线性代数模型

特征值与特征向量

1.差分方程

如求解 Fibonacci数列的通项公式：

$$F_0 = 1, \quad F_1 = 1, \quad F_{k+2} = F_{k+1} + F_k, \quad k = 0, 1, 2, \dots$$

解法：

```
import sympy as sp

k = sp.var('k')
y = sp.Function('y')
f = y(k + 2) - y(k + 1) - y(k)
s = sp.rsolve(f, y(k), {y(0): 1, y(1): 1})
print(s)
```

2.莱斯利（Leslie）种群模型

模型形式：

$$x^{(k)} = Lx^{(k-1)}, k = 1, 2, \dots,$$
$$即：\ x^{(k)} = L^k x^{(0)}$$

其中L为莱斯利矩阵，

$$L = \begin{bmatrix} a_1 & a_2 & \cdots & a_{n-1} & a_n \\ b_1 & 0 & \cdots & 0 & 0 \\ 0 & b_2 & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}$$

其中x^k为t_k时的种群分布向量，$a_i(a_i \geq 0)$为第i个年龄组的生育率，$b_i(0 < b_i \leq 1)$为第i个年龄组的存活率。

解法：

如：已知

$$x^{(0)} = [500, 1000, 50]^T, \quad L = \begin{bmatrix} 0 & 4 & 3 \\ 0.5 & 0 & 0 \\ 0 & 0.25 & 0 \end{bmatrix}$$

求解种群分布情况。

解法：

```
import numpy as np
import sympy as sp

X0 = np.array([500, 1000, 50])
L = np.array([[0, 4, 3], [0.5, 0, 0], [0, 0.25, 0]])
Ls = sp.Matrix([[0, 4, 3], [sp.Rational(1, 2), 0, 0],
[0, sp.Rational(1, 4), 0]])  #定义与L相同的有理矩阵
lamda = sp.var('lamda')
w = Ls.eigenvals()  #求特征值
v = Ls.eigenvects()  #求特征向量
P, D = Ls.diagonalize()  #相似对角化，  P为变换矩阵
Pinv = P.inv()  #求逆矩阵
Pinv = sp.simplify(Pinv)
cc = Pinv @ X0
print('P = \n', P)
print('c = ', cc[0])
```

3.PageRank算法

模型形式：

$$W = (w_{ij})_{N\times N}, \quad r_{ij} = \sum_{j=1}^{N} w_{ij}, \quad i = 1, 2, \ldots, N,$$

$$P = (p_{ij})_{N\times N}, \quad \text{其中 } p_{ij} = \frac{w_{ij}}{r_i}, \quad i, j = 1, 2, \ldots, N,$$

$$P^T x = x, \quad \sum_{i=1}^{N} x_i = 1.$$

W为无向图的邻接矩阵，r为W的行和，P为马尔可夫链的状态转移概率矩阵，x为马尔可夫链的平稳分布。

在随机冲浪模型下，作如下改变：

$$p_{ij} = \frac{1-d}{N} + d\frac{w_{ij}}{r_i}, \quad i, j = 1, 2, \ldots, N,$$

其中d为阻尼因子，一般取$d=0.85$.

解法：

设邻接矩阵

$$W=\begin{bmatrix}0&1&0&0&0&0\\0&0&1&1&0&0\\0&0&0&1&1&1\\1&0&0&0&0&0\\0&0&0&0&0&1\\1&0&0&0&0&0\end{bmatrix}$$

求其RankPage取值。

解法:

```
import numpy as np
from scipy.sparse.linalg import eigs

M = ([0, 1, 0, 0, 0, 0],
[0, 0, 1, 1, 0, 0],
[0, 0, 0, 1, 1, 1],
[1, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 1],
[1, 0, 0, 0, 0, 0])
w = np.array(M)
r = np.sum(w, axis=1, keepdims=True)
P = (1 - 0.85) / w.shape[0] + 0.85 * w / r #非随机冲浪模型下为 P = w / r
val, vec = eigs(P.T, 1)
V = vec.real # 将特征向量vec的实部储存在V中
V = V.flatten() # 将数组V展平为一维数组
V = V / V.sum() # 标准化
print("V =", np.round(V, 4))
```

矩阵的奇异值分解及应用

任何一个矩阵都可以奇异值分解，即对角化。
设$A=(a_{ij})_{m\times n}$为一个秩为r的矩阵，则存在正交矩阵U、V，使得

$$A=U\begin{bmatrix}\Sigma&0\\0&0\end{bmatrix}V^T$$

称之为A的奇异值分解，其中$\Sigma = diag(\sigma_1, \sigma_2, \ldots, \sigma_r)$，$\sigma_i (i = 1, 2, \ldots, r)$称为A的奇异值。
从而A的范数满足：

$$||A||_F^2 = \sum_{i=1}^{r} \sigma_i^2$$

可用于稀疏矩阵的压缩
如：

```
import numpy as np
import pandas as pd

a = np.loadtex('data.txt')
u, sigma, vt = np.linalg.svd(a)
#对a进行奇异值分解，a = u @ sigma @ vt
cs = np.cumsum(sigam ** 2)
rate = cs / cs[-1]
ind = np.where(rate >= 0.9)[0][0] + 1
#ind为奇异值个数，满足信息提出率达到90%
b = np.diag(sigma[:ind]) @ u.T[:ind, :] @ a #b即为压缩所得矩阵
```