

Resolvendo o AHP em ambiente Scilab

Uma boa ferramenta a se empregar na metodologia AHP, que permite calcular de maneira exata os autovetores e autovalores dos problemas, e a fazer alguns cálculos para análises de sensibilidade, é o pacote computacional Scilab. O Scilab é um *software* francês de computação numérica semelhante ao mais conhecido Matlab, de grande aplicação na área de engenharia, e é distribuído gratuitamente pela internet.

O exemplo do Capítulo 5 da escolha da casa, neste capítulo ajuda a ilustrar algumas funções de álgebra linear do Scilab e a mostrar como se pode manipular matrizes inconsistentes até que se tornem quase consistentes.

A representação de uma matriz no Scilab é feita entre colchetes, por linhas, separando cada uma da próxima por ponto e vírgula (;) e os elementos uns dos outros por vírgula (,). É possível escrever cada linha da matriz em uma linha de comando, para facilitar a visualização. Quando a última linha de comando contém ponto e vírgula no final, os dados são apenas guardados pelo programa. Se não houver ponto e vírgula, o programa retorna alguma resposta.

Representando a matriz original do problema:

```
-->A=[1,5,3,7,6,6,1/3,1/4;  
-->1/5,1,1/3,5,3,3,1/5,1/7;  
-->1/3,3,1,6,3,4,6,1/5;  
-->1/7,1/5,1/6,1,1/3,1/4,1/7,1/8;  
-->1/6,1/3,1/3,3,1,1/2,1/5,1/6;  
-->1/6,1/3,1/4,4,2,1,1/5,1/6;  
-->3,5,1/6,7,5,5,1,1/2;  
-->4,7,5,8,6,6,2,1]
```

O retorno é:

A =

1.	5.	3.	7.	6.	6.	0.3333333	0.25
0.2	1.	0.3333333	5.	3.	3.	0.2	0.1428571
0.3333333	3.	1.	6.	3.	4.	6.	0.2
0.1428571	0.2	0.1666667	1.	0.3333333	0.25	0.1428571	0.125
0.1666667	0.3333333	0.3333333	3.	1.	0.5	0.2	0.1666667
0.1666667	0.3333333	0.25	4.	2.	1.	0.2	0.1666667
3.	5.	0.1666667	7.	5.	5.	1.	0.5
4.	7.	5.	8.	6.	6.	2.	1.

A função que extrai um vetor de autovalores e também os autovetores (se existirem) da matriz é a função *spec*. A entrada da função será a matriz *A*, e é preciso nomear as duas saídas. Vamos chamar de x_0 os autovetores resultantes e de c_0 o conjunto de vetores de autovalores.

-->[x0,c0]=spec(A)

c0 =

column 1 to 4

9.6688873	0	0	0
0	- 0.9545952 + 3.1931413i	0	0
0	0	- 0.9545952 - 3.1931413i	0
0	0	0	0.4603379 + 2.3694481i
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

column 5 to 8

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0.4603379 - 2.3694481i	0	0	0
0	- 0.2276361 + 0.3654739i	0	0
0	0	- 0.2276361 - 0.3654739i	0
0	0	0	- 0.2251002

x0 =

column 1 to 4

- 0.3778504	0.1888771 + 0.3752729i	0.1888771 - 0.3752729i	0.1224379 + 0.1919009i
- 0.1178768	- 0.0014761 - 0.0024341i	- 0.0014761 + 0.0024341i	- 0.1743943 + 0.0058600i
- 0.4108114	- 0.6220051	- 0.6220051	0.1415575 + 0.4114070i
- 0.0382353	0.0095593 - 0.0067886i	0.0095593 + 0.0067886i	0.0225484 - 0.0384256i
- 0.0677960	0.0107084 + 0.0071692i	0.0107084 - 0.0071692i	0.0032090 - 0.0681905i
- 0.0793235	0.0025858 - 0.0133876i	0.0025858 + 0.0133876i	- 0.0707350 - 0.0735220i
- 0.3643457	0.1622133 - 0.3496132i	0.1622133 + 0.3496132i	- 0.0993979 + 0.1268430i
- 0.7276467	0.4206676 + 0.3325719i	0.4206676 - 0.3325719i	0.8278997

column 5 to 8

0.1224379 - 0.1919009i	- 0.1635849 + 0.2496050i	- 0.1635849 - 0.2496050i	- 0.2814098
- 0.1743943 - 0.0058600i	- 0.0192655 - 0.2606485i	- 0.0192655 + 0.2606485i	0.1073365
0.1415575 - 0.4114070i	- 0.0935409 + 0.0903740i	- 0.0935409 - 0.0903740i	- 0.1281186
0.0225484 + 0.0384256i	- 0.0502594 - 0.0680043i	- 0.0502594 + 0.0680043i	- 0.0970705
0.0032090 + 0.0681905i	- 0.0390750 + 0.1369817i	- 0.0390750 - 0.1369817i	0.1875855
- 0.0707350 + 0.0735220i	0.1434757 + 0.0500983i	0.1434757 - 0.0500983i	- 0.0813212
- 0.0993979 - 0.1268430i	- 0.0227721 + 0.0583831i	- 0.0227721 - 0.0583831i	0.0150408
0.8278997	0.8782069	0.8782069	0.9172910

Observando os valores de c_0 , a primeira coluna é a que contém o maior valor. O elemento não-nulo é o autovalor máximo da matriz em questão, ou seja, λ_{\max} . Então pega-se essa coluna onde o elemento não-nulo é o maior, usando-se a propriedade de indicar a linha por (:) e a coluna por 1. O uso dos dois pontos simboliza que não se quer tomar uma linha específica, e sim todas. Especificando a coluna, tem-se como resultado todo o vetor, isto é, toda a coluna. Nomeamos como λ_0 (lambda0) o novo vetor.

```
--> lambda0=c0(:,1)
```

```
lambda0 =
```

```
9.6688873
```

```
0
```

```
0
```

```
0
```

```
0
```

```
0
```

```
0
```

```
0
```

Ou, alternativamente, toma-se de uma vez o elemento da primeira linha e primeira coluna, como λ_{\max} .

```
--> lambdamax=c0(1,1)
```

```
lambdamax =
```

```
9.6688873
```

Por ser a primeira coluna de c_0 a que contém o autovalor máximo, a primeira coluna de x_0 será o autovetor principal da matriz.

```
--> x=x0(:,1)
```

```
x =
```

```
- 0.3778504
```

```
- 0.1178768
```

```
- 0.4108114
```

```
- 0.0382353
```

```
- 0.0677960
```

```
- 0.0793235
```

```
- 0.3643457
```

```
- 0.7276467
```

Como queremos ter um vetor de prioridades em que a soma dos valores é igual a 1, normalizamos o vetor x , utilizando a função *sum* do Scilab e depois uma divisão simples.

```
-->soma0=sum(x)
```

```
soma0 =
```

```
- 2.183886
```

```
-->xnorm=x/soma0
```

```
xnorm =
```

```
0.1730175
```

```
0.0539757
```

```
0.1881103
```

```
0.0175079
```

```
0.0310437
```

```
0.0363222
```

```
0.1668337
```

```
0.3331890
```

Vamos chamar, como nos outros capítulos, o autovetor de w :

```
-->w=xnorm;
```

Após encontrar o vetor de prioridades w , queremos saber se os julgamentos são consistentes. Para saber a ordem da matriz, é possível usar a função *size*, e como ela retorna duas colunas iguais, toma-se a primeira.

```
-->n=size(A);
```

```
-->N=n(1)
```

```
N =
```

```
8.
```

Aplica-se simplesmente a fórmula do índice de inconsistência CI:

$$\rightarrow CI = (\lambda_0 - N) / (N - 1)$$

CI =

0.2384125

Retira-se da tabela de Saaty e Mariano o valor de RI:

$$\rightarrow RI = 1.4;$$

E finalmente calcula-se a razão de inconsistência CR.

$$\rightarrow CR = CI / RI$$

CR =

0.1702946

Fica comprovado, assim, que a matriz não tem um nível de consistência aceitável. Continuamos no exemplo para demonstrar os métodos de melhoria de consistência utilizando o Scilab. O primeiro método era baseado em encontrar o elemento que causa maior variação no λ_{\max} através do uso de derivadas parciais. Para isto, além do w era necessário calcular v , o autovetor da matriz A transposta.

A matriz transposta A^T é:

$$\rightarrow AT = A'$$

AT =

1.	0.2	0.3333333	0.1428571	0.1666667	0.1666667	3.	4.
5.	1.	3.	0.2	0.3333333	0.3333333	5.	7.
3.	0.3333333	1.	0.1666667	0.3333333	0.25	0.1666667	5.
7.	5.	6.	1.	3.	4.	7.	8.
6.	3.	3.	0.3333333	1.	2.	5.	6.
6.	3.	4.	0.25	0.5	1.	5.	6.
0.3333333	0.2	6.	0.1428571	0.2	0.2	1.	2.
0.25	0.1428571	0.2	0.125	0.1666667	0.1666667	0.5	1.

Da mesma maneira que se calcula w , procede-se para calcular v , chegando a:

```
-->v=xTnorm
```

```
v =
```

```
0.0468590
```

```
0.1168184
```

```
0.0520582
```

```
0.3494764
```

```
0.1902822
```

```
0.1656790
```

```
0.0590599
```

```
0.0197668
```

Tendo n , A , w e v , utiliza-se a fórmula (7) (página 32). Para isto, cria-se a função *deriva*.

```
-->function[matder]=deriva(n,A,w,v)
```

```
-->for i=1:n-1 do
```

```
-->for j=i+1:n do
```

```
-->matder(i,j)=v(i)*w(j)-A(j,i)*A(j,i)*v(j)*w(i)
```

```
-->end
```

```
-->end
```

```
-->endfunction
```

Considerando que A , w e v já estão definidos, aplica-se a função *deriva* com $n = 8$.

```
-->der=deriva(8,A,w,v)
```

```
der =
```

0.	0.0017208	0.0078139	- 0.0004136	0.0005402	0.0009058	- 0.0841479	- 0.0391071
0.	0.	- 0.0033142	0.0012907	0.0024853	0.0032495	- 0.0602057	- 0.0133568
0.	0.	0.	- 0.0009147	- 0.0023610	- 0.0000570	0.0083765	- 0.0756132
0.	0.	0.	0.	- 0.0191339	- 0.0337173	0.0076377	0.0942929
0.	0.	0.	0.	0.	- 0.0136617	- 0.0140905	0.0413091
0.	0.	0.	0.	0.	0.	- 0.0259888	0.0293554
0.	0.	0.	0.	0.	0.	0.	0.0064870

Foi observado que o elemento com maior derivada parcial era o a_{48} , e em seguida apresentado o outro método, de Harker. No método de Harker, calculava-se a perturbação de cada elemento em relação à matriz consistente. Para isto, é necessário criar a função *perturba*.

```
-->function[epsilon]=perturba(n,A,w)
-->for i=1:n do
-->for j=1:n do
-->epsilon(i,j)=A(i,j)*w(j)/w(i)
-->end
-->end
-->endfunction
```

Com a função *perturba*, constrói-se a matriz de perturbações.

```
-->epsilon=perturba(8,A,w)
epsilon =
```

1.	1.5598335	3.2616984	0.7083405	1.0765512	1.2596021	0.3214197	0.4814383
0.6410940	1.	1.1616975	1.6218317	1.7254264	2.0188085	0.6181808	0.8818492
0.3065887	0.8608093	1.	0.5584351	0.4950877	0.7723596	5.3213577	0.3542485
1.4117505	0.6165868	1.7907183	1.	0.5910418	0.5186544	1.3612932	2.3788475
0.9288922	0.5795669	2.0198441	1.6919278	1.	0.5850172	1.0748313	1.7888171
0.7939015	0.4953417	1.2947337	1.928066	1.7093513	1.	0.9186321	1.5288584
3.111197	1.6176498	0.1879220	0.7345956	0.9303786	1.088575	1.	0.9985662
2.0771094	1.1339807	2.8228768	0.4203716	0.5590287	0.6540828	1.0014358	1.

Neste passo, observou-se que o elemento com a maior perturbação é o a_{37} . De acordo com o método, o elemento a_{37} e seu complementar a_{73} devem ser substituídos por 0 e os elementos opostos a eles (a_{33} e a_{77}) por 2.

Rebatizamos a matriz A de A_{modif} (A modificada), para depois modificar cada um dos quatro termos.

```
-->Amodif=A
Amodif =
```


1.	5.	3.	7.	6.	6.	0.3333333	0.25
0.2	1.	0.3333333	5.	3.	3.	0.2	0.1428571
0.3333333	3.	1.	6.	3.	4.	6.	0.2
0.1428571	0.2	0.1666667	1.	0.3333333	0.25	0.1428571	0.125
0.1666667	0.3333333	0.3333333	3.	1.	0.5	0.2	0.1666667
0.1666667	0.3333333	0.25	4.	2.	1.	0.2	0.1666667
3.	5.	0.1666667	7.	5.	5.	1.	0.5
4.	7.	5.	8.	6.	6.	2.	1.

Basta dizer ao programa o novo valor de um determinado elemento, e ele retorna a matriz completa com apenas aquele elemento modificado. Por exemplo:

-->Amodif(3,7)=0

Amodif =

1.	5.	3.	7.	6.	6.	0.3333333	0.25
0.2	1.	0.3333333	5.	3.	3.	0.2	0.1428571
0.3333333	3.	1.	6.	3.	4.	0.	0.2
0.1428571	0.2	0.1666667	1.	0.3333333	0.25	0.1428571	0.125
0.1666667	0.3333333	0.3333333	3.	1.	0.5	0.2	0.1666667
0.1666667	0.3333333	0.25	4.	2.	1.	0.2	0.1666667
3.	5.	0.1666667	7.	5.	5.	1.	0.5
4.	7.	5.	8.	6.	6.	2.	1.

Fazendo as quatro modificações, o último retorno é:

Amodif =

1.	5.	3.	7.	6.	6.	0.3333333	0.25
0.2	1.	0.3333333	5.	3.	3.	0.2	0.1428571
0.3333333	3.	2.	6.	3.	4.	0.	0.2
0.1428571	0.2	0.1666667	1.	0.3333333	0.25	0.1428571	0.125
0.1666667	0.3333333	0.3333333	3.	1.	0.5	0.2	0.1666667
0.1666667	0.3333333	0.25	4.	2.	1.	0.2	0.1666667
3.	5.	0.	7.	5.	5.	2.	0.5
4.	7.	5.	8.	6.	6.	2.	1.

Com as mesmas funções utilizadas inicialmente na matriz original, para a matriz modificada encontramos $\lambda_{\max}' = 8,811$ e temos um novo w .

wmodif =

0.1743770
0.0620057
0.1020813
0.0193032
0.0338274
0.0412316
0.2225585
0.3446154

É a partir desse novo w que calculamos os a_{37} e a_{73} definitivos utilizando

a relação $a_{ij}' = \frac{w_i'}{w_j'}$.

-->a37=wmodif(3,:)/wmodif(7,:)

a37 =

0.4586718

-->a73=wmodif(7,:)/wmodif(3,:)

a73 =

2.1802081

Saaty sugere que no método AHP os elementos da matriz sejam arredondados para 0.5 e 2. Define-se a matriz A_{final} .

-->Afinal=A;

-->Afinal(3,7)=1/2

Afinal =

1.	5.	3.	7.	6.	6.	0.3333333	0.25
0.2	1.	0.3333333	5.	3.	3.	0.2	0.1428571
0.3333333	3.	1.	6.	3.	4.	0.5	0.2
0.1428571	0.2	0.1666667	1.	0.3333333	0.25	0.1428571	0.125
0.1666667	0.3333333	0.3333333	3.	1.	0.5	0.2	0.1666667
0.1666667	0.3333333	0.25	4.	2.	1.	0.2	0.1666667
3.	5.	0.1666667	7.	5.	5.	1.	0.5
4.	7.	5.	8.	6.	6.	2.	1.

-->Afinal(7,3)=2

Afinal =

1.	5.	3.	7.	6.	6.	0.3333333	0.25
0.2	1.	0.3333333	5.	3.	3.	0.2	0.1428571
0.3333333	3.	1.	6.	3.	4.	0.5	0.2
0.1428571	0.2	0.1666667	1.	0.3333333	0.25	0.1428571	0.125
0.1666667	0.3333333	0.3333333	3.	1.	0.5	0.2	0.1666667
0.1666667	0.3333333	0.25	4.	2.	1.	0.2	0.1666667
3.	5.	2.	7.	5.	5.	1.	0.5
4.	7.	5.	8.	6.	6.	2.	1.

A seguir, se o tomador de decisão estiver de acordo com a modificação nos julgamentos, calculam-se o autovetor principal de A_{final} , o autovalor máximo e a inconsistência, com as mesmas funções já demonstradas.