(xy) \\ \cos(xy) \\ z^2-xy \end{pmatrix}$

```
(C13) div F;
```

(D13) $2z - x\sin(xy) + y\cos(xy)$

```
(C14) rot F;
```

(D14) $\begin{pmatrix} -x \\ y \\ -y\sin(xy) - x\cos(xy) \end{pmatrix}$

```
(C15) rot (rot F);
```

(D15) $\begin{pmatrix} x^2\sin(xy) - \sin(xy) - xy\cos(xy) \\ -xy\sin(xy) + y^2\cos(xy) + \cos(xy) \\ 0 \end{pmatrix}$

```
(C16) grad (rot F);
```

(D16)
$$\begin{pmatrix} \begin{pmatrix} -1 \\ 0 \\ xy\sin(xy) - y^2\cos(xy) - \cos(xy) \end{pmatrix} \\ \begin{pmatrix} 0 \\ 1 \\ x^2\sin(xy) - \sin(xy) - xy\cos(xy) \end{pmatrix} \\ \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \end{pmatrix}$$

```
(C17) div (rot F);
```

(D17) 0

```
(C18) G : 1 / r^2;
```

(D18) $\frac{1}{z^2+y^2+x^2}$

上の (C16) の grad(rot F) は数学的には定義できないが、**maxima** での演算子の特性として rot F の各成分に対して grad を適用したものが値として返されている。

12. あとがきというか言い訳

大学の線形代数の授業の補助として、行列計算の算盤がわりとして **maxima** を使うための手軽な入門書を目指して書き始めた本文書ですが、当初の意に反して、マニアックな記述に満ちた、著者自身のための備忘録となってしまいました。そういうわけで、この文書を読んで失望される方もおられると思いますが、只々、お詫び申し上げるのみです。

この文書自体は、**maxima** の入門書でも、線形代数学の入門書でもありません。それらについては、それ用の文献を見て下さい。この下に、私が好きな参考書のリストを掲載します。

この文書を書くにあたっては、私の３人の友人に読者、モニターになって頂き、貴重な御意見を頂きました。上田秀雄さん、古関義之さん、吉岡雄一さんに感謝申し上げます。

maxima 関係の参考資料

[1] Maxima Manual, http://maxima.sourceforge.net/docs.shtml .
[2] Paulo Ney de Souza, Richard J. Fateman, Joel Moses and Cliff Yapp, The Maxima Book, http://maxima.sourceforge.net/docs.shtml .
[3] Michael Clarkson, DOE-Maxima Reference Manual, http://maxima.sourceforge.net/docs.shtml .

線形代数学の参考文献

[4] 伊原信一郎, 河田敬義, 線型空間、アフィン幾何, 岩波基礎数学選書
[5] 杉浦光夫, 横沼健雄, ジョルダン標準形、テンソル代数, 岩波基礎数学選書
[6] 田坂隆士, ２次形式, 岩波基礎数学選書
[7] 斎藤正彦, 線型代数入門, 東京大学出版会
[8] 長岡亮介, ⟨ 新版 ⟩ 線型代数学-入門と展望, ブレーン出版

その他の参考文献

[9] Joachim von zur Gathen and Jürgen Gerhard, Modern Computer Algebra, Cambridge University Press.
[10] 酒井皇治, やさしい Lisp 入門, 株式会社ニコラステラ
[11] ポール グレアム, ANSI Common Lisp, ピアソン • エデュケーション
[12] Paul Graham, On Lisp, http://www.paulgraham.com
[13] David S. Touretzky, Common Lisp: A Gentle Introduction to Symbolic Computation, http://www-2.cs.cmu.edu/ dst/LispBook/index.html

E-mail address: `kenrou@yo.rim.or.jp`