

MCDM Techniques Using R
Prof. Gaurav Dixit
Department of Management Studies
Indian Institute of Technology - Roorkee

Lecture – 19
Fuzzy AHP – Part II

Welcome to the course MCDM techniques using R, so in previous lecture we actually completed our discussion on fuzzy AHP, so today we are going to do an exercise in R studio environment, in R environment before that let us have a recap of what we discussed in previous lecture, so we talked about fuzzy AHP, we talked about the purpose of traditional AHP.

(Refer Slide Time: 00:52)

Fuzzy AHP

- Purpose of traditional AHP is
 - To capture the expert's knowledge
- Criticism
 - Still cannot really reflect the human thinking style
 - Uses an exact value to express the decision maker's opinion in a comparison of alternatives
 - Use of unbalanced scale of judgments
 - Inability to adequately handle the inherent uncertainty and imprecision in the pairwise comparison process

And why the need for fuzzy AHP, so in a typically, you know researchers feel that the express knowledge has not been adequately captured all by traditional AHP and there are all certain criticism or drawbacks with traditional AHP, for example is not reflecting the human thinking style, it requires decision-makers to express exact values, when they are asked to compare you know criteria or alternatives, you know use of unbalanced scale of judgements, inability to adequately handle the inherent or uncertainty and imprecision in the pairwise comparison process.

So, there are certain limitations, certain drawbacks of our traditional AHP which can actually be overcome by fuzzy AHP, right, so we talked about these particular aspect also, so we talked

about various approaches within fuzzy AHP which are available which have been implemented, then we talked about the particular and the most popular approach that is available is based on extent analysis.

(Refer Slide Time: 02:07)

Fuzzy AHP

- Fuzzy AHP
 - More accurate to give interval judgments than fixed value judgments
 - Typical difficulty to express preference explicitly due to the fuzzy nature of the comparison process
 - Various approaches
 - Fuzzy ratios using triangular fuzzy numbers
 - Trapezoidal fuzzy numbers
 - Extent analysis method using triangular fuzzy numbers
 - Extent fuzzy AHP

So, this extent analysis method using triangular fuzzy number, so triangular fuzzy number is something that we have already discussed in our lecture on fuzzy sets, so you know this part was something that we discussed in the previous lecture, we looked at different steps of extent fuzzy AHP.

(Refer Slide Time: 02:29)

Fuzzy AHP

- Extent fuzzy AHP
 - Linguistic variables are used to express the comparative judgements
 - Weights of criteria and ratings of alternatives
 - TFNs are used to specify linguistic values of these variables
 - An object set: $X = \{x_1, \dots, x_j, \dots, x_n\}$
 - Similar to n alternatives in traditional AHP
 - A goal set: $G = \{g_1, \dots, g_j, \dots, g_m\}$
 - Similar to m criteria in traditional AHP

So, we talked about that linguistic variables are used to express the comparative judgements, which is different from what we doing in classical AHP or traditional AHP, right, so this scale itself is actually used to you know, used for the pairwise comparison process is different, this is a linguistic variables. So, weights of criteria and ratings of alternative are actually they are collected using this particular scale.

And then the values, these linguistic values, these linguistic terms which have been collected, they are then specified using T offence, the Fuzzy numbers, right, so once that is done when is the data collection part is done, we have taken our responses from decision-makers for criteria as well as alternatives, then you know we talked about you know the particular mathematical aspect of fuzzy AHP.

So, the steps are similar to you know traditional AHP however, the whole environment being for the different you know being the fuzzy environment and the scale that is being use is different, so the underlying mathematics and so those kind of differences are you know quite there, so in terms of you know mathematics we talked about object set and the goal set, so we talked about different you know steps that can be taken.

(Refer Slide Time: 04:00)

Fuzzy AHP

- Extent fuzzy AHP
 - Extent analysis is performed on each object x_i w.r.t each goal g_j
 - For object x_i , extent analysis value w.r.t goal g_j is expressed using following notation:

$$M_i^j$$

Where $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, m$

All the M_i^j are triangular fuzzy numbers

- Subsequent steps use these values
-

So, we talked about extent analysis values that would be available to us which are you know, so these are; these values are nothing but triangular fuzzy numbers based on you know the

responses using the linguistic scale that we have you know obtained from the decision-makers, so that is these numbers will have and the subsequent steps, so as we talked about you know use these values, do certain apply certain mathematics.

And then take it forward into subsequent steps, so we talked about you know fuzzy synthetic extent and the actual expression formula that is used to calculate these values as you can see in the slide as well.

(Refer Slide Time: 04:44)

Fuzzy AHP

- Step 2: Compute the degree of possibility of

$$S_2(l_2, m_2, u_2) \geq S_1(l_1, m_1, u_1)$$

Where degree of possibility between two fuzzy synthetic extents is defined as

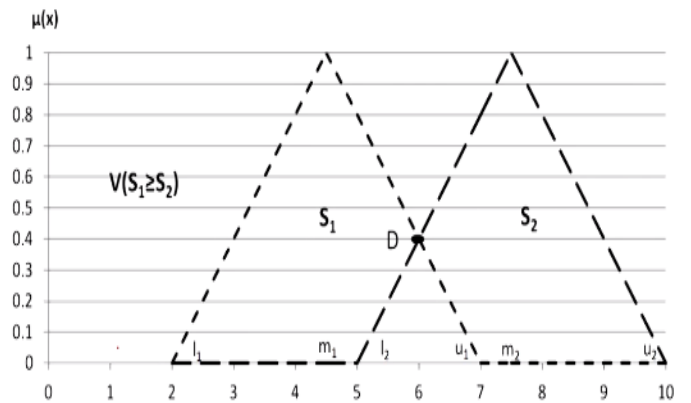
$$V(S_2 \geq S_1) = \sup_{y \geq x} [\min(\mu_{S_2}(y), \mu_{S_1}(x))]$$

or

Then we talked about you know degree of possibility you know to fuzzy extent values or you know when it can be greater than the other, so how to determine this aspect that is also something that we talked about, right, so membership function is also you know something that we learned in the fuzzy sets you know that is used here and how it is defined here, so those aspects also you know something that we talked about in the previous lecture.

(Refer Slide Time: 05:03)

Fuzzy AHP



We also understood you know these steps you know through this graphic as well.

(Refer Slide Time: 05:11)

Fuzzy AHP

- Step 3: Compute the degree of possibility for a convex fuzzy number to be greater than k convex fuzzy numbers $S_i (1,2,\dots,k)$

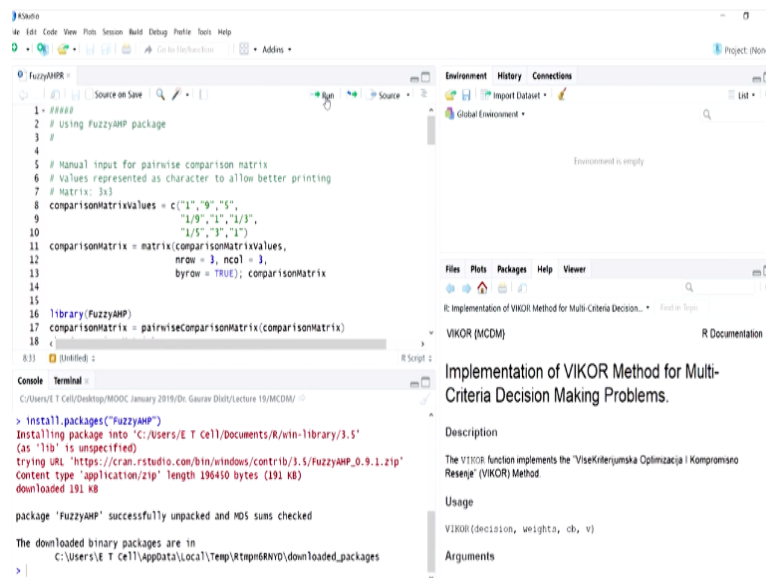
$$\begin{aligned}
 V(S \geq S_1, S_2, \dots, S_k) \\
 &= V[(S \geq S_1) \text{ and } (S \geq S_2) \text{ and } \dots \text{ and } (S \geq S_k)] \\
 &= \min V(S \geq S_i), \quad i = 1, 2, \dots, k
 \end{aligned}$$

And you know step 3 was about you know the degree of possibility for a convex fuzzy number, so again to understand the mathematics, so when it is going to be greater than K fuzzy numbers, so the mathematics involved and then finally the weight factor, so how that is determined, so the all the steps that we have talked about these were mainly about the you know extent value method that is you know, the extent value method that is being the most popular method and being used using a triangular fuzzy numbers.

How are there are you know from the since that time this method was proposed and develop and it has been use in number of you know research studies and in the practitioner community as well and since then I know there have been other proposals have been made by the researchers, so those are also available but this is being the most popular that is why we discussed the steps of; this step.

So, what we are going to do is; we will do an exercise in R, where we will find out which methods have actually implemented, so there have been a certain recent developments also, so those have already been incorporated, so what we discussed in the previous lecture, the extent value method that is that development was by China in 1996 now, certain you know recent developments have happened and which have been you know implemented in R environment as well.

(Refer Slide Time: 06:44)



```
1 #####
2 # Using FuzzyAMP package
3 #
4
5 # Manual input for pairwise comparison matrix
6 # Values represented as character to allow better printing
7 # Matrix: 3x3
8 comparisonMatrixValues = c("1", "9", "5",
9 "1/9", "1", "1/3",
10 "1/5", "1", "1")
11 comparisonMatrix = matrix(comparisonMatrixValues,
12 nrow = 3, ncol = 3,
13 byrow = TRUE); comparisonMatrix
14
15
16 library(FuzzyAMP)
17 comparisonMatrix = pairwiseComparisonMatrix(comparisonMatrix)
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

```
C:\Users\T Cell\Desktop\MOOC January 2019\Dr. Gaurav Dixit\Lecture 19\MCDM > install.packages("FuzzyAMP")
Installing package into 'C:/Users/E T Cell/Documents/R/win-library/3.5'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.5/FuzzyAMP_0.9.1.zip'
Content type: 'application/zip' length 196450 bytes (191 KB)
downloaded 191 KB
package 'FuzzyAMP' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
C:\Users\E T Cell\AppData\Local\Temp\Wtgn6KWD\downloaded_packages
```

```
R: Implementation of VIKOR Method for Multi-Criteria Decision...
VIKOR (MCDM)
R Documentation
Implementation of VIKOR Method for Multi-Criteria Decision Making Problems.
Description
The VIKOR function implements the 'ViseKriterijumska Optimizacija I Kompromiso Resenje' (VIKOR) Method.
Usage
VIKOR(decision, weights, cb, v)
```

So, those methods are available with us, you know in the R environment and we can always compare the results of these methods and these studies are also available which compare the performance of different methods as well, so however you know, we will not go into you know too much you know detail into the theoretical aspect in terms of comparison and all that. Now, we will focus on the R studio environment, the R environment.

And how fuzzy AHP modelling can actually be done with the help of available functions and packages that are there in R environment, so the pack is that we are going to use for this particular exercise is actually the fuzzy AHP package, so this is available for fuzzy AHP implementation are and we are going to take this particular example is more like a manufacture example, so others have been also, the other example that we have used in this particular course have been manufactured.

But here we are not talking about, we not even talking you know about a particular you know decision problems and goals and criteria rather will just look at the numbers and go through the functions and the results that are being produced, so the coding and modelling parts we will focus more on there and once you learn this what we are doing in this you know R environment are depending on your case depending on the example that is in problem that you are tacking.

You can always apply that so, we are going to use fuzzy AHP package so, first we need to check whether this package install or not, so what we will do is; you know we will first check the installed dot package function, we will see whether this package installed, so let us type installed dot packages and as we have been doing in previous lectures as well, so just type the package name that is fuzzy AHP.

So, it is case sensitive, so make sure the alphabets are typed in the correct cases and these installation has already started, so this is install now so now, we can go ahead and do our exercise, so what we are doing is here also using this package, there are 2 things that can be done in terms for loading the data in terms of loading the input that we require for fuzzy AHP modelling.

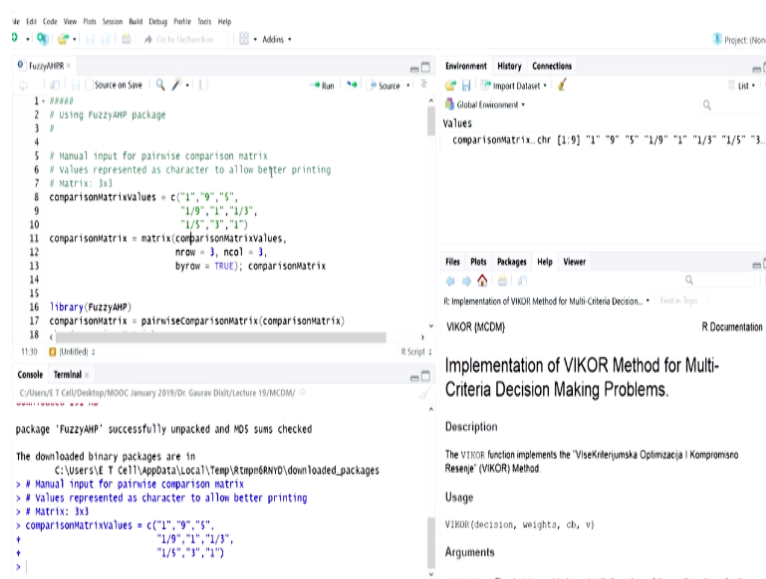
So, manual input can be provided well like in other techniques, we have been doing, we used to give manual you know input, we use to create variables or vectors in the R environment itself, or you can always load these values from some Excel files and import them into R environment, so that is something that can also be done, so you can see here you know manual input for pairwise comparison matrix.

So, values represented as character to allow better printings, so we can see you know this first line of code the values are in double quotes, so when we use double quotes, so automatically it would be you know taken as a character you know value in R environment, so that it is mainly to be able to you know later on use the print function, so that the output that we are going to print you know that would be more comprehensible.

So, for that benefit we are you know inputting the values in the character format otherwise in the numeric format also we can always, you know use the same functions, same code, so again we are using the combined function; C function to you know combine all these you know input values, so these values are for you know criteria so, we have three criteria, we are considering 3 criteria; 3 by 3 matrix.

So, you know we need to pass on 9 values here, as we can see so, comparison matrix value, so we have 1, then 9, 5 then 1 divided by 9 then 1, 1 divided by 3, then 1 / 5, then 3 and 1, so these are comparison; pairwise comparison of you know criteria so, let us run this code.

(Refer Slide Time: 11:17)



The screenshot shows the RStudio interface. The script editor on the left contains the following code:

```
1 #####
2 # Using FuzzyMOP package
3 #
4
5 # Manual input for pairwise comparison matrix
6 # Values represented as character to allow better printing
7 # Matrix: 3x3
8 comparisonMatrixValues = c("1", "9", "5",
9 "1/9", "1", "1/3",
10 "1/5", "3", "1")
11 comparisonMatrix = matrix(comparisonMatrixValues,
12 nrow = 3, ncol = 3,
13 byrow = TRUE); comparisonMatrix
14
15
16 library(FuzzyMOP)
17 comparisonMatrix = pairwiseComparisonMatrix(comparisonMatrix)
18
```

The console on the left shows the output of the script:

```
package 'FuzzyMOP' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
  c:\Users\IE T Cell\AppData\Local\Temp\9t8np68NVD\downloaded_packages
> # Manual input for pairwise comparison matrix
> # Values represented as character to allow better printing
> # Matrix: 3x3
> comparisonMatrixValues = c("1", "9", "5",
+ "1/9", "1", "1/3",
+ "1/5", "3", "1")
> |
```

The Environment pane on the right shows the variable `comparisonMatrix_chr` with the value:

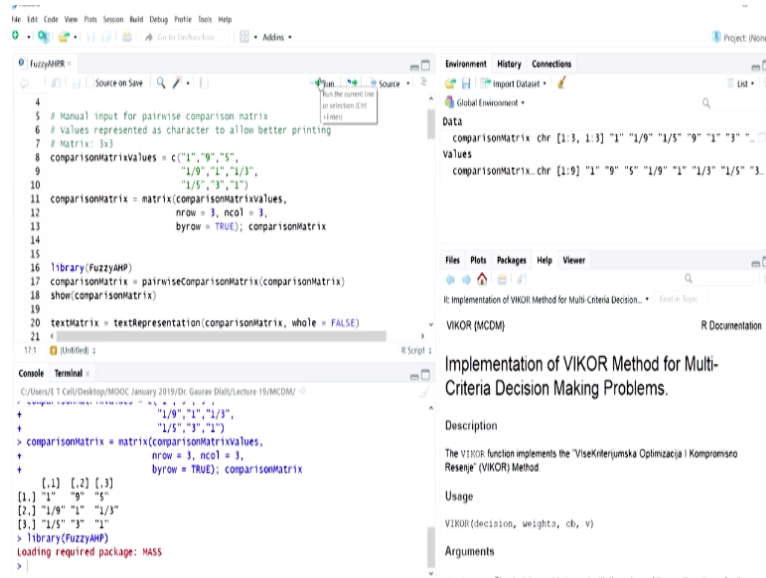
```
comparisonMatrix_chr [1:9] "1" "9" "5" "1/9" "1" "1/3" "1/5" "3"
```

The R Documentation pane on the right shows the documentation for the `VIKOR (MCDM)` function, titled "Implementation of VIKOR Method for Multi-Criteria Decision Making Problems."

So, you can see in the environment section, you can see a; this is; this value has been you know defined comparison matrix values. Now, we can create a comparison matrix using these comparison you know, the values that we have just you know created, so because the matrix is 3

by 3, so we can see y row, so you are already familiar with this function; matrix function and how we can create a matrix, so let us run this code.

(Refer Slide Time: 11:48)



```
4 # Manual input for pairwise comparison matrix
5 # Values represented as character to allow better printing
6 # Matrix: 3x3
7 comparisonMatrixValues = c("1", "9", "5",
8 "1/9", "1", "1/3",
9 "1/5", "3", "1")
10 comparisonMatrix = matrix(comparisonMatrixValues,
11 nrow = 3, ncol = 3,
12 byrow = TRUE); comparisonMatrix
13
14
15
16 library(FuzzyAHP)
17 comparisonMatrix = pairwiseComparisonMatrix(comparisonMatrix)
18 show(comparisonMatrix)
19
20 textMatrix = textRepresentation(comparisonMatrix, whole = FALSE)
21
```

Environment History Connections

Data

```
comparisonMatrix chr [1:3, 1:3] "1" "1/9" "1/5" "9" "1" "3" "1"
Values
comparisonMatrix_chr [1:9] "1" "9" "5" "1/9" "1" "1/3" "1/5" "3"
```

Files Plots Packages Help Viewer

R: Implementation of VIKOR Method for Multi-Criteria Decision...

VIKOR (MCDM) R Documentation

Implementation of VIKOR Method for Multi-Criteria Decision Making Problems.

Description

The VIKOR function implements the "ViseKriterijumska Optimizacija i Kompromisno Resenje" (VIKOR) Method

Usage

```
VIKOR(decision, weights, cb, v)
```

Arguments

Console Terminal

```
C:/Users/T Cell/Desktop/MOOC January 2019/Dr. Gaurav Dhill/Lecture 15/MCDM/
> comparisonMatrix = matrix(comparisonMatrixValues,
+ nrow = 3, ncol = 3,
+ byrow = TRUE); comparisonMatrix
+
+      [,1] [,2] [,3]
+ [1,] "1"  "9"  "5"
+ [2,] "1/9" "1"  "1/3"
+ [3,] "1/5" "3"  "1"
+
+ Loading required package: MASS
+ 
```

So, you can see the matrix here, 1, 2, 3 you know rows, 3 columns, so because we have 3 criteria and this is comparison you know between criteria, so you can see here. Now, the library fuzzy AHP which we have already installed, so let us load this library into R environment, so let us run this, so you can see this has been loaded. Now, what we are going to do is; we will use the pairwise comparison matrix function to convert this comparison matrix into you know into a more you know more comprehensible format.

So, you know let us run this code and what this function actually does is; we can immediately so it using the so function.

(Refer Slide Time: 12:35)

```

7 # Matrix: 3x3
8 comparisonMatrixValues = c("1", "9", "5",
9                             "1/9", "1", "1/3",
10                            "1/5", "3", "1")
11 comparisonMatrix = matrix(comparisonMatrixValues,
12                             nrow = 3, ncol = 3,
13                             byrow = TRUE); comparisonMatrix
14
15
16 library(fuzzyAHP)
17 comparisonMatrix = pairwiseComparisonMatrix(comparisonMatrix)
18 show(comparisonMatrix)
19
20 textMatrix = textRepresentation(comparisonMatrix, whole = FALSE)
21 print(textMatrix)
22
23 print(comparisonMatrix)
24

```

Environment pane shows: comparisonMatrix (Formal class PairwiseComparisonMatrix)

Values: comparisonMatrix_chr [1:9] "1" "9" "5" "1/9" "1" "1/3" "1/5" "3"

Console Terminal:

```

[3,] "1/5" "3" "1"

slot "values":
      [,1] [,2] [,3]
[1,] 1.0000000 9 5.0000000
[2,] 0.1111111 1 0.3333333
[3,] 0.2000000 3 1.0000000

slot "variableNames":
[1] "C1" "C2" "C3"

```

So, this is how we get this so, you would see that you know row names and column names, those variable names have been appropriately assigned as you can see here, C1, C2 and C3 and values you can see in the numeric format and the values in the character format because we had entered in you know the values in character format, so this function really is going to help us in terms of you know in terms of having the access using different format.

Now another way to represent the same matrix is the; this function, text representation so again, it takes a comparison matrix itself as the first argument and then there is another argument whole which actually is you know assigned false here, so it actually is telling that we just need the you know, upper triangular values, so we just run this.

(Refer Slide Time: 13:33)

```

15
16 library(FuzzyMCDM)
17 comparisonMatrix = pairwiseComparisonMatrix(comparisonMatrix)
18 show(comparisonMatrix)
19
20 textMatrix = textRepresentation(comparisonMatrix, whole = FALSE)
21 print(textMatrix)
22
23 print(comparisonMatrix)
24
25 # Three consistency checks
26 CR = consistencyRatio(comparisonMatrix); CR
27 weakConsistency = weakConsistency(comparisonMatrix); weakConsistency
28 strictConsistency = strictConsistency(comparisonMatrix); strictConsistency
29
30 # Calculate weights using geometric mean
31 weights = calculateWeights(comparisonMatrix)
32
...
281

```

Console Terminal:

```

C:\Users\T Cell\Desktop\MOOC January 2019\Dr. Gaurav Dhillon\lecture 15\MCDM >
C_2 1/9 1 1/3
C_3 1/5 3 1
> # Three consistency checks
> CR = consistencyRatio(comparisonMatrix); CR
Consistency ratio is: 0.0279459296138796. The pairwise comparison matrix is consistent for calculations.
[1] 0.02794593
> weakConsistency = weakConsistency(comparisonMatrix); weakConsistency
The comparison matrix is weakly consistent.
[1] TRUE
>

```

Environment:

```

comparisonMatrix Formal class 'PairwiseComparisonMatrix'
textMatrix 3 obs. of 3 variables
Values
comparisonMatrix chr [1:9] "1" "9" "5" "1/9" "1" "1/3" "1/5" "3"
CR 0.0279459296138796
weakConsistency TRUE

```

R Documentation: Implementation of VIKOR Method for Multi-Criteria Decision Making Problems.

And you know run print, so you can see you know much better representation of this matrix; C1 C2 C3 in the columns side and C1, C2, C3 in the row side so those have been named. Earlier the matrix we had the you know the default notation that 1, 2, 3 and all that so, we can see here, so the upper triangular not you know, upper triangular you know matrix values have been you can see here and the diagonal values are also there.

However, as we have talked about in the traditional has been that you know number of comparison; necessary comparisons that we might require or just $N^2 - N$ divided by 2, so here in this case, we have you know 3 criteria, so therefore that value would be $3^2 - 3$ divided by 2 that comes out to be you know 3, so actually we can see that just 3 values are required here, so you can see 9, 5, and 1/3 and other than the diagonal values, which is comparison of criteria with itself.

So, this is how we can actually create the comparison matrix for criteria and presented in the R environment displayed it, display the same in the R environment, now there is another way to display this, we can use the print f function directly and pass on the comparison matrix as the argument; first argument, you can see now, but this will have the affect of displaying the matrix as full.

So, instead we could have used the text representation of function as well where in the second argument whole falls instead of using falls, we could have used it as true and we would have got the same output, so this is how we can actually you know we can actually were you know provide manual input for you know criteria values, right so and then we can also take care of the display, how it is going to be displayed.

Now, under this package, you know certain consistency checks have been implemented, so those can also be you know used here, so first one is consistency ratio something that we talked about in the traditional AHP lecturer as well, so you know here we can use this function consistency, ratio function and pass on the matrix that we have just created comparison matrix and we will get the values.

So, you can see, now in this package in this function we are also getting a descriptive you know result here that consistency ratio is this much and the pairwise comparison matrix is consistent for calculation, so as per the you know consistency ratio method that we talked about you know in the AHP lecture, so the value that we are getting is that is giving us that it is consistent, then there are certain other consistency checks that are available.

So, one is called weak consistency, then other is called strict consistency, if you want to get more details about these consistency check, you can always go into the help section and type the name of the function, weak consistency and strict consistency and you will get the details like who propose this method, when this method was proposed and what is the underlying you know mathematics or idea behind this you know these consistency checks, you can always follow the references and the details in the help section that is already available to you.

So right now, we will just execute this line up code, we will use this weak consistency function as well so we you know use we will again getting the you know, descriptive result here that the comparison matrix is weakly consistent right, so this is available now, if we want to have an more stricter and more tougher consistency check then, we have this is strict consistency function available with us.

(Refer Slide Time: 17:43)

```

20 textMatrix = textRepresentation(comparisonMatrix, whole = FALSE)
21 print(textMatrix)
22
23 print(comparisonMatrix)
24
25 # Three consistency checks
26 CR = consistencyRatio(comparisonMatrix); CR
27 weakConsistency = weakConsistency(comparisonMatrix); weakConsistency
28 strictConsistency = strictConsistency(comparisonMatrix); strictConsistency
29
30 # Calculate weights using geometric mean
31 weights = calculateWeights(comparisonMatrix)
32 print(weights)
33
34 # Data on alternatives for AHP calculation
35 # Using linguistic scale (higher value is better)
36 # No. of columns must be equal to no. of criteria in the data matrix
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

```

[1] TRUE
> strictConsistency = strictConsistency(comparisonMatrix); strictConsistency
Comparison matrix isn't strictly consistent. These indices violate the condition:
[1,] | = ([1,2]*[2,3]) -- 5 | = 3
[1,2] | = ([1,3]*[3,2]) -- 9 | = 15
[2,1] | = ([2,3]*[3,1]) -- 0.333333333333333 | = 0.555555555555556
[2,1] | = ([2,3]*[3,1]) -- 0.111111111111111 | = 0.066666666666667
[3,2] | = ([3,1]*[1,2]) -- 3 | = 1.8
[3,1] | = ([3,2]*[2,1]) -- 0.2 | = 0.333333333333333

```

So, we can just run this: see the results here you can see the descriptive result is also available in the first line of the result itself, the comparison matrix is not a strictly consistent and these indices are given value violate the conditions, so that also something that you can see, right because as per this you know function, the values would be you know very; values would be exactly same, so therefore it is; it provides say you know very strong you know consistency check.

So, these are different consistency checks that are available with us that can be used as per our requirement. Now you know, next thing is we need to calculate weight, so once we have got the input values, the pairwise comparison values for the criteria; we can always go ahead in the next system and compute the weights, so we have this function calculate weights in this package which can be used to compute these weights.

So, first argument is the comparison matrix which we have already created, so we will just run this code, so this particular method is you know by default this is using geometric mean.

(Refer Slide Time: 18:59)

```

30 # calculate weights using geometric mean
31 weights = calculateweights(comparisonMatrix)
32 print(weights)
33
34 # Data on alternatives for AHP calculation
35 # using linguistic scale (higher value is better)
36 # no. of columns must be equal to no. of criteria in the data matrix
37 values = c(8,5,3,
38            1,3,9,
39            8,6,4,
40            3,2,7,
41            6,7,5,
42            4,5,3,
43            NA,9,9,
44            NA,NA,NA)
45
46 values = matrix(values, nrow = length(values)/length(weights/weights),
47                 byrow = TRUE)
48
49 [1] FALSE
50 > # Calculate weights using geometric mean
51 weights = calculateweights(comparisonMatrix)
52 > print(weights)
53      w_c_1 w_c_2 w_c_3
54 0.7514 0.0704 0.1782
55 >

```

So, we run this so, the geometric mean that something that we talked about in the AHP lecture, so that is the method that will be used to generate these weights and we can always print these values using different functions, we can see, weight for criteria one comes out to be .75, weight for criteria 2 comes out to be .07 and weight for criteria 3 comes out to be .17, second one was .07, third one .17.

So, we can see criteria 1 is you know having the highest weightage and quite high in comparison to 2 other criteria, so this is how, so up to now we have computed the criteria weights, now this particular package has its own requirement for AHP modelling, so we will talk about you know those things as far. For example, data on alternatives for AHP calculation, so the weight is required is slightly different.

So again, using linguistic scale, you know higher value is better so, linguistic scale is something that we have already talked about in you know in a previous lecture, so how were while we are using that scale of you know getting the responses from the design makers, in the R environment and particularly this package; fuzzy AHP package, the values that we require on alternatives, this would be in this format, number of columns, the matrix that is going to be used is should have this thing, this characteristic.

That number of columns must be = number of criteria in the data matrix, so number of criteria that we are using is 3, we have already computed credit weights as well, we have 3 by 3 pairwise comparison matrix for criteria, so that part we have computed, so because we are using 3 criteria, so therefore number of columns in this data matrix for alternatives is going to be 3 and the rows are going to be associated with the different alternatives that we have.

So, it is going to be like a decision matrix, so this data matrix is actually like a decision matrix like in other techniques you know in traditional AHP, we used to have you know comparison matrix were alternative were compared with each other, you know with respect to a criteria, so the those matrices for alternatives were also between alternatives and the values were you know comparison between alternative with respect to a given you know criteria.

However here, I know these second; this argument that we are willing of this wherever we are creating for alternative, it is more like a decision matrix, so that we have been using in other techniques, so it is like you know performance is poor, this kind of performance score I know of a you know alternative with respect to criteria, so that kind of input is required, later on we will be using this matrix for AHP modelling.

So, you can see in this values function, we have you know these manage values, we have 3 criteria, so 3 columns and 8 alternatives, so 24 values we have and you can see few values have been specified as NA, so they can be handle, they are going to handle accordingly also that is already you know covered in the available function that we have.

(Refer Slide Time: 22:23)

```

41 6,7,5,
42 4,5,3,
43 NA,9,9,
44 NA,NA,NA)
45 values = matrix(values, nrow = length(values)/length(weights/weights),
46 ncol = length(weights/weights), byrow = TRUE)
47
48 # AHP modeling
49 result = calculateAHP(weights, values)
50 print(result)
51
52 rank = compareResults(result)
53 print(rank)
54
55 fullresult = cbind(values, result, rank)
56 colnames(fullresult) = c("crit1", "crit2", "crit3", "result_value", "ranking")
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

```

> # No. of columns must be equal to no. of criteria in the data matrix
> values = c(4.5,3,
+ 8.6,4,
+ 3.2,7,
+ 6.7,5,
+ 4.5,3,
+ NA,9,9,
+ NA,NA,NA)
> values = matrix(values, nrow = length(values)/length(weights/weights),
+ ncol = length(weights/weights), byrow = TRUE)
> |

```

```

1.3
+ 8.6
+ 3.2
+ 6.7
+ 4.5
+ NA
+ 9.9
+ NA

```

So, let us run this code, we will generate values you know vector and now, we will you know generate the matrix for this, so you can see we are going to generate you know 8 by 3 matrix here because number of row is length of values divided by length of you know weights you know vector that we have already computed that is so this number of rows is going to be 8 and the number of columns going to be 3.

So, therefore, this will have a you know 8 by 3 matrix, now this is this decision matrix is going to be passed on to the function calculate AHP, so this is the function calculate AHP, which is actually used to a you know do AHP modelling here so, if you want; if you are interested in more detail about this book of function, you can always go in to the help section and just type, calculate AHP and you will get this, you can see.

So, you can see calculate AHP, it has two arguments you know weights and data, so weights is criteria; weights that we have already computed and the values which is referring to decision matrix that we have already, you know created, so both the arguments have already been generated, now we just need to do our AHP modelling, so calculate AHP is the function in this package, so let us run this code.

(Refer Slide Time: 23:52)


```

41 6,7,5,
42 4,5,3,
43 NA,9,9,
44 NA,NA,NA,
45
46 values = matrix(values, nrow = length(values)/length(weights/weights),
47                 ncol = length(weights/weights), byrow = TRUE)
48
49 # AHP modeling
50 result = calculateAHP(weights, values)
51 print(result)
52
53 rank = compareResults(result)
54 print(rank)
55
56 fullresult = cbind(values, result, rank)
57 colnames(fullresult) = c("crit1", "crit2", "crit3", "result_value", "ranking")
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

```

> print(result)
      result
[1,] 3.892240
[2,] 2.566257
[3,] 7.140454
[4,] 3.642293
[5,] 5.892240
[6,] 3.892240
[7,] NA
[8,] NA
> rank = compareResults(result)
> print(rank)
      rank
[1,] 3
[2,] 6
[3,] 1
[4,] 5
[5,] 2
[6,] 3
[7,] NA
[8,] NA

```

We will have the results stored in this vector which are vector and we can print this using print function, so we can see for each of these you know alternatives 1 to 8, we have got a score, right, just like in the traditional AHP that we used to so, we have already got this. If you want to rank this, we can always use the compared results function that is available in this package, so you know we will get the rank let us print this, you can see the rank.

(Refer Slide Time: 24:18)

```

43 NA,9,9,
44 NA,NA,NA,
45
46 values = matrix(values, nrow = length(values)/length(weights/weights),
47                 ncol = length(weights/weights), byrow = TRUE)
48
49 # AHP modeling
50 result = calculateAHP(weights, values)
51 print(result)
52
53 rank = compareResults(result)
54 print(rank)
55
56 fullresult = cbind(values, result, rank)
57 colnames(fullresult) = c("crit1", "crit2", "crit3", "result_value", "ranking")
58 print(fullresult)
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

```

> rank = compareResults(result)
> print(rank)
      rank
[1,] 3
[2,] 6
[3,] 1
[4,] 5
[5,] 2
[6,] 3
[7,] NA
[8,] NA

```

So, you can see the rank, so you know view, you know see these values, so alternative one is rank 3, you can see here that you know alternative 3 is having the highest score 7.14, so view go back and see the results for the see the results here, alternative 3 is rank 1, so you can see here till now what we are doing is we are implementing you know we are doing a traditional AHP

modelling, so later on because we want to familiarise to we you know familiarise our self with this package and the same package, same function are going to be use, similar functions are going to be used for fuzzy AHP modelling as we will see later in this lecture.

So, full result can also be you know computed in this fashion where first argument you know values, result and rank all these 3 you know can be you know can be joined using the function C bind, so this is actually for standing for column bind, so all these you know variables are going to be you know joined.

(Refer Slide Time: 25:31)

```

57 colnames(fullresult) = c("crit1", "crit2", "crit3", "result_value", "ranking")
58 print(fullresult)
59
60 #####
61 # Fuzzy AHP
62 # Fuzzification of pairwise comparison matrix
63 # Default fuzzy scale: 1-9
64 # Default width of fuzzy number = 2 (FW)
65 fuzzyComparisonMatrix = fuzzyPairwiseComparisonMatrix(comparisonMatrix)
66 print(fuzzyComparisonMatrix)
67
68 # Fuzzy AHP modeling
69 fuzzyresult = calculateAHP(fuzzyComparisonMatrix, values); fuzzyresult
70
71 # Extract 2nd index from 'fuzzyresult'
72 fuzzyNumber = getFuzzyNumber(fuzzyresult, as.integer(2))
73 print(fuzzyNumber)
74

```

Environment

comparisonMatrix	Formal class PairwiseComparisonMatrix
fullresult	num [1:8, 1:5] 4 1 8 3 6 4 NA NA 5 3 ...
rank	num [1:8, 1] 3 6 1 5 2 3 NA NA
result	num [1:8, 1] 3.89 2.57 7.15 3.64 5.89 ...
textMatrix	3 obs. of 3 variables
values	num [1:8, 1:3] 4 1 8 3 6 4 NA NA 5 3 ...
weights	Formal class weights
values	

```

> colnames(fullresult) = c("crit1", "crit2", "crit3", "result_value", "ranking")
> print(fullresult)
  crit1 crit2 crit3 result_value ranking
[1,] 4 5 3 3.892240 3
[2,] 1 3 9 2.56257 6
[3,] 8 6 4 7.146454 1
[4,] 3 2 7 3.642793 5
[5,] 6 7 5 5.892240 2
[6,] 4 5 3 3.892240 3
[7,] NA 9 9 NA NA
[8,] NA NA NA NA NA

```

Function to calculate result of AHP

Description

This function calculates output of AHP based on **Weights** or **FuzzyWeights** on data represented either by matrix or **FuzzyData**.

Usage

```
calculateAHP(weights, data)
```

S4 method for signature "Weights,matrix"
calculateAHP(weights, data)

S4 method for signature "FuzzyWeights,matrix"

And the new, you know new you know matrices going to be created here that is full result, so you can assign here column names and you can print the full results, you can see criteria 1, criteria 2 and criteria 3, so you can see this is like decision you know decision matrix numbers that we had, so you can see here decision matrix in first 3 columns then we you have the result value that is the course that has been computed after you know traditional AHP modelling and then the ranking also.

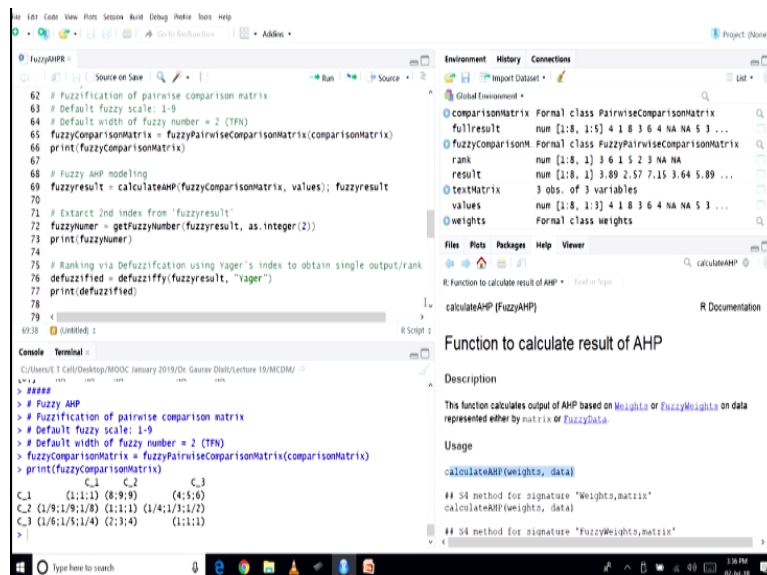
So, this is the complete results, so what we have just done using this fuzzy AHP package, we have actually done our classic you know AHP modelling, the traditional AHP modelling, now let us move towards fuzzy AHP modelling, so you know what we need to do in fuzzy AHP is first

thing we need to do a fuzzification pairwise comparison matrix, so pairwise comparison matrix for criteria we already have.

So, we need to do a fuzzification or default fuzzy scale 1 to 9, default width of fuzzy number is 2 because we are implementing you know triangular fuzzy numbers, so there we have you know 3 numbers you know there, so small you know possible value the you know most you know most promising value and the you know largest possible value, so those 3 value we have in triangular fuzzy numbers, so you know that has been implemented in this you know package.

So, we need to use this fuzzy pairwise comparison matrix to convert the comparison matrix that we have already computed into a fuzzy comparison matrix, so it would be like a fuzzification of the you know traditional comparison matrix, so let us run this code.

(Refer Slide Time: 27:26)



```
62 # Fuzzification of pairwise comparison matrix
63 # Default fuzzy scale: 1-9
64 # Default width of fuzzy number = 2 (TFN)
65 fuzzyComparisonMatrix = fuzzyPairwiseComparisonMatrix(comparisonMatrix)
66 print(fuzzyComparisonMatrix)
67
68 # Fuzzy AHP modeling
69 fuzzyresult = calculateAHP(fuzzyComparisonMatrix, values); fuzzyresult
70
71 # Extract 2nd index from 'fuzzyresult'
72 fuzzyNumber = getFuzzyNumber(fuzzyresult, as.integer(2))
73 print(fuzzyNumber)
74
75 # Ranking via Defuzzification using Yager's index to obtain single output/rank
76 defuzzified = defuzzify(fuzzyresult, "Yager")
77 print(defuzzified)
78
79
```

Console Terminal

```
> ## ##
> # Fuzzy AHP
> # Fuzzification of pairwise comparison matrix
> # Default fuzzy scale: 1-9
> # Default width of fuzzy number = 2 (TFN)
> fuzzyComparisonMatrix = fuzzyPairwiseComparisonMatrix(comparisonMatrix)
> print(fuzzyComparisonMatrix)
      C_1  C_2  C_3
C_1 (1;1;1) (8;9;9) (4;5;6)
C_2 (1/9;1/9;1/8) (1;1;1) (1/4;1/3;1/2)
C_3 (1/6;1/5;1/4) (2;3;4) (1;1;1)
```

Environment History Connections

- Global Environment
- comparisonMatrix: Formal class PairwiseComparisonMatrix
- fullresult: num [1:8, 1:5] 4 1 8 3 6 4 NA NA 5 3 ...
- fuzzyComparisonMatrix: Formal class FuzzyPairwiseComparisonMatrix
- rank: num [1:8, 1] 3 6 1 5 2 3 NA NA
- result: num [1:8, 1] 3.89 2.57 7.15 3.64 5.89 ...
- textMatrix: 3 obs. of 3 variables
- values: num [1:8, 1:3] 4 1 8 3 6 4 NA NA 5 3 ...
- weights: Formal class weights

Function to calculate result of AHP

Description

This function calculates output of AHP based on **Weights** or **FuzzyWeights** on data represented either by matrix or **FuzzyData**.

Usage

```
calculateAHP(weights, data)
calculateAHP(fuzzyWeights, matrix)
```

S4 method for signature "Weights,matrix"
calculateAHP(weights, data)
S4 method for signature "FuzzyWeights,matrix"

And let us print the results, so you can see 3 by 3, you know matrix here in the results and you can see the fuzzy numbers being displayed here, so C1 to C1, this is the you know comparison with itself, so this is 1 1 1, if we look at the comparison of C1 to C2, you can see here 8 9 9, so the middle value is 9, right and the comparison of C1 with C3 4 5 6, middle value is 5, so whatever the matrix that we had in the comparison whatever values we had in the comparison matrix, now these have been fuzzified as using the pairwise comparison matrix.

Now, we have the fussy comparison matrix and we already have the decision matrix with us, so we can do fuzzy AHP modelling here, so what we need to do is; we just call this calculated AHP function again and now, you see the difference that first argument is fuzzy comparison matrix, the second argument is values, the decision matrix, so what we will get is the fuzzy result.

(Refer Slide Time: 28:29)

```

62 # Fuzzification of pairwise comparison matrix
63 # Default fuzzy scale: 1-9
64 # Default width of fuzzy number = 2 (TPN)
65 fuzzyComparisonMatrix = fuzzyPairwiseComparisonMatrix(comparisonMatrix)
66 print(fuzzyComparisonMatrix)
67
68 # Fuzzy AHP modeling
69 fuzzyresult = calculateAHP(fuzzyComparisonMatrix, values); fuzzyresult
70
71 # Extract 2nd index from 'fuzzyresult'
72 fuzzyNumber = getFuzzyNumber(fuzzyresult, as.integer(2))
73 print(fuzzyNumber)
74
75 # Ranking via Defuzzification using Yager's index to obtain single output/rank
76 defuzzified = defuzzify(fuzzyresult, "Yager")
77 print(defuzzified)
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

Environment History Connections

- Global Environment
- Import Dataset
- comparisonMatrix: Formal class PairwiseComparisonMatrix
- fullresult: num [1:8, 1:5] 4 1 8 3 6 4 NA NA 5 3 ...
- fuzzyComparisonMatrix: Formal class FuzzyPairwiseComparisonMatrix
- fuzzyresult: Formal class FuzzyData
- rank: num [1:8, 1] 3 6 1 5 2 3 NA NA
- result: num [1:8, 1] 3.89 2.57 7.15 3.64 5.89 ...
- textMatrix: 3 obs. of 3 variables
- values: num [1:8, 1:3] 4 1 8 3 6 4 NA NA 5 3 ...

Files Plots Packages Help Viewer

R: Function to calculate result of AHP

calculateAHP (FuzzyAHP) R Documentation

Function to calculate result of AHP

Description

This function calculates output of AHP based on `Weights` or `FuzzyWeights` on data represented either by matrix or `FuzzyData`.

Usage

```
calculateAHP(weights, data)
```

S4 method for signature "Weights,matrix"
calculateAHP(weights, data)
S4 method for signature "FuzzyWeights,matrix"

Console Terminal

```

> fuzzyresult = calculateAHP(fuzzyComparisonMatrix, values); fuzzyresult
An object of class "FuzzyData"
Slot "fmin":
  [,1]
[1,] 3.562467
[2,] 2.175840
[3,] 6.994864
[4,] 3.243323
[5,] 5.384559
[6,] 3.562467
[7,] NA
[8,] NA

```

So, fuzzy result we have, so in the fuzzy result, we really look at the kind of result that is displayed is like max value and you know modal value and you know mean value, this is corresponding to you know triangular fuzzy number having 3 values, so the same correspondence with the same correspondence this model result has been generated, the fuzzy result has been generated.

So, if you are interested in extracting you know the result for just you know a particular alternative, so that is something that can be done using the get fuzzy number function, so for example you want to extract the result for second alternative, so it is actually the second index that we are referring to, so we can use the get fuzzy number and pass on the results in the first argument that is fuzzy result here and you know second index and we will get this.

(Refer Slide Time: 29:24)

```

67
68 # Fuzzy AHP modeling
69 fuzzyresult = calculateAHP(fuzzyComparisonMatrix, values); fuzzyresult
70
71 # Extract 2nd index from 'fuzzyresult'
72 fuzzyNumber = getFuzzyNumber(fuzzyresult, as.integer(2))
73 print(fuzzyNumber)
74
75 # Ranking via Defuzzification using vager's index to obtain single output/rank
76 defuzzified = defuzzify(fuzzyresult, "vager")
77 print(defuzzified)
78
79 fuzzyrank = (rownames(values)-1) -
80 sum(is.na(defuzzified))-
81 rank(defuzzified, na.last = "keep", ties.method = "max")
82
83 print(fuzzyrank)
84
76/29 [Ctrl+Q]
C:\Users\T Cell\Desktop\MOOC January 2019\Dr. Gaurav Datta\Lecture 19\MCDM\
[4.] 4.078650
[5.] 4.421476
[6.] 4.237044
[7.] NA
[8.] NA
> # Extract 2nd index from 'fuzzyresult'
> fuzzyNumber = getFuzzyNumber(fuzzyresult, as.integer(2))
> print(fuzzyNumber)
  minimal      modal      maximal
[1.] 2.17584 2.566257 3.055858

```

So, if we run this number, so we can see the numbers for second alternative, so these are the numbers for second alternatives, so in that fashion, you can actually extract, if there are number of alternatives and you are looking to analyse certain alternative what was you know performance, what was the score for that so, you know that you can always extract. Now, the results that we have got they are in terms of you know that is using fuzzy data.

So, triangular fuzzy numbers that is you know difficult to comprehend for you know regular users, so therefore what we can do is; we can do a de-fuzzification, so de-fuzzification will actually help us you know give us the single output, right, so instead of having 3 you know values as per the triangular fuzzy numbers, we can have the de-fuzzification, so we have a vager's index which can be used to defuzzify the fuzzy result.

So, we are using this function defuzzify and the fuzzy result is going to be passed on and the method is vagers.

(Refer Slide Time: 30:32)

```

67
68 # Fuzzy AHP modeling
69 fuzzyresult = calculateAHP(fuzzycomparisonMatrix, values); fuzzyresult
70
71 # Extract 2nd index from 'fuzzyresult'
72 fuzzyNumber = getFuzzyNumber(fuzzyresult, as.integer(2))
73 print(fuzzyNumber)
74
75 # Ranking via Defuzzification using 'yager's index to obtain single output/rank
76 defuzzified = defuzzify(fuzzyresult, "yager")
77 print(defuzzified)
78
79 fuzzyRank = (nrow(values)+1) -
80 sum(is.na(defuzzified)) -
81 rank(defuzzified, na.last = "keep", ties.method = "max")
82
83 print(fuzzyRank)
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

Environment

comparisonMatrix	Formal class PairwiseComparisonMatrix
defuzzified	num [1:8, 1] 3.9 2.6 7.14 3.65 5.9 ...
fullresult	num [1:8, 1:5] 4 1 8 3 6 4 NA NA 5 3 ...
fuzzycomparison	Formal class FuzzyPairwiseComparisonMatrix
fuzzyNumber	num [1, 1:3] 2.18 2.57 3.06
fuzzyresult	Formal class FuzzyData
rank	num [1:8, 1] 3 6 1 5 2 3 NA NA
result	num [1:8, 1] 3.89 2.57 7.15 3.64 5.89 ...

Console

```

> defuzzified = defuzzify(fuzzyresult, "yager")
> print(defuzzified)
[1,]
[2,] 3.892350
[3,] 2.59318
[4,] 7.135438
[5,] 3.614089
[6,] 5.899425
[7,] 3.897250
[8,] NA
[9,] NA
>

```

Function to calculate result of AHP

Description

This function calculates output of AHP based on **Weights** or **FuzzyWeights** on data represented either by matrix or **FuzzyData**.

Usage

```
calculateAHP(weights, data)
```

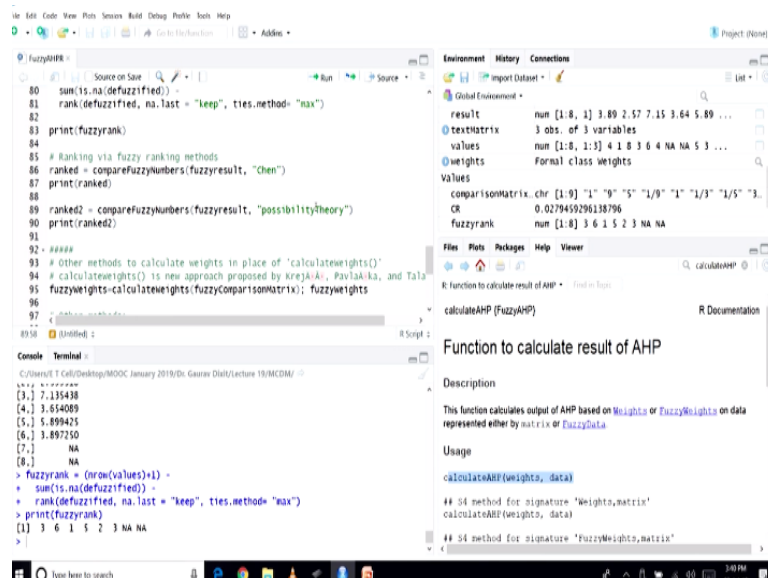
S4 method for signature "Weights,matrix"
calculateAHP(weights, data)
S4 method for signature "FuzzyWeights,matrix"

And if we run this code, you know print this results you can see these values are there, so this result has been defuzzified and you will see the numbers are quite similar to what we had got in the traditional AHP or in the result that we have just generated, right, the this second fn model numbers are you know and the numbers that we have just got are similar. The most possible values; so, what is actually happening is we are getting the you know that column associated with the most you know possible value.

Now, we want to assign rank here, so that also something that we can do here, the code that you can see here is slightly you know slightly different in the sense that we have the you know alternative same 8, they are having NA values, so the way we are you know the way we are going to rank this, we can write the code in this fashion where you know total number of values you know will have 8 +1 and NA values are 2.

So that is going to be you know subtracted, so that we will have you know 6 and then the rank number that we get is like index, so you know we will get the fuzzy rank.

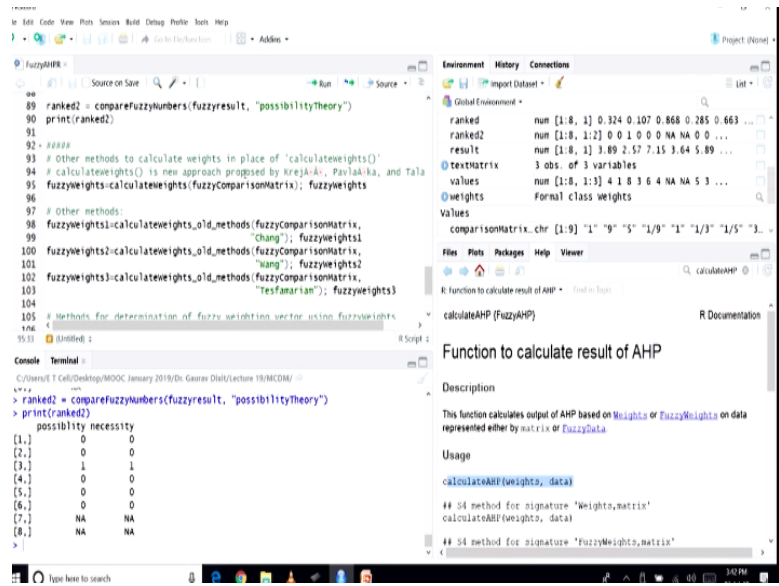
(Refer Slide Time: 31:46)



Just then look at the result, we will get the what we are trying to do here, you can see the alternative number 3 is ranked 1, so we can see it has the highest value, so this is another way to present the ranking now, you know this was one way you know ranking via de-fuzzification, and another way of ranking is using fuzzy ranking methods, so we have a number of methods available in this package.

So, we have this function, compare fuzzy numbers, so there we can pass on fuzzy result as the first argument and the different methods are available for example, Chen here and the possibility theory, so these are different methods are available which can be used to actually you know to print the results, so let us run this.

(Refer Slide Time: 32:39)

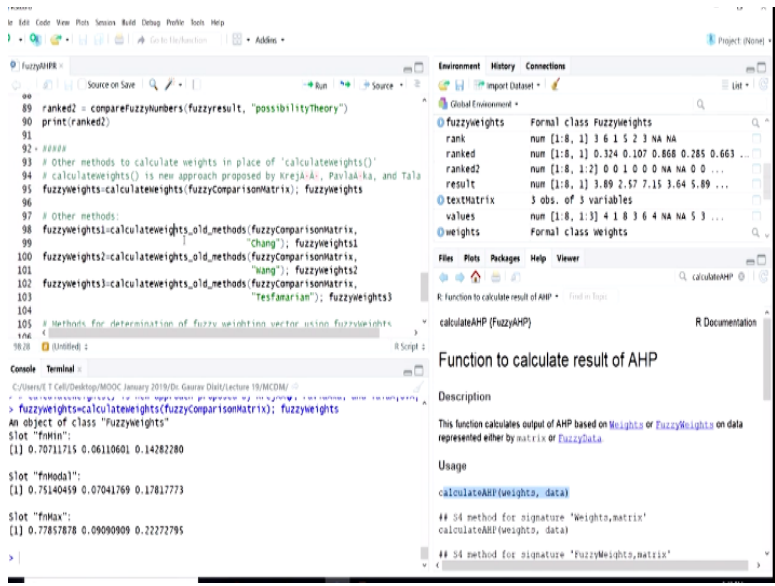


We can see again you know we have got the numbers similarly, the possibility theory method, we can see the numbers are in different weights in format, here you can see that rank alternative number 3, so this is 1 1, so here we would see that you know if there are very large number of you know best solution that are there, then this program method could be really useful but if there are really 1 dominant solution, then this region look quite descriptive here.

But if there are number of you know best solution available then this method can be really useful in terms of identifying that which alternatives were actually part of this best solution, so these are different methods to actually describe the results. Then the in the calculated AHP method, that we used for fuzzy AHP modelling as well as the traditional AHP modelling, so there you know we can you know pass on in the recent AHP modelling, we actually passed on the weights argument.

However, when we did the fuzzy AHP modelling, you would have notice that we passed on the fuzzy pairwise comparison matrix directly instead of the weights, however there are methods available which can be used to compute the weights as well rather fuzzy weights, so the default approach is that we have is you know the calculate weight function, so this is the new approach that has been proposed you know in 2016, you can you know the you can see here.

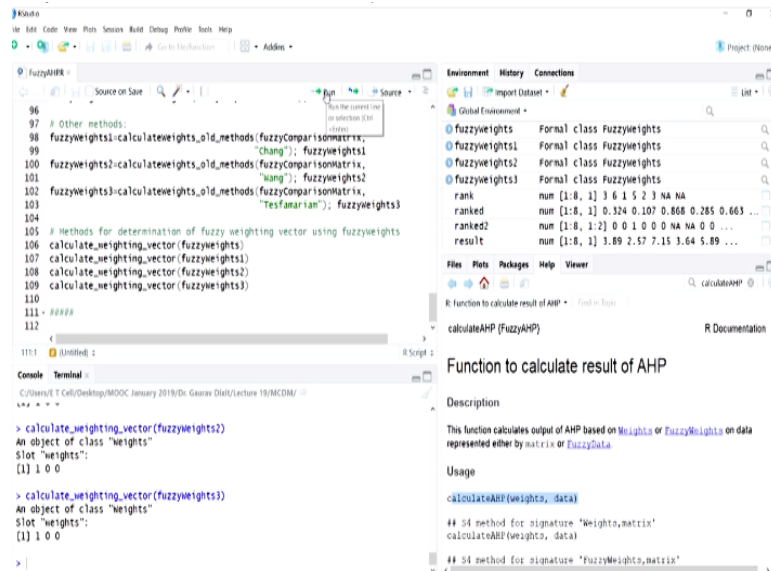
(Refer Slide Time: 34:25)



So that is the default approach, so you can run this code and see the values here, the other approach is which are based on Chang and the second approach is wang and, so these are the old approaches so those function calculated weight_ old methods is available which can be used and the second argument is you know can be used to specify the actual method and we will get the weights as per the you know these methods as well.

So you can see the Chang method is also implemented, so you can compute the you know the weights; fuzzy weights using this particular method and pass on these weights to the you know calculate AHP you know function and you can do the modelling you know as for the way it was suggested by you know Chang, something that we have discussed in the you know previous lecture as well.

(Refer Slide Time: 35:17)



So, the methods are; similar methods are been implemented here, so we run this, so if you try to compare these values that we are getting out of different results, the weight values are slightly different actually, so if we look at this, if you notice these values, for example using Chang we have just run, if you notice the most likely value, here it is .059, .069 and .092, if you just keep a tab on you know, remember these values.

And then the method that is now default method, the newest method that we have you know that is implemented in this package, the values are slightly different, this is .07, there it was .069, here it is .06, so the values are slightly different in these different methods that are implemented here you can see here, this value is .092 here, so there it was different. So, similarly other methods are also available, so we can use them.

And compare the results as well, these are available, now based on this fuzzy weights, we can also compute the fuzzy weighing you know vector, so that is also available, so this function calculate weighing vector, so the fuzzy weights that we have computed just now using 4 different methods, so those can be passed on as the first argument and we will get the weighing vector, so this is also available in this package.

So, calculate weighing vector, you will see that results are similar because the first criteria is you know that is having carrying more weight, so this was our exercise in R environment for fuzzy

AHP, you also saw that we can also do traditional AHP modelling using this package, so that completes our discussion and exercise on fuzzy AHP method, thank you.