# 線形数学とＪ－Ｊを関数電卓として用いる

SHIMURA Masato
jcd02773@nifty.ne.jp

2019 年 1 月 21 日

## 目次

1　高校数学での線形代数の扱い　1
2　何故 J 言語を薦めるか　2
3　特集 2[速習・線形代数] をレビューする　2
付録 A　Projecteuler　13
付録 B　[どうやって線形代数を学んだらいいの]　13

概要

## 1　高校数学での線形代数の扱い

月刊のソフトウエアデザイン誌（技術評論社刊）は 1985 年から高水準を保つ貴重な雑誌だ。ここの 2019 新年号は特集で「IT エンジニアのための機械学習と線形代数入門」を取り上げていた。

ここに中西崇文「高校教育課程の変化と大学出の数学」という記事が目に留まった。氏は武蔵野大学に 2019 年 4 月から開設されるデータサイエンス学部の准教授で「データと数理 I」を担当されるとのことだが、高校数学の実情が紹介されている。

- 線形数学が高校数学に入ったのは 1970 の「数 IIB」
- 1989 数学 C に移る
- 2009 数学 C がなくなり、消えた

- 数学活用の中に一部が残っているが限定的
- 複素平面が線形数学と裏表で浮沈している。

## 2　何故 J 言語を薦めるか

J 言語の生い立ちは Appendix を参照のこと

1. ベクトル、マトリクスに配列環が通っている。
2. 複素数も同様
3. 数の表現方法が豊富
4. サイエンスプロットで簡単にグラフが描ける。ガウス座標も取り扱える

J はインタプリッタなので高級電卓代わりに使え、こんなに簡単にできますよといくつかの例題を紹介したい。

## 3　特集 2[速習・線形代数] をレビューする

この項の著者は橘慎太郎氏。高校数学の復習との副題がついている。サポートは次で

`https:`/umentulab.com/

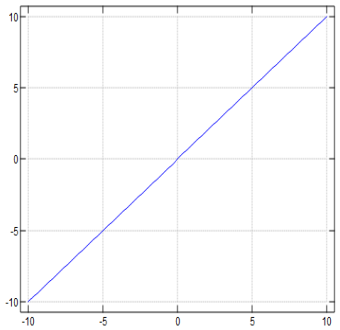
### 3.1　線形空間

1. グラフはサイエンスプロットで描く。最初に plot と出力の png をロードする。
   `require 'plot png '`
   書式は `plot 区間 ; '関数'`
2. $f(x) = x$ と $f(x) = x^2$ を描く

$$f(x) = x$$

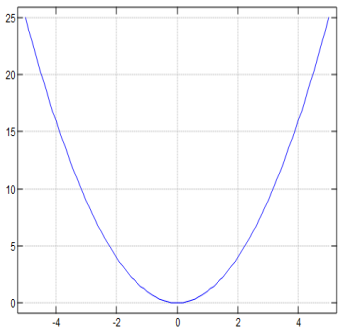
```
plot _10 10;']'

pd 'save png c:/temp/plot_01.png'
```



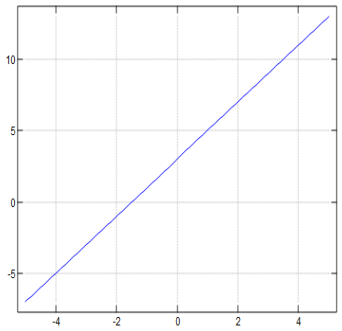
$$f(x) = x^2$$

```
plot _5 5;'*:'
```



$$f(x) = 2x + 3$$

```
plot _5 5 ;'3 + 2&* '
```



筆者は線形性を持つ関数とは「原点を通る直線」といささか窮屈な定義をしている

[J Grammar]

- _5 5　　J ではマイナスの演算記号と符合を区別する。符合は _5 とアンダーバーで書く。K.E.Iverson のセンスがでており、数学の弱点を正している。したがって括弧なしで次のように表せる

```
   5-_5
10
```

- ]　　右引数を採る。ここでは区間を取り込む
- *:　　2 乗. x^2 でもよい
- 2&*　　$2 \times x$ . * 2 でもよい

### 3.2　ベクトル

アレープロッセッシング言語はベクトルも簡単に使える。FORTRAN は 90 で配列をサポートした。おなじみの手続き型言語と比較してみてください。

1 2 3 4 5

これを 5 次のベクトル、一行のみの配列、単なる数字の塊、どう見るかは自由です。

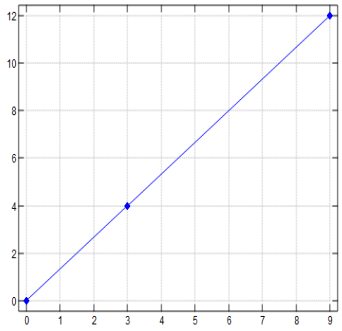
J は 1 行はリスト、複数行はテーブル、枚数のあるのはレポートと呼んでいます。レポートは支店毎の月別売上のようなものです。

ベクトルとして取り扱えば厳密にベクトル計算をします。

1. ベクトルの表現
   - ベクトルのスカラー倍

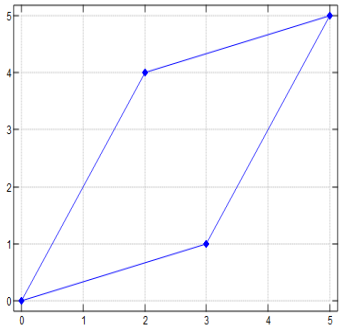
```
   3 4 * 3
9 12

'line,marker' plot
0 3 9;0 4 12
```



- ベクトル同士の加算

```
   3 1 + 2 4
5 5
   'line marker'  plot
   0 3 5 2 0 ;0 1 5  4 0
```



2. マトリクスの乗算

（a）マトリクス同士の乗算

（b）右をトランスポーズする

```
   (i. 3 3);(9+i. 3 3);(i.3 3)*|: 9+i.3 3
+-----+--------+---------+
|0 1 2| 9 10 11| 0 12  30|
|3 4 5|12 13 14|30 52  80|
|6 7 8|15 16 17|66 98 136|
+-----+--------+---------+
```

（c）トランスポーズしないとベタの掛け合わせ

```
   (i. 3 3);(9+i. 3 3);(i.3 3)* 9+i.3 3
+-----+--------+----------+
|0 1 2| 9 10 11| 0  10  22|
|3 4 5|12 13 14|36  52  70|
|6 7 8|15 16 17|90 112 136|
+-----+--------+----------+
```

3. 内積

```
    (i. 3 3);(9+i. 3 3);(i.3 3) +/ . * 9+i.3 3
+-----+--------+-----------+
|0 1 2| 9 10 11| 42  45  48| NB. 42= 0 1 2 * 9 12 15// 45 = 0 1 2
|3 4 5|12 13 14|150 162 174|
```

```
|6 7 8|15 16 17|258 279 300|
+-----+--------+-----------+
```

4. ベクトルの内積

$$\begin{pmatrix}1\\2\\3\end{pmatrix}\cdot\begin{pmatrix}4\\5\\6\end{pmatrix}=32$$

```
   1 2 3 * 4 5 6
4 10 18

   1 2 3 +/ . * 4 5 6
32
```

[J Grammar]

- 内積は +/ (space) . *
- 内積を mp と定義してしまおう。ベクトルにも行列にも用いることができる。

```
   mp=: +/ . *
   1 2 3 mp 4 5 6
32
```

- この内積は*を他の演算記号に変えて一般内積として用いることができる

5. ベクトルのノルム

$$\|x\| = \sqrt{x, x}$$

```
   ([: %: +/@: *:) 1 _2 3
3.74166
```

## 3.3 行列

1. 行列の生成
   - 順序数による

     i. は integer で 0 から始まる順序数を打ち出す。3 行 4 列を指定

```
   i. 3 4
0 1  2  3
4 5  6  7
8 9 10 11
```

   - 乱数による生成

     同じ乱数を出すのは?.

     $ は Shape で指定したマトリクスの行と列を構成する

```
   ? 4 5 $100
93 88 97 72 53
22 85 39 97 94
82 35 79 90 23
81 52 67 34 13
```

   - 0 行列

```
   3 4 $ 0
0 0 0 0
0 0 0 0
0 0 0 0
```

   - 手入力

     行ごとにカンマで区切り最終行はラミネート (,:) で繋ぐ

```
 ]a=: 1 2 3 4,5 6 7 8,:9 10 11 12

1  2  3  4
```

```
  5  6  7  8
 9 10 11 12
```

2. 転置/Transpose　　|: を用いる

```
   i. 4 3
 0  1  2
 3  4  5
 6  7  8
 9 10 11
```

```
    |: i. 4 3
 0 3 6  9
 1 4 7 10
 2 5 8 11
```

3. 単位行列
イディオム　=/˜ が便利。他にもいくつか手法がある

```
     =/˜ i. 3
1 0 0
0 1 0
0 0 1
```

4. 逆行列 (matrix inverse %. ) 縦長の行列の逆行列も計算できる。

```
 a=. ?.  4 3 $ 20
```

```
14 16  8
 6  5  8
 6 16 16
19 13 12
```

```
 %.  a
```

```
_0.00141225 0.0183164 _0.0581375 0.0662473
  0.106047 _0.072367  0.0322677 _0.065477
 _0.100045 0.0778555  0.0481502  0.033926
```

5. 掃き出しと連立方程式の解
鶴亀算の初出は中国は南北朝時代（三国志の魏に取って代わった晋が押されて南に逃げた時代）の「孫氏算経」に雉兎算として出ている。著者の孫氏は兵法家とは別人

（a）ガウスの掃き出し法は万能。手計算で叩き込もう

$$\begin{cases} x + 2y = 8 \\ 2x + 3y = 13 \end{cases}$$

- 拡大係数行列

  A=. 1 2 8,: 2 3 13

  移項ではないので右項の符合は変えない

$$\begin{matrix} 1 & 2 & 8 \\ 2 & 3 & 13 \end{matrix}$$

- 0 行を-2 倍して 1 行に加える

$$\begin{matrix} 1 & 2 & 8 \\ 0 & -1 & -3 \end{matrix}$$

- 1 行を 2 倍して 0 行に加え

$$\begin{matrix} 1 & 0 & 2 \\ 0 & -1 & -3 \end{matrix}$$

- 1 行の符合を反転すると、左項は単位行列になる

$$\begin{matrix} 1 & 0 & 2 \\ 0 & 1 & 3 \end{matrix}$$

（b）クラメール法

拡大係数行列を見かけたらクラメール法で解こう。このクラメール法はあまり類書で紹介されていない

拡大係数行列を係数行列で割る。J の強力な行列除算機能を用いる

```
   cr=: %. }:"1
```

$$\begin{matrix} 1 & 2 & 8 \\ 2 & 3 & 13 \end{matrix}$$

---

$$\begin{matrix} 1 & 2 \\ 2 & 3 \end{matrix}$$

```
   cr A
1 0 2
0 1 3
```

6. 行列式

行列式は関孝和とライプニッツが個別にほぼ同時に考案した。関の名を紹介する数学書はほとんどない。和算家の中には関は途中で間違えていると指摘する人もいるがライプニッツの方がより低次で間違えており、関のは弟子の建部賢弘が正している。

行列式は従属し元を張れない行列や潰れている行列をを $detA = 0$ などで示してくれる。

```
   ] a=. 2 4 6,_1 2 3,:4 1 2
 2 4 6
_1 2 3
 4 1 2

   det=: -/ . *
   det a
4
```

7. トレース

対角行列を取り出すイディオム

```
 tr=:  (<1 0)&|:

    >:i. 4 5
 1  2  3  4  5
 6  7  8  9 10
11 12 13 14 15
16 17 18 19 20
   (<0 1)&|: >:i.4 5
1 7 13 19
```

対角行列が取り出せれば $\sum, \Pi$ は　+/ */ で扱える

8. 内積

内積演算　mp=: +/ . *

mp はベクトルにもマトリクスにも用いることができる

```
   (1 2 3,: 4 5 6) +/ . * 7 8,9 10,:11 12
 58  64
139 154
```

$$\begin{pmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{pmatrix}\cdot\begin{pmatrix}7 & 8\\ 9 & 10\\ 11 & 12\end{pmatrix}$$

$$\begin{aligned}(1\times7)\ &+(2\times9)\ &+(3\times11)\ &= 58\\ (1\times8)\ &+(2\times10)\ &+(3\times12)\ &= 64\\ (4\times7)\ &+(5\times9)\ &+(6\times11)\ &= 139\\ (4\times8)\ &+(5\times10)\ &+(6\times12)\ &= 154\end{aligned}$$

9. シュミットの直交化

潰れかけた行列を補強するシュミットの直交化。書き出すと論文いっぺんになるので手法だけ示す。J では外部接続詞に QR 法を求める手法の仮置きとして入っている。

```
   A=: 1 2 1,1 _1 2,: _1 1 3
   A
 1  2 1
 1 _1 2
_1  1 3
   clean 128!:0 A
+--------------------------+----------------------+
| 0.57735  0.816497        0|1.73205       0       0|
| 0.57735 _0.408248 0.707107|      0 2.44949 1.22474|
|_0.57735  0.408248 0.707107|      0       0 3.53553|
+--------------------------+----------------------+
```

左がシュミットの直交化、右が上三角行列

10. LAPACK

固有値と固有ベクトルはちょっと大掛かりになるので addons/math/LAPACK に入っている LAPACK を用いるのが早道だ

```
require '~addons/math/lapack/lapack.ijs'
require '~addons/math/lapack/geev.ijs'
```

```
 }. geev_jlapack_ K0
+----------+-----------------------------------------------+
|1j1 1j_1 1|          0.725476            0.725476 _0.57735|
|          |_0.435286j0.145095 _0.435286j_0.145095  0.57735|
|          |0.507833j0.0725476 0.507833j_0.0725476 _0.57735|
+----------+-----------------------------------------------+
```

## 付録 A　Projecteuler

Projecteuler で世界の腕に覚えのある回答者 100 万人近くの使用している言語を見ると

- 双璧は C 連合 (C C++ C# JAVA) と Python である。
- 実に多様な言語が用いられている
- BASIC は意外に少ない
- Mathematica もメジャーではない
- Pencil/Paper という猛者もいる

APL/J/K も数は多くないがしっかりとした地合いを占めている

J は K.E.Iverson が IBM で開発した APL の方言として、R.Hui の手助けでキーボード記号を用いて新しい機能も盛り込んで開発した言語である。

少数民族の関数型言語を用いる目的の一つが 2，30 年進んだ先進機能を使うことにあるといわれる。APL や J は基本設計が配列計算言語であり、多くの配列計算機能を持つ言語の源流となっている。

## 付録 B　[どうやって線形代数を学んだらいいの]

引用論文の著者のまとめは次の 3 点

1. 手を動かすことが一番
2. 名著と呼ばれる本を手に取ってみる
3. 餅は餅屋に聞く

付加えれば、ブランド大学の数学者の本は最後にする。洋書など例題が丁寧に書いてある本を選び例題を Pencil/Paper や高級関数電卓で実際に演算してみることが有用である y