

基本行列にみる行列の座標変換的意味

えいみん

2019.09.22

全ての正則な行列は基本行列に分解できる。つまり、基本行列の座標変換的意味を理解することによって、行列の座標変換 (一次変換) 的な意味を理解できる。

本稿では、導入として一次変換 (座標変換) の説明を具体例とサンプルのプログラムにて示し、座標変換の特殊な例として回転行列を紹介する。そして、最後に、基本行列と基本行列の座標変換的意味を説明する。

1 座標変換 (一次変換)

座標変換 (一次変換) の例を下記に示す。下記の図 1, 図 2 は巻末のリスト 1 にて生成した。プログラム中の rotate の係数を変更することで座標変換の形を変えられる。

図 1 は行列 (式 1) にて変換する前 (単位行列にて生成) のものであり、 x 軸、 y 軸の大きさに対して RGB 値の大きさ (色) が対応している。図 2 は行列 (式 1) にて変換した後のものである。

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \quad (1)$$

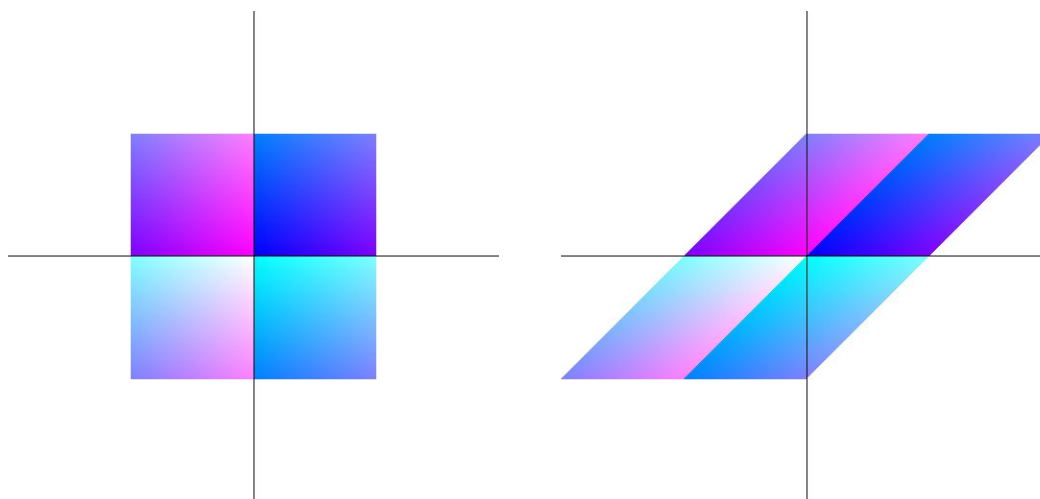


図 1: 変換前 (水平線: x 軸 / 垂線: y 軸)

図 2: 変換後 (水平線: x' 軸 / 垂線: y' 軸)

x 軸、 y 軸に対して各ピクセルの RGB 値 (色) が一意 (ユニーク) に定まっており、同じ RGB 値 (同じ色) のピクセルは存在しない。一意な各ピクセルが、行列によって一対一対応で一意に座標変換 (一次変換) されていることが図から分かる。

2 座標変換の直感的理解 (回転行列)

座標変換の直感的理解の例として回転行列を紹介する。回転行列は原点 $(0, 0)$ を中心に θ だけ座標系を回転させる。次の式 2 で表される。

図 3, 図 4 は巻末のリスト 2 にて生成した。プログラム中の `theta` の値を変更することで写真 (`img.jpg`)¹ の回転の大きさを変えられる。図 3 は行列 (式 2) にて変換する前 (単位行列にて生成) のものであり、図 4 は行列 (式 2) にて変換した後のものである。図よりプログラムにて設定した $\theta = \pi/6$ だけ座標系が回転していることが分かる。

本プログラムの実装上小数点以下は `int` 関数にて切り捨てているため、拡大すると白いピクセルが存在することが分かる。概念的な理解を目的としているため、許容することとした。

$$\begin{aligned} \begin{pmatrix} x' \\ y' \end{pmatrix} &= \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \\ &= \begin{pmatrix} x \cos \theta - y \sin \theta \\ x \sin \theta + y \cos \theta \end{pmatrix} \end{aligned} \quad (2)$$



図 3: 変換前 (水平線: x 軸 / 垂線: y 軸)

図 4: 変換後 (水平線: x' 軸 / 垂線: y' 軸)

3 基本行列

基本行列の定理^[1] を以下に引用する。

定理 4.4

A が正則ならば、左 [あるいは右] 基本変形だけによって A を単位行列に変形することができる。逆も正しい。

上記の定理は、つまり、正則な行列 (逆行列のある行列) は単位行列と有限個の基本行列の積で表されるという意味であり、逆行列のある全ての行列は基本行列に分解できるという意味である。

正則な全ての行列 (逆行列のある全ての行列) は基本行列に分解できるのだから、基本行列の座標変換の意味を理解することによって、正則な逆行列の座標変換の意味を明らかにできる。

まず、基本行列の形を示す。次に基本行列の座標変換の意味を説明する。

¹Wikipedia よりアオサギの写真。

3.1 座標軸の交換

i 行 i 列 ($p_{ii} = 0$) と j 行 j 列 ($p_{jj} = 0$) の係数が 0 であり、 j 行 i 列と i 行 j 列の係数が 1 の行列を以下の通り定義する。ただし、 i 行 i 列と j 行 j 列以外の対角成分は 1 であって、 j 行 i 列と i 行 j 列以外の非対角成分は 0 である。この行列の座標変換の意味は座標軸の交換である。

$$P_n(i, j) = \begin{pmatrix} 1 & & & & & & & O \\ & \ddots & & & & & & \\ & & 1 & & & & & \\ i \text{ 行} & \cdots & \cdots & 0 & \cdots & \cdots & \cdots & 1 \\ & & & \vdots & 1 & & & \vdots \\ & & & \vdots & & \ddots & & \vdots \\ & & & \vdots & & & 1 & \vdots \\ j \text{ 行} & \cdots & \cdots & 1 & \cdots & \cdots & \cdots & 0 \\ & & & \vdots & & & & \vdots & 1 \\ & & & \vdots & & & & \vdots & & \ddots \\ O & & i \text{ 列} & & & j \text{ 列} & & & & 1 \end{pmatrix}, \quad (i \neq j)$$

例題.

つぎのベクトル $\mathbf{b} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{pmatrix}$ と基本行列 $P_5(2, 3)$ の積を求めよ。

$$\begin{aligned} P_5(2, 4)\mathbf{b} &= P_5(1, 5) \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{pmatrix} \\ &= \underline{\begin{pmatrix} 5 \\ 2 \\ 3 \\ 4 \\ 1 \end{pmatrix}} \end{aligned}$$

補足.

例として、図 5 の点群 (RGB 値 (色) は座標依存) を基本行列 $P_2(1, 2) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ にて座標変換した時の図を、図 6 として示す。図は巻末のリスト 1 にて生成した。
座標軸 ($x \rightarrow y (= x')$, $y \rightarrow x (= y')$) が入れ替わっていることが分かる。

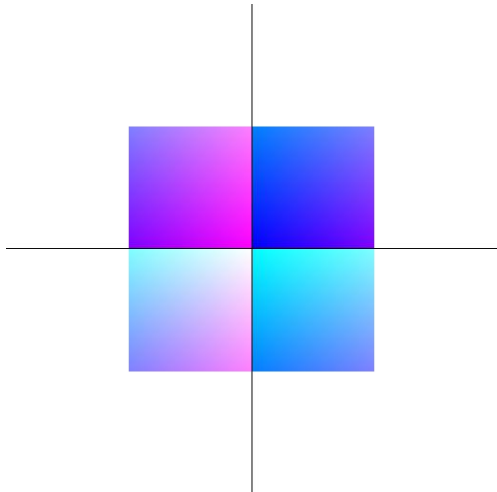


図 5: 変換前 (水平線: x 軸 / 垂線: y 軸)

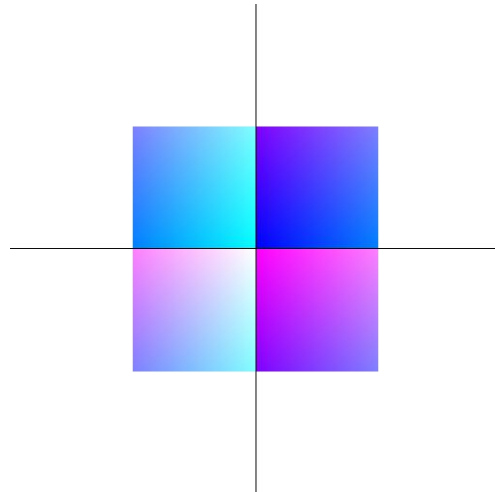


図 6: 変換後 (水平線: y 軸 / 垂線: x 軸)

3.2 座標軸の拡大縮小

i 行 i 列 ($q_{ii} = c$) の係数が c であり、 i 行 i 列以外の対角成分は 1 であって非対角成分は 0 である行列を以下の通り定義する。この行列の座標変換的意味は座標軸の拡大縮小 (c 倍) である。

$$Q_n(i; c) = \begin{pmatrix} 1 & & & & & O \\ & \ddots & & & & \\ & & 1 & & & \\ i \text{ 行} & \cdots & \cdots & c & & \\ & & \vdots & 1 & & \\ & & \vdots & & \ddots & \\ O & & & i \text{ 列} & & 1 \end{pmatrix}, \quad (c \neq 0)$$

例題.

つぎのベクトル $\mathbf{b} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ と基本行列 $Q_3(2; 3.14)$ の積を求めよ。

$$\begin{aligned} Q_3(2; 3.14)\mathbf{b} &= Q_3(2; 3.14) \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 3.14 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \\ &= \underline{\begin{pmatrix} 1 \\ 3.14 \\ 1 \end{pmatrix}} \end{aligned}$$

補足.

例として、図 7 の点群 (RGB 値 (色) は座標依存) を基本行列 $Q_2(1, 0.5) = \begin{pmatrix} 0.5 & 0 \\ 0 & 1 \end{pmatrix}$ にて座標変換した時の図を、図 8 として示す。図は巻末のリスト 1 にて生成した。

座標軸 x' が座標軸 x に対して 0.5 倍に縮小されていることが分かる。

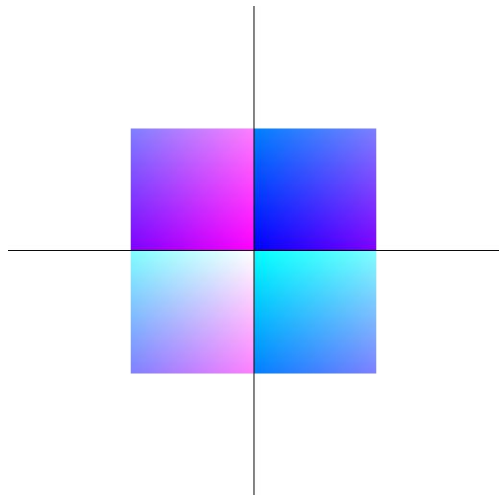


図 7: 変換前 (水平線: x 軸 / 垂線: y 軸)

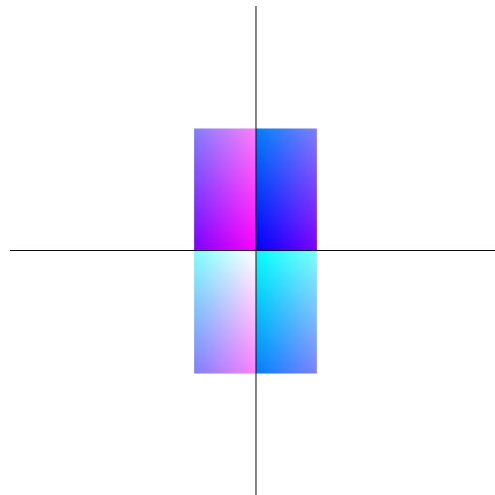


図 8: 変換後 (水平線: x' 軸 / 垂線: y' 軸)

3.3 座標軸の結合

非対角成分の i 行 j 列 ($r_{ij} = c$) の係数が c であって、対角成分は 1 であり、非対角成分は 0 である行列を以下の通り定義する。この行列の座標変換的意味は座標軸の結合 (c 倍) である。

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

例題.

変数 x, y, z で構成されるつぎのベクトル $\mathbf{b} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ と基本行列 $R_3(1, 3; 2)$ の積を求めよ。

$$\begin{aligned} R_3(1, 3; 2)\mathbf{b} &= R_3(1, 3; 2) \begin{pmatrix} x \\ y \\ z \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \\ &= \begin{pmatrix} x + 2z \\ y \\ z \end{pmatrix} \end{aligned}$$

補足.

例として、図 9 の点群 (RGB 値 (色) は座標依存) を基本行列 $R_2(1, 2; 2) = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$ にて座標変換した時の図を、図 10 として示す。図は巻末のリスト 1 にて生成した。

座標軸 x' が座標軸 x と座標軸 y との結合 ($c = 2$) によって表されることが分かる。

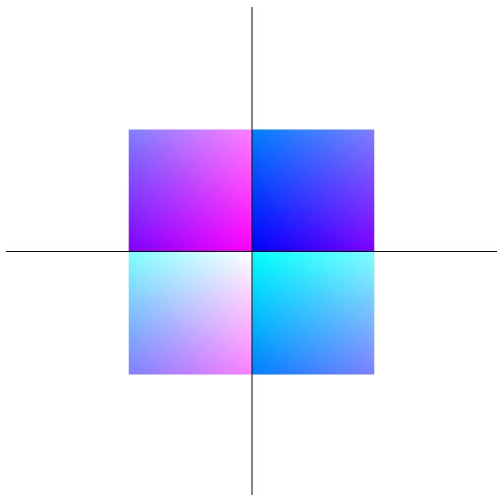


図 9: 変換前 (水平線: x 軸 / 垂線: y 軸)

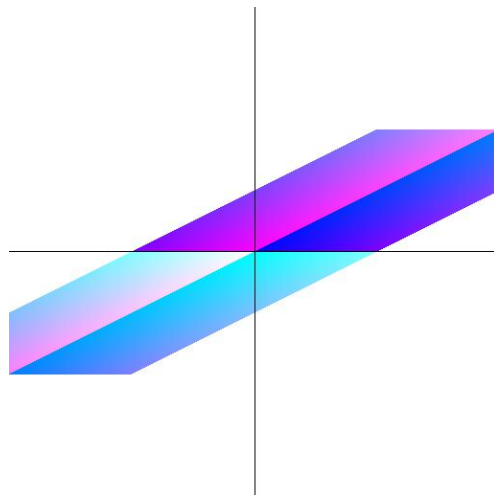


図 10: 変換後 (水平線: x' 軸 / 垂線: y' 軸)

A Python プログラム (Script)

下記に Python プログラム”rotate_rgb.py”および Python プログラム”rotate.py”を示す。

Listing 1: rotate_rgb.py

```
1  #coding: utf-8
2
3  import numpy
4  from PIL import Image, ImageOps
5
6  if __name__ == '__main__':
7      rotate = [[1, 0], [0, 1]]
8      output_fname = "./outing.jpg"
9
10     boffset_x = 128
11     boffset_y = 128
12     bmax_x = 256
13     bmax_y = 256
14     aoffset_x = 128 * 2
15     aoffset_y = 128 * 2
16     amax_x = 256 * 2
17     amax_y = 256 * 2
18     bpic = numpy.empty((bmax_x, bmax_y, 3), dtype = "uint8")
19     apic = numpy.empty((amax_x, amax_y, 3), dtype = "uint8")
20
21
22     for i in range(0, bmax_x, 1):
23         for j in range(0, bmax_y, 1):
24             bpic[i, j, 0] = i - 128 #折り返される。(uint8)
25             bpic[i, j, 1] = j - 128 #折り返される。(uint8)
26             bpic[i, j, 2] = 255
27
28     apic[ : , : , : ] = 255
29
30     for i in range(0, bmax_x, 1):
31         for j in range(0, bmax_y, 1):
32             x = rotate[0][0] * (i - boffset_x) + rotate[0][1] * (j -
                boffset_y)
33             y = rotate[1][0] * (i - boffset_x) + rotate[1][1] * (j -
                boffset_y)
34             x = int(x) + aoffset_x
35             y = int(y) + aoffset_y
```

```

36         if( 0 <= x and x < amax_x and 0 <= y and y < amax_y):
37             apic[x, y, :] = bpic[i, j, :]
38
39     apic[aoffset_x, :, :] = (0, 0, 0)
40     apic[:, aoffset_y, :] = (0, 0, 0)
41
42     output_pic = Image.fromarray(apic)
43     output_pic = output_pic.rotate(90, expand = True)
44     output_pic.save(output_fname)
45
46     print("出力画像ファイル:" + output_fname)
47     print("下記行列にて一次変換が完了しました。")
48     print( "(" + str(rotate[0][0]) + "," + str(rotate[0][1]) + "
           )\n" + "(" + str(rotate[1][0]) + "," + str(rotate[1][1]) + "
           ")")

```

Listing 2: rotate.py

```

1  #coding: utf-8
2
3  import numpy
4  from PIL import Image, ImageOps
5  import math
6
7  if __name__ == '__main__':
8      cos = math.cos
9      sin = math.sin
10     pi = math.pi
11     theta = pi / 6
12     rotate = [[cos(theta), -sin(theta)], [sin(theta), cos(theta)
13         ]]
14     input_fname = "./img.jpg"
15     output_fname = "./outimg.jpg"
16
17     input_pic = Image.open(input_fname)
18
19     input_pic = input_pic.rotate(-90, expand = True)
20     bpic = numpy.array(input_pic)
21     size = bpic.shape
22     offset_x = size[0] * 2

```

```

22  offset_y = size[1] * 2
23  max_x = size[0] * 4
24  max_y = size[1] * 4
25  apic = numpy.empty((max_x , max_y, size[2]), dtype = "uint8")
26
27  apic[ : , : , : ] = 255
28  apic[offset_x , : , : ] =(0, 0, 0)
29  apic[ : , offset_y , : ] =(0, 0, 0)
30  for i in range(0, size[0], 1):
31      for j in range(0, size[1], 1):
32          x = rotate[0][0] * i + rotate[0][1] * j + offset_x
33          y = rotate[1][0] * i + rotate[1][1] * j + offset_y
34          x = int(x)
35          y = int(y)
36          if( 0 <= x and x < max_x and 0 <= y and y < max_y):
37              apic[x, y, :] = bpic[i, j, :]
38
39  output_pic = Image.fromarray(apic)
40  output_pic = output_pic.rotate(90, expand = True)
41  output_pic.save(output_fname)
42  print("入力画像ファイル:" + input_fname)
43  print("出力画像ファイル:" + output_fname)
44  print("下記行列にて一次変換が完了しました。")
45  print( "(" + str(rotate[0][0]) + "," + str(rotate[0][1]) + "
           )\n" + "(" + str(rotate[1][0]) + "," + str(rotate[1][1]) +
           ")" )

```

また、実行例を下記に示す。

実行例

```

1  $ python rotate_rgb.py
2  出力画像ファイル: ./outimg.jpg
3  下記行列にて一次変換が完了しました。
4  (1,0)
5  (0,1)
6  $ python rotate.py
7  入力画像ファイル: ./img.jpg
8  出力画像ファイル: ./outimg.jpg
9  下記行列にて一次変換が完了しました。
10 (0.866025403784,-0.5)
11 (0.5,0.866025403784)

```

参考文献

- [1] 斉藤正彦. 線形代数入門. 東京大学出版会, 1966.