

[〆切] 2020/12/16 19:00

[IX. 固有値問題と特異値分解 (1)]

IX-B. 表 1 にある 8 匹のポケモンのステータスのデータから 3×3 の相関行列を構成し、ベキ乗法を用いて最大固有値 λ_1 とそれに対応する固有ベクトル u_1 を求めるプログラムを作成せよ。固有ベクトルは長さを 1 に正規化し、値は有効数字 4 衔で 5 衔目を四捨五入して求めよ。

※ 固有ベクトルを決める際に残る向きの自由度はどちらの方向を選んでも構わない。

ポケモン	HP	攻撃	防御
ピカチュウ	111	112	96
ライチュウ	155	193	151
イーブイ	146	104	114
コイキング	85	29	85
ギャラドス	216	237	186
カビゴン	330	190	169
ミュウ	225	210	210
ミュウツー	214	300	182

表 1 「ポケモン GO」に出てくる 8 匹のポケモンのステータス。データは以下のサイトを参照した
(<https://pokemongo.gamewith.jp/article/show/35945>)。

(I) ポケモンの数を $n=8$ とおき、 i 番目 ($i=1, 2, \dots, n$) のポケモンの HP, 攻撃, 防御

のステータスをそれぞれ h_i , a_i , d_i とおくと、平均値 \bar{h} , \bar{a} , \bar{d} は

$$\left\{ \begin{array}{l} \bar{h} = \frac{1}{n} \sum_{i=1}^n h_i = \frac{1482}{8} = 185.3 \\ \bar{a} = \frac{1}{n} \sum_{i=1}^n a_i = \frac{1375}{8} = 171.9 \\ \bar{d} = \frac{1}{n} \sum_{i=1}^n d_i = \frac{1193}{8} = 149.1 \end{array} \right.$$

となる。

また、 h_i , a_i , d_i の不偏分散は s_h^2 , s_a^2 , s_d^2 はである。

$$\left\{ \begin{array}{l} s_h^2 = \frac{1}{n-1} \sum_{i=1}^n (h_i - \bar{h})^2 = \frac{1}{n-1} \left\{ \left(\sum_{i=1}^n h_i^2 \right) - 2 \bar{h} \left(\sum_{i=1}^n h_i \right) + n \bar{h}^2 \right\} \\ \quad \text{L } h_i^2 - 2 \bar{h} h_i + \bar{h}^2 \\ \quad \text{R } n \bar{h} \\ = \frac{1}{n-1} \left\{ \left(\sum_{i=1}^n h_i^2 \right) - n \bar{h}^2 \right\} = 6046 \\ \\ s_a^2 = \frac{1}{n-1} \left\{ \left(\sum_{i=1}^n a_i^2 \right) - n \bar{a}^2 \right\} = 7355 \\ \\ s_d^2 = \frac{1}{n-1} \left\{ \left(\sum_{i=1}^n d_i^2 \right) - n \bar{d}^2 \right\} = 2102 \end{array} \right.$$

となる。

```
import numpy as np
n = 8
h = np.array([111, 155, 146, 85, 216, 330, 225, 214])
a = np.array([112, 193, 104, 29, 237, 190, 210, 300])
d = np.array([96, 151, 114, 85, 186, 169, 210, 182])
h_ave = 0
a_ave = 0
d_ave = 0
for i in range(n):
    h_ave += h[i]
    a_ave += a[i]
    d_ave += d[i]
h_ave = h_ave / n
a_ave = a_ave / n
d_ave = d_ave / n
```

これらの平均をとることで、それらの標準偏差 s_h, s_a, s_d を求めると

$$\left\{ \begin{array}{l} S_h = \sqrt{s_h^2} = 77.76 \\ S_a = \sqrt{s_a^2} = 85.77 \\ S_d = \sqrt{s_d^2} = 45.85 \end{array} \right.$$

```

s_h = 0
s_a = 0
s_d = 0
for i in range(n):
    s_h += h[i]**2
    s_a += a[i]**2
    s_d += d[i]**2
s_h = np.sqrt((s_h - n*h_ave**2) / (n-1))
s_a = np.sqrt((s_a - n*a_ave**2) / (n-1))
s_d = np.sqrt((s_d - n*d_ave**2) / (n-1))

```

つづいて、 $\pm s$ (= 3 倍数の不偏共分散 S_{ha}, S_{ad}, S_{dh}) は?

$$\left\{ \begin{array}{l} S_{ha}^2 = \frac{1}{n-1} \left\{ \sum_{i=1}^n (h_i - \bar{h})(a_i - \bar{a}) \right\} = \frac{1}{n-1} \left\{ \sum_{i=1}^n h_i a_i - \bar{h} \left(\sum_{i=1}^n a_i \right) - \bar{a} \left(\sum_{i=1}^n h_i \right) + n \bar{h} \bar{a} \right\} \\ \quad \text{左} \bar{h} \bar{a} \quad \text{右} n \bar{h} \\ = \frac{1}{n-1} \left\{ \left(\sum_{i=1}^n h_i a_i \right) - n \bar{h} \bar{a} \right\} = 4374 \\ S_{ad}^2 = \frac{1}{n-1} \left\{ \left(\sum_{i=1}^n a_i d_i \right) - n \bar{a} \bar{d} \right\} = 3437 \\ S_{dh}^2 = \frac{1}{n-1} \left\{ \left(\sum_{i=1}^n d_i h_i \right) - n \bar{d} \bar{h} \right\} = 2724 \end{array} \right.$$

```

s2_ha = 0
s2_ad = 0
s2_dh = 0
for i in range(n):
    s2_ha += h[i] * a[i]
    s2_ad += a[i] * d[i]
    s2_dh += d[i] * h[i]
s2_ha = (s2_ha - n*h_ave*a_ave) / (n-1)
s2_ad = (s2_ad - n*a_ave*d_ave) / (n-1)
s2_dh = (s2_dh - n*d_ave*h_ave) / (n-1)

```

つづいて、これらの計算をもとに相関行列は

$$A = \begin{bmatrix} h & a & d \\ h & \frac{S_{ha}^2}{S_h S_a} & \frac{S_{dh}^2}{S_a S_d} \\ a & \frac{S_{ha}^2}{S_h S_a} & \frac{S_{ad}^2}{S_a S_d} \\ d & \frac{S_{dh}^2}{S_a S_d} & \frac{S_{ad}^2}{S_a S_d} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0.6559 & 0.7642 \\ 0.6559 & 1 & 0.8742 \\ 0.7642 & 0.8742 & 1 \end{bmatrix}$$

```

A = np.array([
    [1, s2_ha/(s_h * s_a), s2_dh/(s_d * s_h)],
    [s2_ha/(s_h * s_a), 1, s2_ad/(s_a * s_d)],
    [s2_dh/(s_d * s_h), s2_ad/(s_a * s_d), 1]])

```

キーフレーズ

データの正規化を行うと
分散・共分散行列は相関行列
になります。

つづいて、

初期のタミ-ベクトルを $\psi = [1.0, 0.0, 0.0]$ とベキ乗法のプログラムは
以下のようになります。

```
u = np.array([1.0,0.0,0.0])
y = np.array([1.0,0.0,0.0])
m = 3

for k in range(10):
    norm = 0
    for i in range(m):
        norm += u[i]**2
    norm = np.sqrt(norm)

    for i in range(m):
        y[i] = u[i] / norm

    for i in range(m):
        u[i] = 0
        for j in range(m):
            u[i] += A[i][j] * y[j]
    eigen_val = 0

    for i in range(m):
        eigen_val += y[i] * u[i]
    print(k, y, u, eigen_val)
    residue = 0
    for i in range(m):
        residue += (u[i] - eigen_val * z[i])**2
    residue = np.sqrt(residue)
    if residue <= 10**-15:
        break
```

```
0 [1. 0. 0.] [1. 0.6559015 0.76423727] 1.0
1 [0.7045984 0.46214715 0.53848036] [1.41924818 1.39502432 1.48096159] 2.44217524582853
2 [0.57212842 0.56236328 0.59700638] [1.39723786 1.45951559 1.52585628] 2.5311233895414547
3 [0.55184578 0.57644267 0.60264424] [1.39049858 1.46522009 1.52830135] 2.5329781643104314
4 [0.54895444 0.57845372 0.60335754] [1.38947142 1.46595825 1.528563] 2.5330155138018484
5 [0.54854431 0.57874026 0.60345576] [1.38932429 1.46606165 1.52859827] 2.5330162657286674
6 [0.54848613 0.57878099 0.60346958] [1.38930339 1.4660763 1.52860323] 2.5330162808671717
7 [0.54847788 0.57878677 0.60347154] [1.38930043 1.46607838 1.52860393] 2.533016281171957
8 [0.54847671 0.57878759 0.60347182] [1.38930001 1.46607868 1.52860403] 2.5330162811780927
9 [0.54847654 0.57878777 0.60347185] [1.38929995 1.46607872 1.52860405] 2.5330162811782158
```

$$\lambda_1 = 2.533, \mathbf{u}_1 = [0.5485, 0.5788, 0.6035]^T$$

#

④ データの正規化について

- 元のデータの平均値を0、分散を1とするなどの変換を「データの正規化」と呼ぶ。
- 元のデータが x_i ($1 \leq i \leq n$) で与えられた時、

$$x'_i = \frac{x_i - \bar{x}}{s_x} \quad (\bar{x} \text{--- } x_i \text{の平均}, s_x \text{--- } x_i \text{の標準偏差})$$

という変数変換がデータの正規化。

- $|X-B|$ の PDTM-GO の解析で正規化を行う場合

$$h'_i = \frac{h_i - \bar{h}}{s_h}, \quad a'_i = \frac{a_i - \bar{a}}{s_a}, \quad d'_i = \frac{d_i - \bar{d}}{s_d}$$

計算。

	h	a	d		h'	a'	d'
ピカチュウ	111	112	96	ピカチュウ	-0.954893	-0.698119	-1.158775
ライチュウ	155	193	151	ライチュウ	-0.389030	0.246309	0.040898
イーブイ	146	104	114	イーブイ	-0.504775	-0.791396	-0.766155
コイキング	85	29	85	コイキング	-1.289266	-1.665866	-1.398710
ギャラドス	216	237	186	ギャラドス	0.395461	0.759332	0.804326
カビゴン	330	190	169	カビゴン	1.861559	0.211330	0.433518
ミュウ	225	210	210	ミュウ	0.511205	0.444522	1.327820
ミュウツー	214	300	182	ミュウツー	0.369740	1.493887	0.717077

- h'_i, a'_i, d'_i で「分散・共分散行列」を作ると相関行列になれる。

$$\bar{h}' = \frac{1}{n} \left\{ \sum_{i=1}^n \frac{h_i - \bar{h}}{s_h} \right\} = \frac{1}{s_h} \left\{ \frac{1}{n} \sum_{i=1}^n h_i - \frac{1}{n} \times n \bar{h} \right\} = \frac{1}{s_h} (\bar{h} - \bar{h}) = 0$$

$$\bar{a}' = \bar{d}' = 0$$

$$S_{h'}^2 = \frac{1}{n-1} \left\{ \sum_{i=1}^n (h'_i - \bar{h}')^2 \right\} = \frac{1}{n-1} \left\{ \sum_{i=1}^n \frac{(h_i - \bar{h})^2}{s_h^2} \right\} = \frac{\frac{1}{n-1} \sum_{i=1}^n (h_i - \bar{h})^2}{s_h^2} = \frac{s_h^2}{s_h^2} = 1$$

$$S_{a'}^2 = S_{d'}^2 = 1$$

$$S_{ha'}^2 = \frac{1}{n-1} \left\{ \left(\sum_{i=1}^n h'_i a'_i \right) - n \bar{h}' \bar{a}' \right\} = \frac{1}{n-1} \frac{\sum_{i=1}^n (h_i - \bar{h})(a_i - \bar{a})}{s_h s_a} = \frac{s_{ha}^2}{s_h s_a}$$

$$S_{a'd'}^2 = \frac{S_{ad}^2}{s_a s_d}, \quad S_{d'h'}^2 = \frac{S_{dh}^2}{s_d s_h}$$

$$C = \begin{bmatrix} S_h^2 & S_{ha}^2 & S_{dh}^2 \\ S_{ad}^2 & S_a^2 & S_{d'a'}^2 \\ S_{dh}^2 & S_{d'a'}^2 & S_{a'}^2 \end{bmatrix} = \begin{bmatrix} 1 & \frac{S_{ha}^2}{s_h s_a} & \frac{S_{dh}^2}{s_d s_h} \\ \frac{S_{ha}^2}{s_h s_a} & 1 & \frac{S_{ad}^2}{s_a s_d} \\ \frac{S_{dh}^2}{s_d s_h} & \frac{S_{ad}^2}{s_a s_d} & 1 \end{bmatrix} \quad [\text{相関行列}]$$

[分散・共分散行列]

[〆切] 2020/12/16 19:00

[X. 固有値問題と特異値分解 (2)]

X-A. IX-B で作成した相関行列について考える逆反復法により残りの固有対 $(\lambda_2, \mathbf{u}_2), (\lambda_3, \mathbf{u}_3)$ を求める問題を考える。逆反復法で指定する固有値の近似値 σ の値を $0, \lambda_1 \times 1/200, \lambda_1 \times 2/200, \dots, \lambda_1 \times 199/200$ と変えながら問 1 で求めた相関行列の固有値・固有ベクトルを求めるプログラムを作成し残りの 2 つの固有対を求めよ。固有ベクトルは長さを 1 に正規化し、値は有効数字 4 桁で 5 桁目を四捨五入して求めよ。

※ 固有ベクトルを決める際に残る向きの自由度はどちらの方向を選んでも構わない。

```

u = np.array([1.0, 0.0, 0.0])
y = np.array([0.0, 0.0, 0.0])
z = np.array([0.0, 0.0, 0.0])
L = np.array([[0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0]])
M = np.array([[0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0]])
m = 3

for s in reversed(range(200)):
    sigma = 2.5330162811782184 * s / 200
    B = copy.copy(-A)
    B[0][0] = sigma + B[0][0]
    B[1][1] = sigma + B[1][1]
    B[2][2] = sigma + B[2][2]
    for k in range(m):
        for i in range(k+1, m):
            M[i][k] = B[i][k]/B[k][k]
            B[i][k] = 0
        for j in range(k+1, m):
            B[j][j] = B[j][j] - M[i][k]*B[k][j]
            L[i][k] = M[i][k]

    for k in range(200):
        norm = 0
        for i in range(m):
            norm += u[i]**2
        norm = np.sqrt(norm)

        for i in range(m):
            y[i] = u[i] / norm

        # Forward substitution
        z[0] = y[0]
        for i in range(1, m): # 1, 2, ..., m-1
            sum = 0
            for j in range(i):
                sum += L[i][j] * z[j]
            z[i] = y[i] - sum

    # Backward substitution
    u[m-1] = z[m-1] / B[m-1][m-1]
    for i in range(m-2, -1, -1):
        sum = 0
        for j in range(i+1, m):
            sum += B[i][j] * u[j]
        u[i] = (z[i] - sum) / B[i][i]
    eigen_val = 0
    for i in range(m):
        eigen_val += y[i] * u[i]
    residue = 0
    for i in range(m):
        residue += (u[i] - eigen_val * y[i])**2
    residue = np.sqrt(residue)
    if residue <= 10**-15:
        print("s", s, "k", k, y, u, sigma - 1/eigen_val)
        break
    
```

$B = \sigma I - A$ の準備

LU 分解

L が上三角行列

である

前进代入

後退代入

$eigen_val = \frac{1}{\sigma - \lambda}$

$\lambda = \sigma - \frac{1}{eigen_val}$

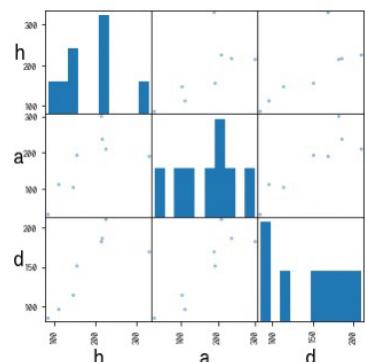
キーワード

逆反復法

①ポケモン GO ステータス解析

	h	a	d
ピカチュウ	111	112	96
ライチュウ	155	193	151
イーピイ	146	104	114
コイキング	85	29	85
ギャラドス	216	237	186
カビゴン	330	190	169
ミュウ	225	210	210
ミュウツー	214	300	182

	h'	a'	d'
ピカチュウ	-0.954893	-0.698119	-1.158775
ライチュウ	-0.389030	0.246309	0.040898
イーピイ	-0.504775	-0.791396	-0.766155
コイキング	-1.289266	-1.665866	-1.398710
ギャラドス	0.395461	0.759332	0.804326
カビゴン	1.861559	0.211330	0.433518
ミュウ	0.511205	0.444522	1.327820
ミュウツー	0.369740	1.493887	0.717077



• 相関行列

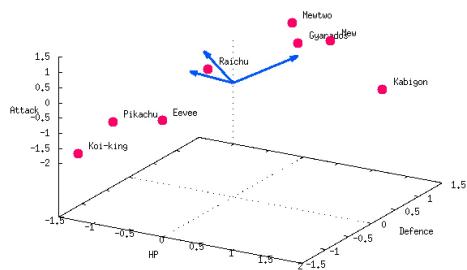
	h	a	d
h	1.000	0.656	0.764
a	0.656	1.000	0.874
d	0.764	0.874	1.000

↑ 相関係数を要素に持つ行列

● ● ● Gnplot

● ● ● Gnplot

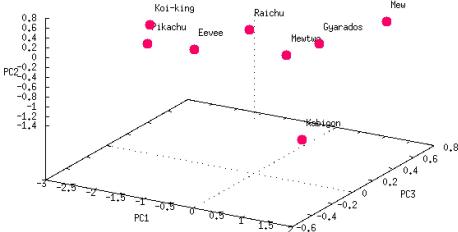
3D plot



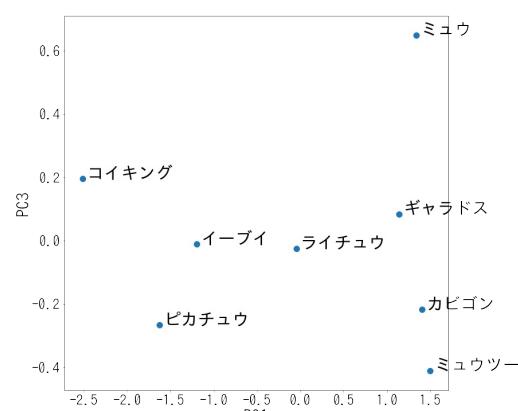
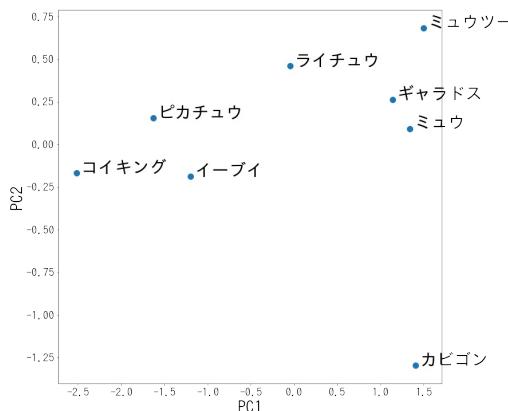
view: 60.000, 30.000 scale: 1.00000, 1.00000

● ● ● Gnplot

3D plot

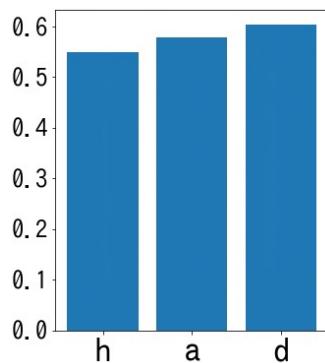


view: 60.000, 30.000 scale: 1.00000, 1.00000

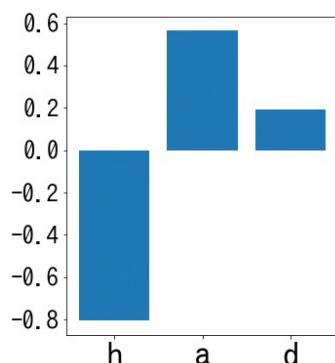


- $\lambda = [\lambda_1, \lambda_2, \lambda_3]$ に対して、 $\frac{\lambda_1}{\|\lambda\|}, \frac{\lambda_2}{\|\lambda\|}, \frac{\lambda_3}{\|\lambda\|}$ をそれぞれの固有ベクトルの **寄与率** とする

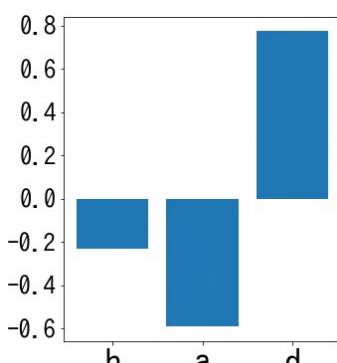
PC1 ($\frac{\lambda_1}{\|\lambda\|} = 0.8443$)



PC2 ($\frac{\lambda_2}{\|\lambda\|} = 0.1198$)



PC3 ($\frac{\lambda_3}{\|\lambda\|} = 0.03586$)



$$u_1 = [0.5485, 0.5788, 0.6035]^T$$

$$u_2 = [-0.8040, 0.5633, 0.1904]^T$$

$$u_3 = [-0.2297, -0.5896, 0.7743]^T$$