行列演算の演習問題

ここでは，行列計算の演習問題を実施します．これらの問題は，最初に紙に書いて手計算でおこなってください．次に検証も兼ねてPythonによる計算を実施してください．このように手計算を実施することは，理論についての理解を深めます．また，その後でPythonで計算することは，現実の問題に立ち向かうときのITノウハウを養ってくれます．着実に身に付けるように学習してください．

- 最初に手書きで解いてください．
- 次にPythonで解いてください．

問題-1

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

2次正方行列どうしの積です．

問題-2

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \\ 31 & 32 & 33 \end{pmatrix}$$

単位行列の積です．

問題-3

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \\ 31 & 32 & 33 \end{pmatrix}$$

行を入替える行う行列の積です．

問題-4

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

2行3列の行列と3行2列の行列の積です．

問題-5

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

3行2列の行列と2行3列の行列の積です．前問と比較して，行列の積についての理解を深めてください．

問題-6

$$\begin{pmatrix} 3 & 4 & 5 \end{pmatrix} \begin{pmatrix} 3 \\ 4 \\ 5 \end{pmatrix}$$

1行3列の行列と3行1列の行列の積です．

問題-7

$$\begin{pmatrix} 3 \\ 4 \\ 5 \end{pmatrix} \begin{pmatrix} 3 & 4 & 5 \end{pmatrix}$$

3行1列の行列と1行3列の行列の積です．

問題-8

$$\begin{pmatrix} 1+i & 2i \\ -i & -1 \end{pmatrix} \begin{pmatrix} 1-i & i \\ i & -1 \end{pmatrix}$$

複素数を成分とする2次正方行列どうしの積です．

解答編

PythonではNumPyを使用するので，事前にインポートします．

```
import numpy as np
```

```
In [1]: import numpy as np
```

解答-1

この問題は2行2列の行列どうしの積なので，計算可能です．まず，行列の積の手順に沿って手計算を実施します．

$$\begin{pmatrix} 2 & 1 \\ -3 & 7 \end{pmatrix} \begin{pmatrix} 3 & -5 \\ -4 & 1 \end{pmatrix} = \begin{pmatrix} 2 \times 3 + 1 \times (-4) & 2 \times (-5) + 1 \times 1 \\ -3 \times 3 + 7 \times (-4) & -3 \times (-5) + 7 \times 1 \end{pmatrix} = \begin{pmatrix} 6 - 4 & -10 + 1 \\ -9 - 28 & 15 + 7 \end{pmatrix}$$
$$= \begin{pmatrix} 2 & -9 \\ -37 & 22 \end{pmatrix}$$

この計算をPythonで実施してみます．

```
np.array([[2,1],[-3,7]]).dot(np.array([[3,-5],[-4,1]]))
```

```
In [2]: np.array([[2,1],[-3,7]]).dot(np.array([[3,-5],[-4,1]]))
Out[2]: array([[  2,  -9],
               [-37,  22]])
```

解答-2

この問題は3行3列の行列どうしの積なので，計算可能です．この問題で注目すべきは，左側の行列が単位行列$\mathbf{I}_3$ということです．単位行列を掛けても相手側の行列の値を変更しないことを，実際に計算して確認します．

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \\ 31 & 32 & 33 \end{pmatrix}$$
$$= \begin{pmatrix} 1 \times 11 + 0 \times 21 + 0 \times 31 & 1 \times 12 + 0 \times 22 + 0 \times 32 & 1 \times 13 + 0 \times 23 + 0 \times 33 \\ 0 \times 11 + 1 \times 21 + 0 \times 31 & 0 \times 12 + 1 \times 22 + 0 \times 32 & 0 \times 13 + 1 \times 23 + 0 \times 33 \\ 0 \times 11 + 0 \times 21 + 1 \times 31 & 0 \times 12 + 0 \times 22 + 1 \times 32 & 0 \times 13 + 0 \times 23 + 1 \times 33 \end{pmatrix}$$
$$= \begin{pmatrix} 11 + 0 + 0 & 12 + 0 + 0 & 13 + 0 + 0 \\ 0 + 21 + 0 & 0 + 22 + 0 & 0 + 23 + 0 \\ 0 + 0 + 31 & 0 + 0 + 32 & 0 + 0 + 33 \end{pmatrix} = \begin{pmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \\ 31 & 32 & 33 \end{pmatrix}$$

計算結果として，確かに右側の行列の値を維持しました．これをPythonで確認します． 3次元の単位行列は，numpy.identity(3)で生成できます．

```
np.identity(3).dot(np.array([[11,12,13],[21,22,23],[31,32,33]]))
```

```
In [3]: np.identity(3).dot(np.array([[11,12,13],[21,22,23],[31,32,33]]))
Out[3]: array([[ 11.,  12.,  13.],
               [ 21.,  22.,  23.],
               [ 31.,  32.,  33.]])
```

解答-3

この問題は3行3列の行列どうしの積なので，計算可能です．この問題で注目すべきは，左側の行列が行を入替える行列$\mathbf{P}_3(2,3)$ということです．右の行列の2行目と3行目が入れ替わることを，実際に計算して確認します．

$$
\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \\ 31 & 32 & 33 \end{pmatrix}
$$

$$
= \begin{pmatrix} 1\times 11 + 0\times 21 + 0\times 31 & 1\times 12 + 0\times 22 + 0\times 32 & 1\times 13 + 0\times 23 + 0\times 33 \\ 0\times 11 + 0\times 21 + 1\times 31 & 0\times 12 + 0\times 22 + 1\times 32 & 0\times 13 + 0\times 23 + 1\times 33 \\ 0\times 11 + 1\times 21 + 0\times 31 & 0\times 12 + 1\times 22 + 0\times 32 & 0\times 13 + 1\times 23 + 0\times 33 \end{pmatrix}
$$

$$
= \begin{pmatrix} 11+0+0 & 12+0+0 & 13+0+0 \\ 0+0+31 & 0+0+32 & 0+0+33 \\ 0+21+0 & 0+22+0 & 0+23+0 \end{pmatrix} = \begin{pmatrix} 11 & 12 & 13 \\ 31 & 32 & 33 \\ 21 & 22 & 23 \end{pmatrix}
$$

計算結果として，確かに右側の行列の2行目と3行目が入れ替わりました．これをPythonで確認します．

```
np.array([[1,0,0],[0,0,1],[0,1,0]]).dot(np.array([[11,12,13],[21,22,23],[31,32,33]]))
```

```
In [4]: np.array([[1,0,0],[0,0,1],[0,1,0]]).dot(np.array([[11,12,13],[21,22,23],[31,32,3
        3]]))

Out[4]: array([[11, 12, 13],
               [31, 32, 33],
               [21, 22, 23]])
```

解答-4

この問題は左側の列数と右側の列数が3で等しいので計算可能です． 2行3列の行列と3行2列の行列の積は2行2列の正方行列になります．

$$
\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix} = \begin{pmatrix} 1\times 1 + 2\times 3 + 3\times 5 & 1\times 2 + 2\times 4 + 3\times 6 \\ 4\times 1 + 5\times 3 + 6\times 5 & 4\times 2 + 5\times 4 + 6\times 6 \end{pmatrix}
$$

$$
= \begin{pmatrix} 1+6+15 & 2+8+18 \\ 4+15+30 & 8+20+36 \end{pmatrix} = \begin{pmatrix} 22 & 28 \\ 49 & 64 \end{pmatrix}
$$

これをPythonで確認します．

```
np.array([[1,2,3],[4,5,6]]).dot(np.array([[1,2],[3,4],[5,6]]))
```

```
In [5]: np.array([[1,2,3],[4,5,6]]).dot(np.array([[1,2],[3,4],[5,6]]))

Out[5]: array([[22, 28],
               [49, 64]])
```

解答-5

この問題は左側の列数と右側の列数が3で等しいので計算可能です． 3行2列の行列と2行3列の行列の積は3行3列の正方行列になります．

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix} \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} = \begin{pmatrix} 1\times1+2\times4 & 1\times2+2\times5 & 1\times3+2\times6 \\ 3\times1+4\times4 & 3\times2+4\times5 & 3\times3+4\times6 \\ 5\times1+6\times4 & 5\times2+6\times5 & 5\times3+6\times6 \end{pmatrix}$$
$$= \begin{pmatrix} 1+8 & 2+10 & 3+12 \\ 3+16 & 6+20 & 9+24 \\ 5+24 & 10+30 & 15+36 \end{pmatrix} = \begin{pmatrix} 9 & 12 & 15 \\ 19 & 26 & 33 \\ 29 & 40 & 51 \end{pmatrix}$$

これをPythonで確認します．

```
np.array([[1,2],[3,4],[5,6]]).dot(np.array([[1,2,3],[4,5,6]]))
```

```
In [6]: np.array([[1,2],[3,4],[5,6]]).dot(np.array([[1,2,3],[4,5,6]]))

Out[6]: array([[ 9, 12, 15],
               [19, 26, 33],
               [29, 40, 51]])
```

解答-6

この問題は左側の列数と右側の列数が3で等しいので計算可能です． 1行3列の行列と3行1の行列の積は1行1列の正方行列になりますが，これはスカラーと同一視できるので，結果は1次行列ではなく単にスカラーで表現します．

$$\begin{pmatrix} 3 & 4 & 5 \end{pmatrix} \begin{pmatrix} 3 \\ 4 \\ 5 \end{pmatrix} = \begin{pmatrix} 3\times3+4\times4+5\times5 \end{pmatrix} = \begin{pmatrix} 9+16+25 \end{pmatrix} = \begin{pmatrix} 50 \end{pmatrix} = 50$$

これをPythonで確認します．まず，数学的に正しい記述方法でプログラムを書いてみます．

```
np.array([3,4,5]).dot(np.array([[3],[4],[5]]))
```

```
In [7]: np.array([3,4,5]).dot(np.array([[3],[4],[5]]))

Out[7]: array([50])
```

数学的には正しい表現ではないですが，現実的プログラムでは次のように記述します．上のプログラムとの違いについて，確認してください．

```
np.array([3,4,5]).dot(np.array([3,4,5]))
```

```
In [8]: np.array([3,4,5]).dot(np.array([3,4,5]))
Out[8]: 50
```

このようなPythonによる計算と数学的定義の違いを上手く利用できると，効率的なプログラムを書けるようになります．数学的に厳密でないプログラム仕様を見極めるポイントは，数学的には定義されていない場合であることです．このPythonによる記述も数学的に書けば，1行3列どうしの積 $\begin{pmatrix}3 & 4 & 5\end{pmatrix}\begin{pmatrix}3 & 4 & 5\end{pmatrix}$ という表現なので，数学的には定義されていないものです．

解答-7

この問題は左側の列数と右側の列数が1で等しいので計算可能です． 3行1列の行列と1行3の行列の積は3行3列の正方行列になります．

$$\begin{pmatrix}3\\4\\5\end{pmatrix}\begin{pmatrix}3 & 4 & 5\end{pmatrix} = \begin{pmatrix}3\times3 & 3\times4 & 3\times5\\4\times3 & 4\times4 & 4\times5\\5\times3 & 5\times4 & 5\times5\end{pmatrix} = \begin{pmatrix}9 & 12 & 15\\12 & 16 & 20\\15 & 20 & 25\end{pmatrix}$$

これをPythonで確認します．次のように記載すれば良いかと思われるかもしれませんが，このように記載するとエラーとなります．

```
np.array([[3],[4],[5]]).dot(np.array([3,4,5]))
```

Pythonに右側のオブジェクトが1行3列の行列であることを明示するために配列を2重配列で定義します．

```
np.array([[3],[4],[5]]).dot(np.array([[3,4,5]]))
```

```
In [9]: np.array([[3],[4],[5]]).dot(np.array([[3,4,5]]))
Out[9]: array([[ 9, 12, 15],
               [12, 16, 20],
               [15, 20, 25]])
```

解答-8

この問題は2行2列の行列どうしの積なので，計算可能です．まず，行列の積の手順に沿って手計算を実施します．

$$\begin{pmatrix}1+i & 2i\\-i & -1\end{pmatrix}\begin{pmatrix}1-i & i\\i & -1\end{pmatrix} = \begin{pmatrix}(1+i)\times(1-i)+2i\times i & (1+i)\times i+2i\times(-1)\\-i\times(1-i)+(-1)\times i & -i\times i+(-1)\times(-1)\end{pmatrix}$$
$$= \begin{pmatrix}2-2 & i-1-2i\\-i-1-i & 1+1\end{pmatrix} = \begin{pmatrix}0 & -1-i\\-1-2i & 2\end{pmatrix}$$

この計算をPythonで実施してみます． Pythonで虚数単位はjで表し，虚数単位の前には必ず数字を記載します．

```
np.array([[1+1j,2j],[-1j,-1]]).dot(np.array([[1-1j,1j],[1j,-1]]))
```

```
In [10]: np.array([[1+1j,2j],[-1j,-1]]).dot(np.array([[1-1j,1j],[1j,-1]]))

Out[10]: array([[ 0.+0.j, -1.-1.j],
                [-1.-2.j,  2.+0.j]])
```

このように計算結果がゼロの成分でもオブジェクト型が複素数なので$0+0j$となります．