

# 层次分析法

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

## 解题步骤

### 1. 解决评价类问题

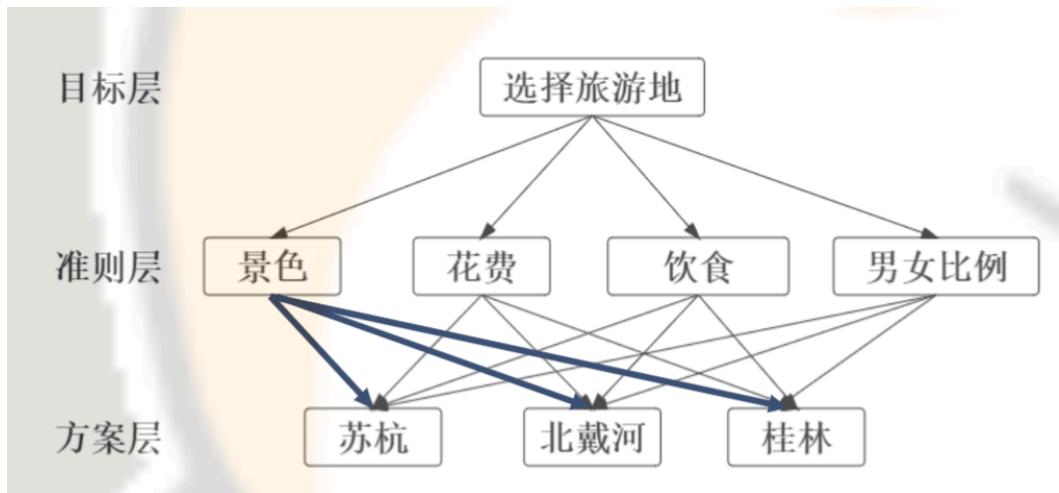
#### ► 对评价类问题进行打分

唐僧师徒一行人取经结束后，打算组织一场大乘佛法全国巡回演唱会，但是对于第一场去苏杭、北戴河还是桂林，师徒四人有了分歧……一番讨论后，他们最关心的是以下问题：

景色优美 价格亲民 饮食情况 男女比例



### 2. 画出层级结构图（目标层、准则层、方案层）（出现在论文里）



### 3. 构造判断矩阵（确定评价指标孰重孰轻）

有多少个评价标准，就有多少个判断矩阵和一个总的判断矩阵

### 4. 依照评价指标对各个方案进行打分

景色	苏杭	北戴河	桂林
苏杭	1	2	5
北戴河	1/2	1	2
桂林	1/5	1/2	1

- 判断一致性
- 求出权重，填表，求得最后得分
- 层次总排序一致性检验

## 计算一致性比例

### 层次分析法-矩阵一致性

➤ 一致性检验方法

计算一致性指标,  $CI = \frac{\lambda_{\max} - n}{n - 1}$ ,  $CI = \begin{cases} 0, & \text{有完全一致性} \\ \text{接近} 0, & \text{满意的一致性} \\ \text{越大,} & \text{一致性越差} \end{cases}$

为了衡量  $CI$  的大小, 引入随机一致性指标  $RI$ , 先构造 500 个判断矩阵  $A_1, A_2, \dots, A_{500}$  分别计算其  $\lambda_{\max}$ , 于是得到它们的一致性指标  $CI_1, CI_2, \dots, CI_{500}$

定义一致性指标  $RI = \frac{CI_1 + CI_2 + \dots + CI_{500}}{500} = \frac{\lambda_1 + \lambda_2 + \dots + \lambda_{500} - n}{n - 1}$ , 常用随机一致性指标如下表所示

$n$	1	2	3	4	5	6	7	8	9	10	11
$RI$	0	0	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49	1.51

微信公众号: 大师兄的知识库 微博: 花果山没有眼泪 QQ交流群: 653489407

---

更多优质资源欢迎关注“大师兄的知识库”微信公众号, 回复“数学建模”可获取本套课程课件和代码及所需软件安装包

大师兄的知识库

### 层次分析法-矩阵一致性

➤ 一致性检验方法

定义一致性比例  $CR = \frac{CI}{RI}$

如果  $CR < 0.1$ , 则可认为判断矩阵的一致性可以接受; 否则需要对判断矩阵进行修正。(证明见精通篇)

所以进行一致性检验需要用到以下两个公式:

一致性指标  $CI = \frac{\lambda_{\max} - n}{n - 1}$ , 一致性比例  $CR = \frac{CI}{RI} \begin{cases} < 0.1, & \text{判断矩阵一致} \\ \geq 0.1, & \text{判断矩阵不一致} \end{cases}$

如果大于 0.1 怎么办?

强行往一致性上靠, 调整成倍数关系即可

微信公众号: 大师兄的知识库 微博: 花果山没有眼泪 QQ交流群: 653489407

- 以下为 Matlab 代码:

```
CI = (Max_eig - n) / (n-1);
% Max_eig 为最大特征值, n 为判断矩阵的维度。
% 最大特征值的计算在特征值法求权重时会介绍。
```

```
RI=[0 0.0001 0.52 0.89 1.12 1.26 1.36 1.41 1.46 1.49 1.52 1.54 1.56
1.58 1.59];
% 这里的 RI 需要查表。
% 这里 n=2 时, RI=0, 我们为了避免分母为0, 将这里的第二个元素改为了很接近0
的正数, 即 0.0001。
```

```
CR=CI/RI(n);
disp('最大特征值为:');
disp(Max_eig);
disp('一致性指标CI=');disp(CI);
disp('一致性比例CR=');disp(CR);
if CR<0.10
    disp('CR<0.10, 该判断矩阵A的一致性可以接受!');
else
    disp('注意: CR >= 0.10, 该判断矩阵需要进行修改!');
end
```

## 权重的计算

### 算数平均法

**层次分析法-权重的计算**

➤ 算术平均法求权重论文表述

对于判断矩阵  $A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$ , 先将其归一化, 再将归一化的矩阵按列相加, 并将每个元素除以  $n$

得到权重向量, 即  $\omega_i = \frac{1}{n} \sum_{j=1}^n \frac{a_{ij}}{\sum_{k=1}^n a_{kj}} \quad (i=1, 2, \dots, n)$ .

以下图判断矩阵为例:

景色	苏杭	北戴河	桂林
苏杭	1	2	5
北戴河	1/2	1	2
桂林	1/5	1/2	1

## 1. 输入矩阵 A。

```
In [ ]: A = [[1, 2, 5], [1 / 2, 1, 2], [1 / 5, 1 / 2, 1]]
A = np.array(A)
print(A)
```

```
[[1.  2.  5. ]
 [0.5  1.  2. ]
 [0.2  0.5  1. ]]
```

## 2. 先将矩阵按列求和。

```
In [ ]: Sum_A = np.sum(A, axis=0)
print(Sum_A)
```

```
[1.7  3.5  8. ]
```

## 3. 再将矩阵扩大到三行。

```
In [ ]: size = len(A)
SUM_A = np.tile(Sum_A, (size, 1))
print(SUM_A)
```

```
[[1.7  3.5  8. ]
 [1.7  3.5  8. ]
 [1.7  3.5  8. ]]
```

## 4. 将 A 中的每一个元素与 SUM\_A 中的对应元素相除，得到标准化矩阵。

$$\begin{aligned} \text{第一列: 苏杭} &= \frac{1}{1+0.5+0.2} = 0.5882, \text{北戴河} = \frac{0.5}{1+0.5+0.2} = 0.2941 \\ \text{桂林} &= \frac{0.2}{1+0.5+0.2} = 0.1176 \\ \text{第二列: 苏杭} &= 0.5714, \text{北戴河} = 0.2857, \text{桂林} = 0.1428 \\ \text{第三列: 苏杭} &= 0.6250, \text{北戴河} = 0.2500, \text{桂林} = 0.1250 \end{aligned}$$

```
In [ ]: Stand_A = A / SUM_A
print(Stand_A)
```

```
[[0.58823529  0.57142857  0.625    ]
 [0.29411765  0.28571429  0.25    ]
 [0.11764706  0.14285714  0.125    ]]
```

5. 把三列的结果按行相加，再做一个算术平均值

得到算术平均值如下：

$$\text{苏杭} = \frac{0.5882 + 0.5714 + 0.6250}{3} = 0.5949, \text{北戴河} = 0.2766,$$

$$\text{桂林} = 0.1285, \text{即权重 } \omega = [0.5949 \ 0.2766 \ 0.1285]^T$$

```
In [ ]: w1 = np.sum(Stand_A, axis=1) / np.size(Stand_A, axis=1)
print(w1)
```

```
[0.59488796 0.27661064 0.1285014 ]
```

## 特征值法

1. 先求特征值和特征向量

```
[V,D] = eig(A); %% D是特征值，V是特征向量
Max_eig = max(max(D));% 先按列求最大值，得到行向量，再从这个向量里面求最大值
[r,c]=find(D == Max_eig , 1);
```

```
In [ ]: V, D = np.linalg.eig(A)
```

```
print("特征向量:")
print(D)
print("\n特征值:")
print(V)
```

特征向量：

```
[[ -0.89021421+0.j          -0.89021421+0.j          -0.89021421-0.j          ]
 [ -0.41320083+0.j          0.20660042+0.35784242j    0.20660042-0.35784242j    ]
 [ -0.19179084+0.j          0.09589542-0.16609574j    0.09589542+0.16609574j    ]]
```

特征值：

```
[ 3.00553511e+00+0.j          -2.76755587e-03+0.12895082j
 -2.76755587e-03-0.12895082j]
```

2. 求出最大特征值对应的特征向量，并确认最大特征值的位置

matlab 代码：

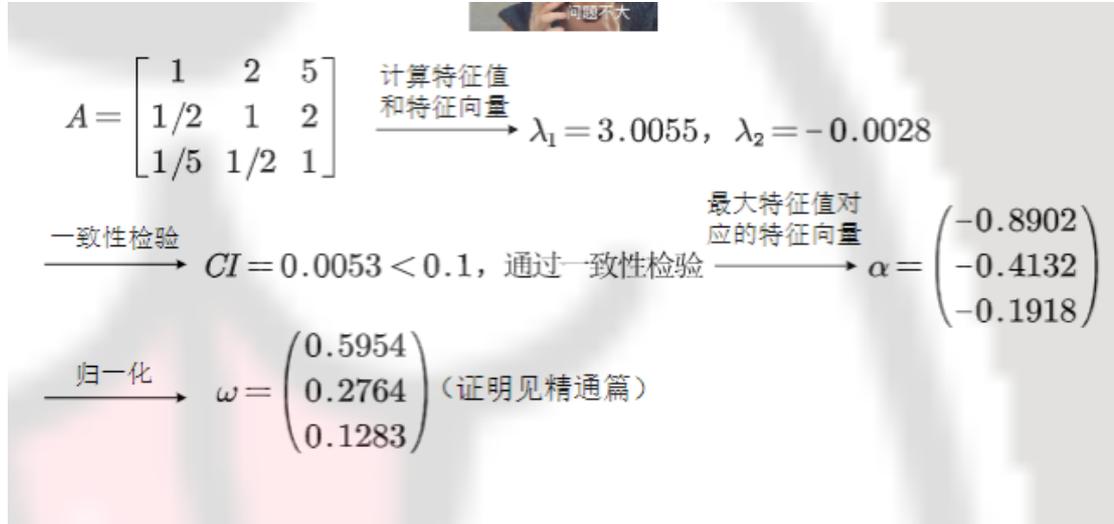
```
Max_eig = max(max(D));
% 先按列求最大值，得到行向量，再从这个向量里面求最大值
% 结果是 3.0055
```

```
[r,c]=find(D == Max_eig , 1);
% 确认最大特征值的位置
% r = 1
% c = 1
% 在第一行第一列
```

3. 把第  $c$  列中的元素除以第  $c$  列所有元素的和

```
w2 = V(:,c) ./ sum(V(:,c));
disp(w2)
```

总过程:



```
In [ ]: # 求特征值中的最大值
Max_eig = np.max(V)
Max_eig
# 找到特征值中最大值对应的行和列
r = np.where(V == Max_eig)

print("\n特征值中的最大值:", Max_eig)
```

特征值中的最大值: (3.0055351117384976+0j)

```
In [ ]: w2 = D[:, r] / np.sum(D[:, r])
w2
```

```
Out [ ]: array([[0.59537902-0.j]],
               [[0.27635046-0.j]],
               [[0.12827052-0.j]])
```