# Multiple Criteria Decision Making I
## Analytic Hierarchy Process

孙子曰：兵者，国之大事，死生之地，存亡之道，不可不察也。故经之以五事，校之以计，而索其情：一曰道，二曰天，三曰地，四曰将，五曰法。

Sun Tze said: War is a matter of vital importance to the State; the province of life or death; the road to survival or ruin. It is mandatory that it be thoroughly studied. Therefore, appraise it in terms of the five fundamental factors and make comparisons. The factors are: (1) Moral influence; (2) Weather; (3) Terrain; (4) Command; and (5) Doctrine.

Multiple Criteria Decision Making in Sun Tze's Art of War

# 1 Introduction

- The **Analytic Hierarchy Process** (AHP) is a Multiple Criteria Decision Making (MCDM) method developed by Thomas L. Saaty.

- It is an easy-to-use decision-making tool for problem solving and decision making in complex environment that involves consideration of multiple and possibly conflicting criteria as well as in situations that require judgements on qualitative factors.

- AHP is based on four steps:

  1. **Decompositions:** A complex problem is decomposed into a hierarchy with a goal or objective at the top, each lower level consisting of a few elements; element is also, in turn, decomposed and so on. The available alternatives are usually at the lowest level of the hierarchy.

  2. **Prioritization:** The impact of the elements at each level of the hierarchy on it parent element is assessed through pairwise comparisons on a ratio scale. This produces a series of pairwise comparison matrixes which are individually evaluated to produce the elements' local weights.

  3. **Synthesis**: The priorities in Step 2 are aggregated together through the **Principle of Hierarchic Composition** to compute the overall assessments or **Global Weights** of the alternatives.

  4. **Sensitivity Analysis**: The impact on decision outcomes to changes in the importance of the elements in the hierarchy is determined via one-way sensitivity analysis to gain managerial insight about the problem.

## 2 Prioritization

### 2.1 Priority or Preference Weights

- We first concentrate on Step 2 of the AHP which is the prioritization of elements at each level of the model.

- Suppose we want to assess the **relative importance** or **priority** on a set of elements or items. These elements can be the criteria, sub-criteria, or alternatives in the AHP model.

- We express the degrees of importance or priority of each items using a set of weights which are usually *normalized* (i.e., they add up to exactly 1 or 100%).

- More formally, suppose we have $n$ items to compare, we seek a vector

$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$

such that $\sum_{i=1}^{n} w_i = 1$, where $w_i$ expresses the importance, priority or preference weight for item $i$.

- We will simply refer to the $w_i$'s as the **weights**.

### 2.2 Method of Pairwise Comparison

**Pairwise Comparison Matrix**

- Suppose we have $n$ items to compare. We first create a pairwise comparison reciprocal matrix

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} = [a_{ij}]$$

such that each element $a_{ij} = \dfrac{w_i}{w_j}$ for all $i, j$.

**Valid Pairwise Comparison Matrix**

- Matrix $\boldsymbol{A}$ is valid if and only if it satisfies the followings properties:

1. The entries of $A$ must be *positive*, *i.e.,* $a_{ij} > 0$ all $i, j$.

2. The matrix $\boldsymbol{A}$ is a *reciprocal matrix* with $a_{ij} = \dfrac{1}{a_{ji}}$ for all $i, j$.

3. The diagonal elements of $\boldsymbol{A}$ are always one, i.e. $a_{ii} = 1$, for all $i$.

**Example**

- Consider 3 elements or items $x_1$, $x_2$, $x_3$ which we wish to compare and prioritize.

- Suppose we assess that importance of $x_2$ is about *twice* that of $x_1$, the importance of $x_1$ is about *thrice* that of $x_3$, and the importance of $x_2$ is about *five times* that of $x_3$, i.e.,

$$\frac{w_2}{w_1} \approx 2; \ \frac{w_1}{w_3} \approx 3; \ \frac{w_2}{w_3} \approx 5$$

- Then $a_{21} = 2$, $a_{13} = 3$, and $a_{23} = 5$.

- The pairwise comparison matrix for $x_1$, $x_2$ and $x_3$ is as follows:

$$A = \begin{bmatrix} 1 & \frac{1}{2} & 3 \\ 2 & 1 & 5 \\ \frac{1}{3} & \frac{1}{5} & 1 \end{bmatrix}$$

- We call the pairwise comparison matrix the $A$-matrix.

**Input requirements to construct the *A*-matrix**

- In general, given $n$ items, $n(n -1)/2$ number of comparisons are needed.

**Perfectly Consistent *A*-matrix**

- A pairwise comparison matrix is said to be ***Perfectly Consistent*** if and only if

  1. It is a valid $A$-matrix, and
  2. $a_{ij} = \dfrac{a_{ik}}{a_{jk}} = a_{ik} a_{kj}$ for all $i, j$, and $k$.

**Example**

- The following matrix is perfectly consistent:

$$\begin{bmatrix} 1 & \frac{1}{2} & 3 \\ 2 & 1 & 6 \\ \frac{1}{3} & \frac{1}{6} & 1 \end{bmatrix}$$

since $a_{12} = 1/2$, $a_{13} = 3$, and $a_{23} = 6 = a_{13}/a_{12}$.

- The following matrix is valid but not perfectly consistent:

$$\begin{bmatrix} 1 & \frac{1}{2} & 3 \\ 2 & 1 & 5 \\ \frac{1}{3} & \frac{1}{5} & 1 \end{bmatrix}$$

since $a_{12} = 1/2$, $a_{13} = 3$, and $a_{23} = 5 \neq a_{13}/a_{12} = 6$.

**Relation between *A* and *w***

- Given a perfectly consistent *A*-matrix, if we post-multiply ***A*** by the column vector $\boldsymbol{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$,

  we get

$$\mathbf{A}w = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} \frac{w_1}{w_1} & \frac{w_1}{w_2} & \cdots & \frac{w_1}{w_n} \\ \frac{w_2}{w_1} & \frac{w_2}{w_2} & \cdots & \frac{w_2}{w_{n1}} \\ \vdots & \vdots & & \vdots \\ \frac{w_n}{w_1} & \frac{w_n}{w_2} & \cdots & \frac{w_n}{w_{n1}} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} n\, w_1 \\ n\, w_2 \\ \vdots \\ n\, w_n \end{bmatrix} = n\, w$$

$$\Rightarrow \qquad \boldsymbol{A}\,\boldsymbol{w} = n\,\boldsymbol{w}$$

$$\Rightarrow \qquad (\boldsymbol{A} - n\,\mathbf{I}\,)\,\boldsymbol{w} = 0.$$

- This gives us a relationship between ***A*** and ***w***.

## 2.3 Computing Priority Weights from a Pairwise Comparison Matrix

**Computing *w* when *A* is perfectly consistent.**

- If ***A*** is a Perfectly Consistent pairwise comparison matrix, then ***w*** may be computed by normalizing any column *j* of ***A***.

  i.e. $w_i = \dfrac{a_{ij}}{\sum_{k=1}^{n} a_{kj}}$ for *i* = 1 to *n*.

**Example**

- Given $A = \begin{bmatrix} 1 & \frac{1}{2} & 3 \\ 2 & 1 & 6 \\ \frac{1}{3} & \frac{1}{6} & 1 \end{bmatrix}$ which is a perfectly consistent matrix.

- Normalizing the first column, we obtain $w = \begin{bmatrix} \frac{1}{1+2+1/3} \\ \frac{2}{1+2+1/3} \\ \frac{1/3}{1+2+1/3} \end{bmatrix} = \begin{bmatrix} 0.3 \\ 0.6 \\ 0.1 \end{bmatrix}$

- The same result is obtained by normalizing column 2 or column 3.

**Computing *w* when *A* is not perfectly consistent.**

- In reality, ***A*** is usually imperfect since it is based on human expert's judgments. That is, $a_{ij} \neq a_{ik} a_{kj}$ for some *i*, *j*, *k*.

- The best estimate for ***w*** can be estimated from the relation $(\boldsymbol{A} - \lambda\, \mathbf{I})\, \boldsymbol{w} = 0$ where $\lambda$ is a constant that is approximately equal to *n*.

- But from linear algebra, $\lambda$ and ***w*** are the eigenvalue and eigenvector of ***A*** respectively.

- It can be shown that a positive reciprocal matrix has only *one real dominant* eigenvalue which shall be denoted as $\lambda_{max}$.

**Methods for finding $\lambda_{max}$ and *w* given *A***

1. Linear Algebra method (exact but solution method may be numerical).
2. Numerical method (very good solution within known tolerance limits).
3. Approximation methods (not exact, but fast and good enough; errors not controllable)

### 2.4 Linear Algebra Method

- We illustrate the Linear Algebra method with a numerical example:

- Given matrix $\boldsymbol{A} = \begin{bmatrix} 1 & \frac{1}{3} & \frac{1}{2} \\ 3 & 1 & 3 \\ 2 & \frac{1}{3} & 1 \end{bmatrix}$ which is valid but not perfectly consistent:

- The relationship between $\boldsymbol{\lambda}$ and $\boldsymbol{w}$ is the equation $(\boldsymbol{A} - \lambda\, \mathbf{I})\, \boldsymbol{w} = \mathbf{0}$, i.e.

$$
\begin{aligned}
(1-\lambda)w_1 + \tfrac{1}{3}w_2 + \tfrac{1}{2}w_3 &= 0 \\
3w_1 + (1-\lambda)w_2 + 3w_3 &= 0 \\
2w_1 + \tfrac{1}{3}w_2 + (1-\lambda)w_3 &= 0
\end{aligned}
$$

- A trivial solution is obviously $w_1 = w_2 = w_3 = 0$, and $\lambda$ is any number.

- For non- trivial solutions, the determinant of the coefficients matrix $(\boldsymbol{A} - \lambda\, \mathbf{I})$ must be zero:

$$
\mathrm{Det}(\boldsymbol{A} - \lambda \mathbf{I}) = \begin{vmatrix} 1-\lambda & \frac{1}{3} & \frac{1}{2} \\ 3 & 1-\lambda & 3 \\ 2 & \frac{1}{3} & 1-\lambda \end{vmatrix} = 0
$$

- Use cofactors expansion, we obtain a cubic equation in $\lambda$:

$$
(1-\lambda)\begin{vmatrix} 1-\lambda & 3 \\ \frac{1}{3} & 1-\lambda \end{vmatrix} - \tfrac{1}{3}\begin{vmatrix} 3 & 3 \\ 2 & 1-\lambda \end{vmatrix} + \tfrac{1}{2}\begin{vmatrix} 3 & 1-\lambda \\ 2 & \frac{1}{3} \end{vmatrix} = 0
$$

$\Rightarrow (1-\lambda)[(1-\lambda)^2 - 1] - (1/3)[3(1-\lambda) - 6] + (1/2)[1 - 2(1-\lambda)] = 0$

$\Rightarrow (1-\lambda)(1-2\lambda + \lambda^2 - 1) + (1 + \lambda) + (1/2)(2\lambda - 1) = 0$

$\Rightarrow (1-\lambda)(\lambda^2 - 2\lambda) + 1 + \lambda + \lambda - 0.5 = 0$

$\Rightarrow -\lambda^3 + 3\,\lambda^2 + 0.5 = 0$

$\Rightarrow \lambda = 3.0536$ // 2 other complex roots ignored

- We can try to find $\boldsymbol{w}$ by substituting $\lambda = 3.0536$ back into $(\boldsymbol{A} - \lambda\, \mathbf{I})\, \boldsymbol{w} = \mathbf{0}$, and solve the following set of equations:

$$
\begin{aligned}
-2.053622w_1 + \tfrac{1}{3}w_2 + \tfrac{1}{2}w_3 &= 0 \\
3w_1 - 2.053622w_2 + 3w_3 &= 0 \\
2w_1 + \tfrac{1}{3}w_2 - 2.053622w_3 &= 0
\end{aligned}
$$

- However, the 3 equations are not linearly independent. We may drop any of the 3 equations and add the normalizing equation $w_1 + w_2 + w_3 = 1$ to obtain 3 linearly independent equations.

- After replacing the first equation by the normalization equation, we solve the following 3 equations:

$$
\begin{aligned}
3w_1 - 2.053622w_2 + 3w_3 &= 0 \\
2w_1 + \tfrac{1}{3}w_2 - 2.053622w_3 &= 0 \\
w_1 + w_2 + w_3 &= 1
\end{aligned}
$$

- In matrix notations:

$$\begin{bmatrix} 3 & -2.053622 & 3 \\ 2 & 1/3 & -2.053622 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

- Solution:

$$\begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} 3 & -2.053622 & 3 \\ 2 & 1/3 & -2.053622 \\ 1 & 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.15706 \\ 0.59363 \\ 0.25931 \end{bmatrix}$$

- Hence the priority weights are:

  $w_1$ = 0.1571
  $w_2$ = 0.5936
  $w_3$ = 0.2493

- The steps for computing $\lambda_{max}$ and $\boldsymbol{w}$ using linear algebra method:

  1. Given a valid pairwise comparison matrix $\boldsymbol{A}$ of size $n$.
  2. Determine $\lambda$ by solving $|\boldsymbol{A} - \lambda\,\mathbf{I}| = \mathbf{0}$ // Can use numerical method
  3. Form a set of $n$ linear equations: $(\boldsymbol{A} - \lambda\,\mathbf{I})\,\boldsymbol{w} = \mathbf{0}$
  4. Replace any equation in Step 3 with $\sum_{j=1}^{n} w_j = 1$
  5. Determine $\boldsymbol{w}$ by solving the set of $n$ linear equations.

### Weights in Distributive and Ideal forms

- Weights are said to be expressed in **Normalized** or **Distributive Form** if they add up to 1.0 or 100%. It tells us how to allocate importance or priorities among the items. The weights in the previous example are expressed in distributive form.

- If we divide all the priority weights by the largest weight, we obtain the weights expressed in **Idealized Form:** It tells us the importance or performance of each item relative to the best (or ideal) item which is allocated a weight 1.0.

- In the above example, the weights expressed in ideal form are:

  $w_2$ = 1
  $w_3$ = 0.420
  $w_1$ = 0.265

## 2.5 The Scale for Pairwise Comparison and Consistency Measures

- Saaty recommended a 9-point scale whose validity is supported by some empirical studies. 9 is also the maximum number of concepts a person can reason without any background noise. (Miller's magic number 7±2).

**Saaty's Intensity of Importance Scale**

| Intensity of Importance | Definition | Explanation |
|---|---|---|
| 1 | Equal importance | Two activities contribute equally to the objective |
| 3 | Weak importance of one over another | Experience and judgment slightly favor one activity over another |
| 5 | Essential or strong importance | Experience and judgment strongly favor one activity over another |
| 7 | Demonstrated importance | An activity is strongly favored and its dominance demonstrated in practice |
| 9 | Absolute importance | The evidence favoring one activity over another is of the highest possible order of affirmation |
| 2,4,6,8 | Intermediate values between the two adjacent judgments | When compromise is needed |
| Reciprocals of above non-zero numbers | If activity *i* has one of the above non-zero numbers assigned to it when compared with activity *j*, then has the reciprocal when compared with *i* | |

### The Consistency Index (CI)

- In practice, we cannot ensure that the $\boldsymbol{A}$ matrix is perfectly consistent, i.e., for all $i$, $j$ and $k$, $a_{ij} = a_{ik} / a_{jk}$.

- If $\boldsymbol{A}$ were perfectly consistent, we can find the values of $\boldsymbol{w}$ by simply normalizing any column $j$ of $\boldsymbol{A}$, i.e.,

$$w_i = \frac{a_{ij}}{\sum_{k=1}^{n} a_{kj}} \quad \forall i = 1,2,\ldots,n. \qquad \text{and} \qquad \lambda_{max} = n \quad \text{exactly.}$$

- Unfortunately, some inconsistency in judgments cannot be totally avoided. However, we must ensure that the $\boldsymbol{A}$-matrix does not contain too much inconsistencies.

- It can be shown that for any positive reciprocal matrix of any size $n$, $\lambda_{max} \geq n$, and $(\lambda_{max} - n)$ increases with the amount of inconsistency in $\boldsymbol{A}$.

- Saaty defined the **Consistency Index** (CI) of $\boldsymbol{A}$ to be:

$$CI = \frac{\lambda_{\max} - n}{n - 1}$$

**(Note that actually CI is a measure of inconsistency and not of consistency)**

**Example**

- For the matrix $A = \begin{bmatrix} 1 & \frac{1}{3} & \frac{1}{2} \\ 3 & 1 & 3 \\ 2 & \frac{1}{3} & 1 \end{bmatrix}$, we have found that $\lambda = 3.0536$.

- Hence $CI = \dfrac{\lambda_{\max} - n}{n-1} = \dfrac{3.0536 - 3}{3-1} = 0.0268$

- The matrix $A$ has a CI of 0.0268. But how do we know if this is acceptable?

## The Consistency Ratio and the 10% Rule

- Consider a positive reciprocal matrix of size $n > 2$ whose entries are randomly selected from the 9-point scale. This can be done using Monte Carlo simulation.

- These matrices are those with the highest inconsistency.

- The average CI values for these matrices are called **Random Indices** (RI) and are given below:

| Size of matrix | Random Index (RI) |
|---|---|
| **3** | 0.58 |
| **4** | 0.90 |
| **5** | 1.12 |
| **6** | 1.24 |
| **7** | 1.32 |
| **8** | 1.41 |
| **9** | 1.45 |
| **10** | 1.49 |
| **11** | 1.51 |
| **12** | 1.54 |
| **13** | 1.56 |
| **14** | 1.57 |
| **15** | 1.58 |

- The **Consistency Ratio** (CR) of $A$ is defined to be:

$$\text{CR} = \frac{\text{CI of A}}{\text{RI for size } n}$$

**The 10% Rule of Practice**:

- A matrix with C.R. $\leq 0.1$ is typically considered acceptable.

- If CR $> 0.1$, there is a need to reassess some of the entries to reduce the amount of inconsistencies.

**Example**

- For the matrix $\boldsymbol{A} = \begin{bmatrix} 1 & \frac{1}{3} & \frac{1}{2} \\ 3 & 1 & 3 \\ 2 & \frac{1}{3} & 1 \end{bmatrix}$, we have found that CI =0.0268.

- Hence $\text{CR} = \dfrac{\text{CI of A}}{\text{RI for size 3}} = \dfrac{0.0268}{0.58} = 0.0463 < 0.1$

  $\Rightarrow$ Inconsistency is within acceptable limit of 0.1.

### Transitivity Relation in Pairwise Comparisons

- Given three items: $x_i$, $x_j$ and $x_k$ to be compared.

- Suppose $x_i \succ x_j$ and $x_j \succ x_k$. If $x_i \succ x_k$ then we said that the preference or importance of the items satisfy the transitivity property.

- Transitivity implies that in the pairwise comparison matrix, $a_{ij} > 1$ and $a_{jk} > 1 \Rightarrow a_{ik} > 1$.

- Ensuring transitivity in a pairwise comparison matrix is likely to achieve good consistency ratio.

**Example**

- Given the matrix:

$$\begin{bmatrix} 1 & 3 & 1/3 \\ 1/3 & 1 & 5 \\ 3 & 1/5 & 1 \end{bmatrix}, \text{ we obtain } \mathbf{w} = \begin{bmatrix} 0.33014 \\ 0.39142 \\ 0.27845 \end{bmatrix}, \ \lambda = 4.838, \ CI = 0.919, \ CR = 1.584 >>> 0.1$$

- We have
  $a_{12} = 3 > 1 \Rightarrow x_1$ is preferred to $x_2$
  $a_{23} = 5 > 1 \Rightarrow x_2$ is preferred to $x_3$
  but $a_{31} = 3 > 1 \Rightarrow x_3$ is preferred to $x_1$

- Hence transitivity is violated and the $CR$ of the matrix is very high.

- Suppose the violation was unintentional and $a_{31}$ was supposed to be 1/3.

- Fixing entries $a_{31}$ and $a_{13}$ so that transitivity is satisfied:

$$\begin{bmatrix} 1 & 3 & 3 \\ 1/3 & 1 & 5 \\ 1/3 & 1/5 & 1 \end{bmatrix}, \quad \mathbf{w} = \begin{bmatrix} 0.56660 \\ 0.32296 \\ 0.11045 \end{bmatrix}, \qquad \lambda = 3.295, \ CI = 0.14739, \ CR = 0.254 > 0.1$$

- The $CR$ has dramatically improved from 1.584 to 0.254 by just satisfying transitivity, but the new $CR$ is still not within acceptable limit. Some of the entries in the matrix should be reassessed until $CR < 0.1$.

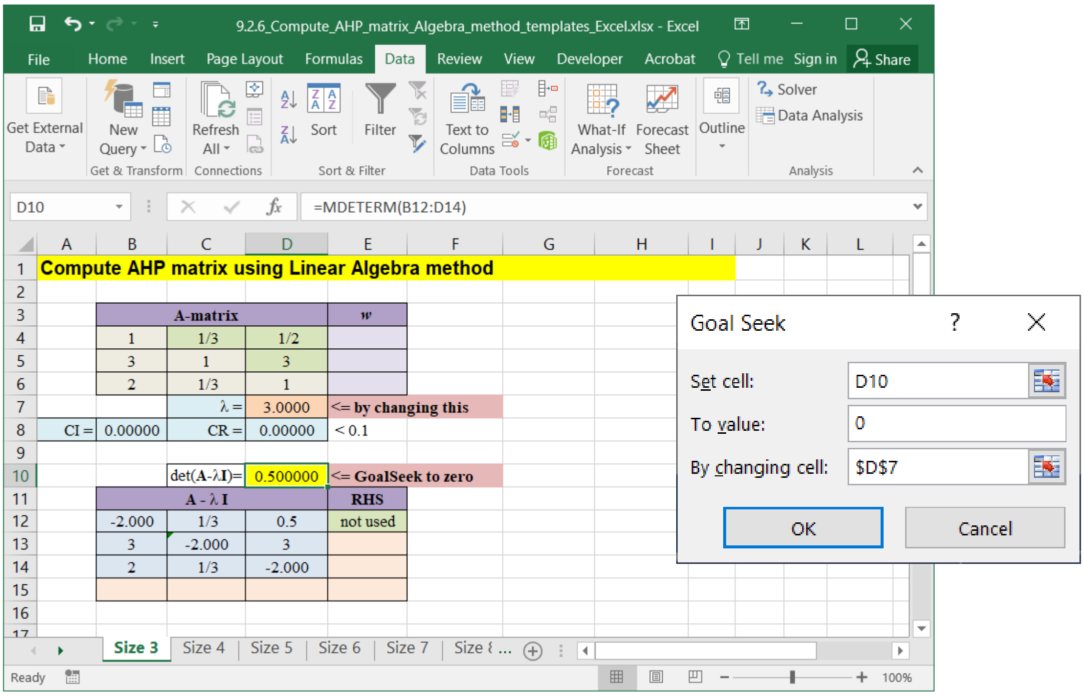
## 2.6 Using Computing Tools to Compute $w$ and $\lambda_{max}$

## 1. Using only Excel Worksheet Functions

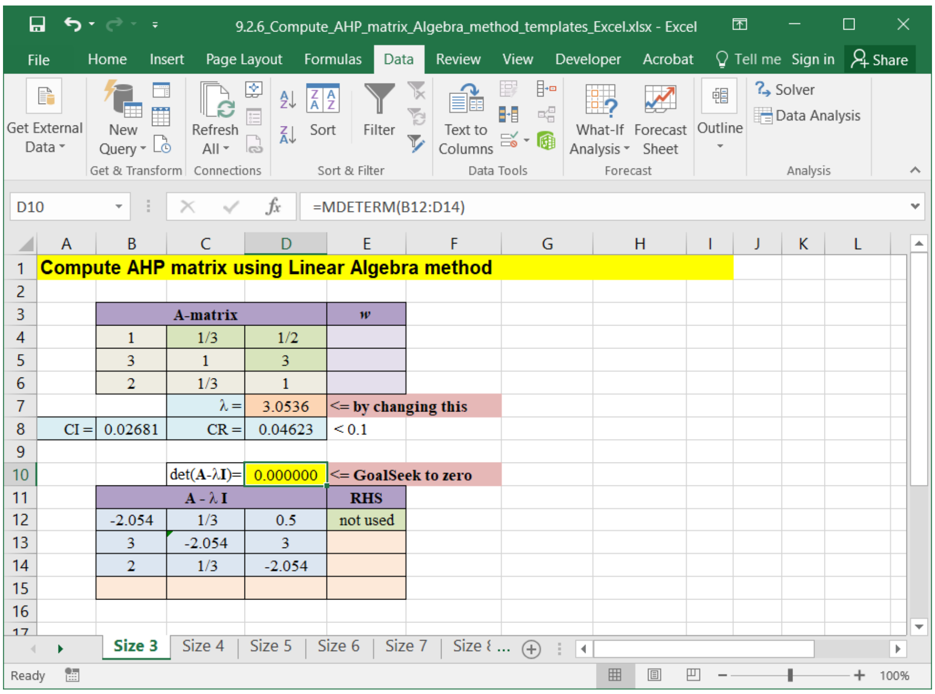
- Excel does not have a built-in function to compute eigenvalues and eigenvectors of matrices. The following process uses only Excel Worksheet Functions: Goal Seek to find $\lambda$, and then MMULT and MINVRSE functions to find $\boldsymbol{w}$. No VBA code required.

**Step 1:** Set up the pairwise comparison matrix $\boldsymbol{A}$ and compute the $\boldsymbol{A} - \lambda\, \mathbf{I}$ matrix with a guess value for $\lambda$.

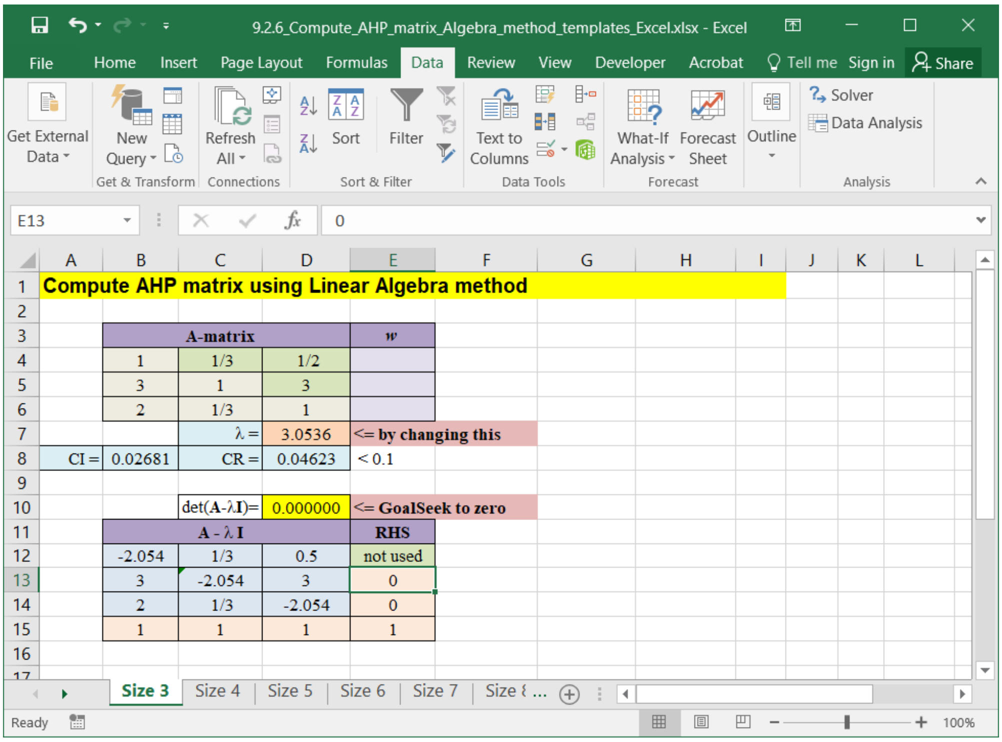
**Step 2:** Use Goal Seek to find $\lambda$ by solving the equation $|\, \boldsymbol{A} - \lambda\, \mathbf{I}\, | = 0$.



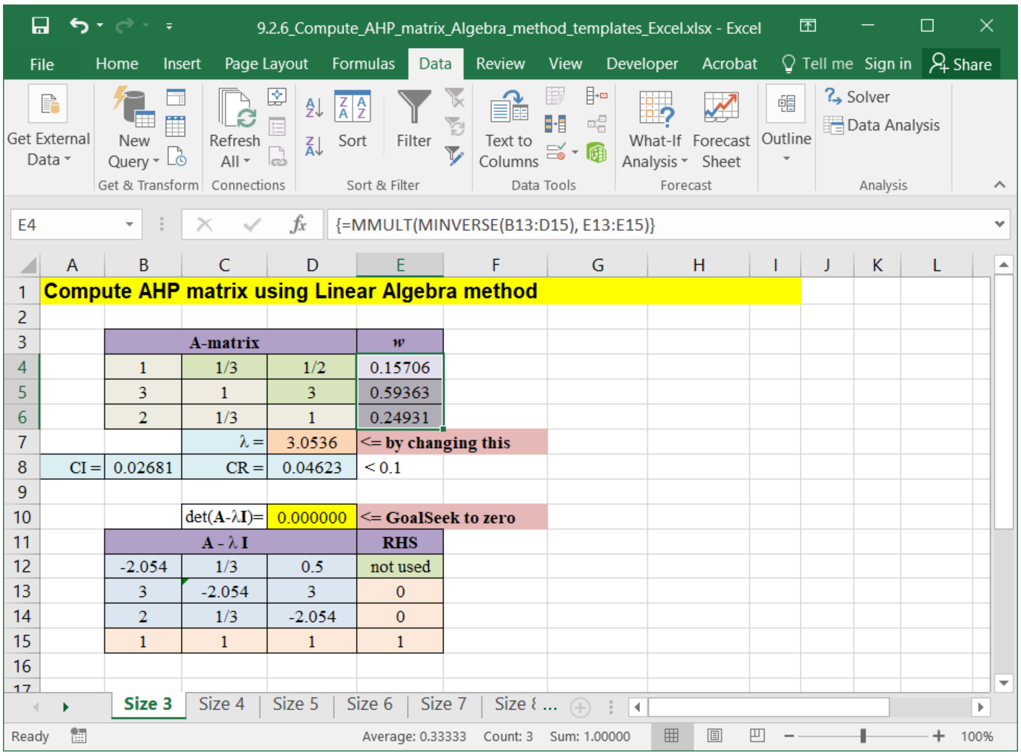

- Solution obtained:



**Step 3:** Set up the set of linear equations to solve by adding a row of all ones, and right-hand size vector of zeros.




**Step 4:** Compute $\boldsymbol{w}$ using Excel Worksheet functions MMULT and MINVERSE:




**Step 5:** Compute $CI$ and $CR$ using $\lambda$ found in Step 2.

- Results:

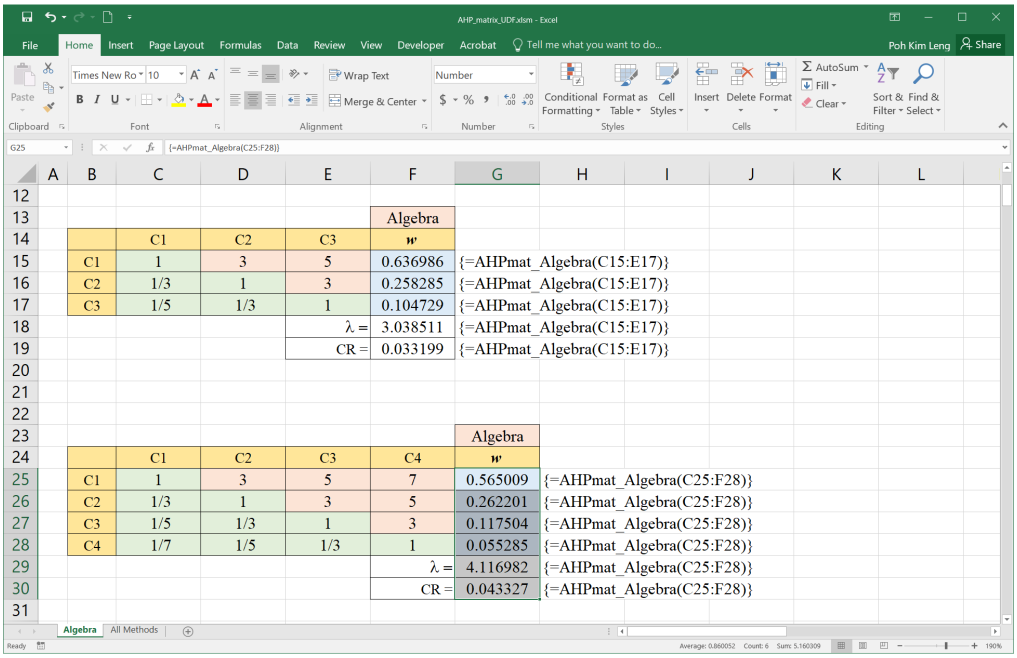
  $w_1 = 0.157056, \quad w_2 = 0.59363, \quad w_3 = 0.24931$

  $\lambda_{\max} = 3.0536, \quad CI = 0.02681, \quad CR = 0.04623 < 0.1$

## 2. Using Excel User Defined Function

- An Excel User Defined Function (UDF) can be used to hide all the messy workings above. It takes a only single argument matrix $A$ and return a column array $[w,\ \lambda_{max}, CR\ ]^T$of length $N$+2.

- The numerical bisection search method is used to solve the determinant equation to find $\lambda_{max}$. You can download the VBA source code from CodeHub.



| | C1 | C2 | C3 | Algebra *w* | |
|---|---|---|---|---|---|
| C1 | 1 | 3 | 5 | 0.636986 | {=AHPmat_Algebra(C15:E17)} |
| C2 | 1/3 | 1 | 3 | 0.258285 | {=AHPmat_Algebra(C15:E17)} |
| C3 | 1/5 | 1/3 | 1 | 0.104729 | {=AHPmat_Algebra(C15:E17)} |
| | | | λ = | 3.038511 | {=AHPmat_Algebra(C15:E17)} |
| | | | CR = | 0.033199 | {=AHPmat_Algebra(C15:E17)} |

| | C1 | C2 | C3 | C4 | Algebra *w* | |
|---|---|---|---|---|---|---|
| C1 | 1 | 3 | 5 | 7 | 0.565009 | {=AHPmat_Algebra(C25:F28)} |
| C2 | 1/3 | 1 | 3 | 5 | 0.262201 | {=AHPmat_Algebra(C25:F28)} |
| C3 | 1/5 | 1/3 | 1 | 3 | 0.117504 | {=AHPmat_Algebra(C25:F28)} |
| C4 | 1/7 | 1/5 | 1/3 | 1 | 0.055285 | {=AHPmat_Algebra(C25:F28)} |
| | | | | λ = | 4.116982 | {=AHPmat_Algebra(C25:F28)} |
| | | | | CR = | 0.043327 | {=AHPmat_Algebra(C25:F28)} |

## 3. Using Python (Linear Algebra method)

In [1]:
```
""" Compute AHP matrix using Linear Algebra method """
import numpy as np
from scipy.optimize import root
```

In [2]:
```
A = np.array([[1, 1/3, 1/2],
              [3,  1,   3 ],
              [2, 1/3,  1 ]])

n, _ = A.shape    # Number of elements
I = np.eye(n)     # Identity matrix
```

In [3]:
```
# Find lambda_max by finding root of the determinant equation
eq = lambda y: np.linalg.det(A-I*y)
sol = root(eq, x0=n, options={'xtol':1e-12})
lambda_max = sol.x[0]
print(f"lambda_max = {lambda_max:.6f}")
```

```
lambda_max = 3.053622
```

In [4]:
```
# Find w by solving a set of linear equations M w = b
M = A - I*lambda_max    # M = A - lambda_max I for first n-1 rows
M[n-1] = np.ones(n)     # Replace the last row with [1, 1..., 1]
b = np.append(np.zeros(n-1), [1])  # b = [0, 0, ..., 1]
w = np.linalg.solve(M,b)
print(f"w = {w}")
```

```
w = [0.15705579 0.59363369 0.24931053]
```

In [5]:
```
# Compute CI and CR
CI = (lambda_max-n)/(n-1)
CR = CI/0.58    # RI = 0.58 for n = 3
print(f"CI= {CI:.6f}, CR= {CR:.6f}")
```

```
CI= 0.026811, CR= 0.046225
```

## 4. Using Python numpy.linalg.eig() function

- You may also use the `numpy.linalg.eig()` function to first compute all the eigenvalues and eigenvectors (reals and complex's), and then extract out the dominant real eigenvalue and its corresponding eigenvector.

In [1]:
```
""" Compute AHP matrix using np.linalg.eig function """
import numpy as np
```

In [2]:
```
A = np.array([[1, 1/3, 1/2],
              [3,  1,   3 ],
              [2, 1/3,  1 ]])
```

In [3]:
```
# Compute all the eigenvalues and eigenvectors
eigVal, eigVec = np.linalg.eig(A)

# Get the dominant real eigenvalue and its eigenvector
lambda_max, w = max([(val.real, vec.real) for val, vec
                 in zip(eigVal, eigVec.T) if np.isreal(val)])
w = w/w.sum()  # Normalize w. Can idealize it also.
print(f"lambda_max = {lambda_max:.6f}")
print(f"w = {w}")
```

```
lambda_max = 3.053622
w = [0.15705579 0.59363369 0.24931053]
```

In [4]:
```
n, _ = A.shape
CI = (lambda_max - n)/(n - 1)
CR = CI/0.58   # RI for size 3 is 0.58
print(f"CI = {CI:.6f}, CR = {CR:.6f}")
```

```
CI = 0.026811, CR = 0.046225
```

## 2.7 Approximation Methods for Computing *w* and $\lambda_{max}$

- Approximation methods can be used to find approximate values of $\lambda_{max}$ and ***w*** very quickly with little computing resources.

- Two easy-to-use approximation methods are:
  1. Row Geometric Mean Method
  2. Column Normalization Method

## 1. Row Geometric Mean (RGM) Approximation Method

**Procedure:**

1. Given a valid pairwise comparison matrix ***A***, compute the geometric mean of each row of ***A***.

2. Normalize the numbers obtain in Step 1 to obtain an approximate ***w***.

3. For each row *i*, compute an approximate value of $\lambda$ by finding the product of the $i^{th}$ row of ***A*** and the column vector ***w***, and then dividing by $w_i$.

4. The best approximate value of $\lambda$ is obtained by averaging all the values of $\lambda$ found in Step 3.

**Example**

- Consider matrix $\boldsymbol{A} = \begin{bmatrix} 1 & \frac{1}{3} & \frac{1}{2} \\ 3 & 1 & 3 \\ 2 & \frac{1}{3} & 1 \end{bmatrix}$

| | | | Row geometric mean | Normalized weights | Approximate λ using row *i* |
|---|---|---|---|---|---|
| 1 | 1/3 | 1/2 | $\sqrt[3]{(1 \cdot \frac{1}{3} \cdot \frac{1}{2})} = 0.55032$ | **0.1571** | 3.0536 |
| 3 | 1 | 3 | $\sqrt[3]{(3 \cdot 1 \cdot 3)} = 2.08008$ | **0.5936** | 3.0536 |
| 2 | 1/3 | 1 | $\sqrt[3]{(2 \cdot \frac{1}{3} \cdot 1)} = 0.87358$ | **0.2493** | 3.0536 |
| | | | Total 3.50398 | 1.0000 | λ=3.0536 |

- Normalizing the row geometric means:
  $w_1 =$ 0.55032 / 3.50398 = 0.1571
  $w_2 =$ 2.08008 / 3.50398 = 0.5936
  $w_3 =$ 0.87358 / 3.50398 = 0.2493

- Computing an approximate value of λ using each row:
  Row 1: $\lambda \approx$ [ (1)(0.1571) + (1/3)(0.5936) + (1/2)(0.2493) ] / 0.1571 = 3.0536
  Row 2: $\lambda \approx$ [ (3)(0.1571) + (1)(0.5936) + (3)(0.2493) ] / 0.5936 = 3.0536
  Row 3: $\lambda \approx$ [ (2)(0.1571) + (1/3)(0.5936) + (1)(0.2493) ] / 0.2493 = 3.0536

- Results: ***w*** ≈ [ 0.1571, 0.5936, 0.2493]
  $\lambda \approx 3.0536$
  CI = (3.0536 – 3)/2 = 0.0268
  CR = 0.0268/0.58 = 0.0462

**Justification of the Row Geometric Mean Method**

- Given a perfectly consistent $A$-matrix:

$$\begin{bmatrix} \frac{w_1}{w_1} & \frac{w_1}{w_2} & \cdots & \frac{w_1}{w_n} \\ \frac{w_2}{w_1} & \frac{w_2}{w_2} & \cdots & \frac{w_2}{w_n} \\ \vdots & \vdots & & \vdots \\ \frac{w_n}{w_1} & \frac{w_n}{w_2} & \cdots & \frac{w_n}{w_n} \end{bmatrix}$$

- The Geometric Mean of the $i^{\text{th}}$ row of $\boldsymbol{A}$ is

$$\mu_i = \sqrt[n]{\frac{w_i}{w_1}\frac{w_i}{w_2}\cdots\frac{w_i}{w_n}} = \frac{w_i}{\sqrt[n]{w_1 w_2 \cdots w_n}} = Kw_i \qquad \text{for } i = 1 \text{ to } n$$

  where $K$ is a constant.

- By normalizing the $\mu_i$'s, we eliminate $K$ and retrieve the values of $w_i$'s.

- Hence if $\boldsymbol{A}$ is perfectly consistent, then this method gives the exact weights, and when $\boldsymbol{A}$ is not perfectly consistent, it can give a good approximation of the weights.

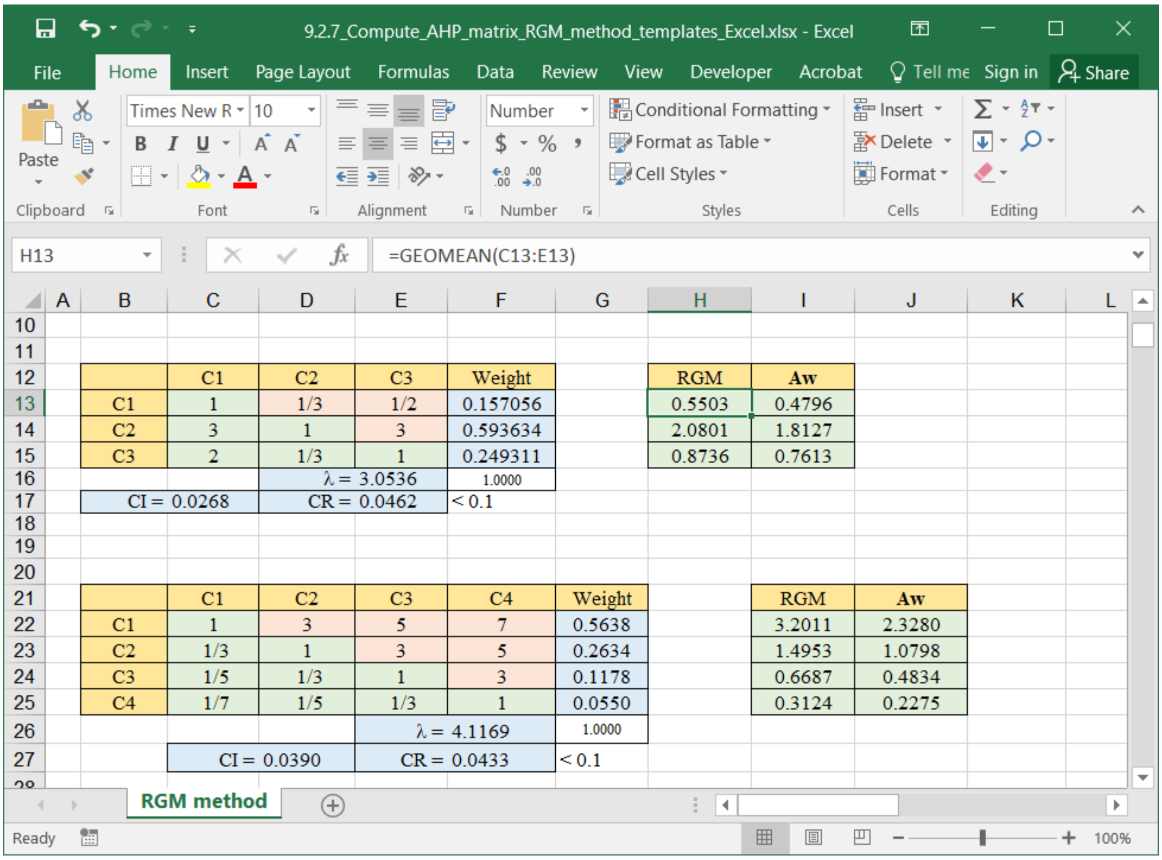
- To find $\lambda$, first we observe that if $\boldsymbol{w}$ is exact, then $\mathbf{Aw} = \lambda\mathbf{w} = \begin{bmatrix} \lambda w_1 \\ \lambda w_2 \\ \vdots \\ \lambda w_n \end{bmatrix}$.

  Hence, by dividing any $i^{\text{th}}$ entry of $\mathbf{A}\boldsymbol{w} = \lambda\boldsymbol{w}$ by $w_i$, we will get the exact value of $\lambda$.

  However, when $\boldsymbol{w}$ is not exact, we use can use all the rows of $\boldsymbol{A}$ and the approximate $\boldsymbol{w}$ to obtain $n$ approximate values of $\lambda$, and then take their average.

  Hence $\qquad \lambda_{\max} \approx \frac{1}{n}\sum_{i=1}^{n}\lambda_i \qquad$ where $\lambda_i = \frac{\sum_{j=1}^{n} a_{ij} w_j}{w_i}$ for $i = 1$ to $n$.

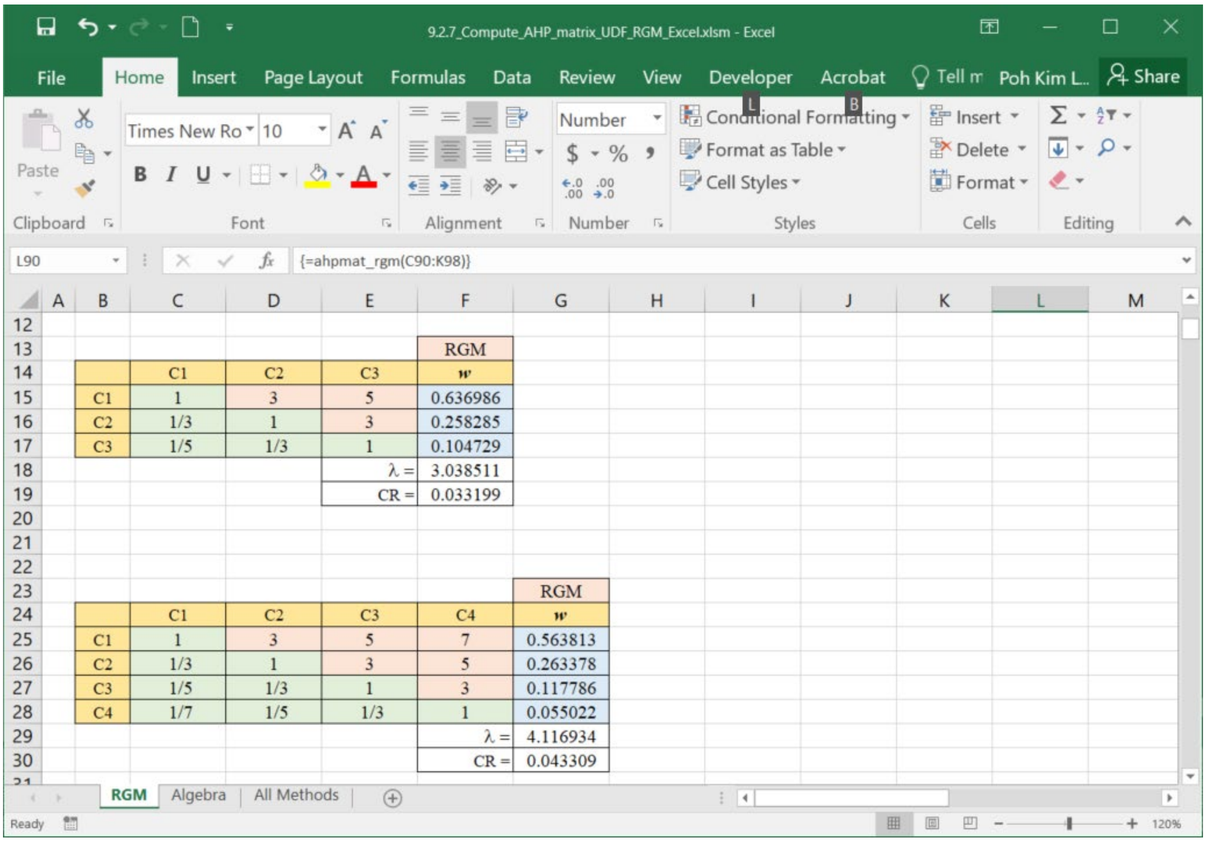
## Using only Excel Worksheet Functions (no VBA)




| | C1 | C2 | C3 | Weight | | RGM | Aw |
|---|---|---|---|---|---|---|---|
| C1 | 1 | 1/3 | 1/2 | 0.157056 | | 0.5503 | 0.4796 |
| C2 | 3 | 1 | 3 | 0.593634 | | 2.0801 | 1.8127 |
| C3 | 2 | 1/3 | 1 | 0.249311 | | 0.8736 | 0.7613 |
| | | λ = 3.0536 | | 1.0000 | | | |
| CI = 0.0268 | | CR = 0.0462 | | < 0.1 | | | |

| | C1 | C2 | C3 | C4 | Weight | RGM | Aw |
|---|---|---|---|---|---|---|---|
| C1 | 1 | 3 | 5 | 7 | 0.5638 | 3.2011 | 2.3280 |
| C2 | 1/3 | 1 | 3 | 5 | 0.2634 | 1.4953 | 1.0798 |
| C3 | 1/5 | 1/3 | 1 | 3 | 0.1178 | 0.6687 | 0.4834 |
| C4 | 1/7 | 1/5 | 1/3 | 1 | 0.0550 | 0.3124 | 0.2275 |
| | | | λ = 4.1169 | | 1.0000 | | |
| | CI = 0.0390 | | CR = 0.0433 | | < 0.1 | | |

## Using Excel User Defined Function (VBA)




| | C1 | C2 | C3 | RGM $w$ |
|---|---|---|---|---|
| C1 | 1 | 3 | 5 | 0.636986 |
| C2 | 1/3 | 1 | 3 | 0.258285 |
| C3 | 1/5 | 1/3 | 1 | 0.104729 |
| | | | λ = | 3.038511 |
| | | | CR = | 0.033199 |

| | C1 | C2 | C3 | C4 | RGM $w$ |
|---|---|---|---|---|---|
| C1 | 1 | 3 | 5 | 7 | 0.563813 |
| C2 | 1/3 | 1 | 3 | 5 | 0.263378 |
| C3 | 1/5 | 1/3 | 1 | 3 | 0.117786 |
| C4 | 1/7 | 1/5 | 1/3 | 1 | 0.055022 |
| | | | | λ = | 4.116934 |
| | | | | CR = | 0.043309 |

## Using Python

In [1]:
```
""" Compute AHP matrix using Row Geometric Mean approximation method """
import numpy as np
from scipy.stats import gmean
```

In [2]:
```
A = np.array([[ 1, 1/3, 1/2],
              [ 3,   1,   3 ],
              [ 2, 1/3,   1 ]] )
```

In [3]:
```
# Compute the geometric mean of each row, then normalize it.
rgm = gmean(A, axis=1)
w = rgm/rgm.sum()
print(f"w = {w}")
```

```
w = [0.15705579 0.59363369 0.24931053]
```

In [4]:
```
# Estimate lambda_max using all rows
lambda_max = (np.dot(A,w)/w).mean()
n, _ = A.shape
CI = (lambda_max-n)/(n-1)
CR = CI/0.58
print(f"lambda_max= {lambda_max:.6f}, CI= {CI:.6f}, CR= {CR:.6f}")
```

```
lambda_max= 3.053622, CI= 0.026811, CR= 0.046225
```

## 2. The Column Normalization Approximation Method

**Procedure:**

1. Given a valid pairwise comparison matrix $\boldsymbol{A}$, normalize each column of $\boldsymbol{A}$.

2. Compute the average across each row of the matrix to obtain an approximate $\boldsymbol{w}$.

3. For each row $i$, compute an approximate value of $\lambda$ by finding the product of the $i^{th}$ row of $\boldsymbol{A}$ and the column vector $\boldsymbol{w}$, and then dividing by $w_i$.

4. An approximate value of $\lambda$ is obtained by averaging all the values of $\lambda$ in Step 3.

**Example**

- Consider matrix $\boldsymbol{A} = \begin{bmatrix} 1 & \frac{1}{3} & \frac{1}{2} \\ 3 & 1 & 3 \\ 2 & \frac{1}{3} & 1 \end{bmatrix}$

| | | | Normalized columns | | | Row Average | Approximate λ using row *i* |
|---|---|---|---|---|---|---|---|
| | | | Column 1 | Column 2 | Column 3 | | |
| 1 | 1/3 | 1/2 | 0.1667 | 0.2000 | 0.1111 | **0.1593** | 3.0226 |
| 3 | 1 | 3 | 0.5000 | 0.6000 | 0.6667 | **0.5889** | 3.0942 |
| 2 | 1/3 | 1 | 0.3333 | 0.2000 | 0.2222 | **0.2518** | 3.0449 |
| 6.000 | 1.667 | 4.500 | | | | 1.0000 | λ=**3.0539** |

- Computing approximate values of λ using each row:

Row 1: $\lambda \approx [\,(1)(0.1593) + (1/3)(0.5889) + (1/2)(0.2518)\,] / 0.1593 = 3.0226$
Row 2: $\lambda \approx [\,(3)(0.1593) + (1)(0.5889) + (3)(0.2518)\,] / 0.5889 = 3.0942$
Row 3: $\lambda \approx [\,(2)(0.1593) + (1/3)(0.5889) + (1)(0.2518)\,] / 0.2518 = 3.0449$

- Results: $\boldsymbol{w} \approx [\,0.1593, 0.5889, 0.2518]$
  $\lambda \approx 3.0539$
  CI = (3.0539 – 3)/2 = 0.0270
  CR = 0.0270/0.58 = 0.0465

**Justification of the Column Normalization Method**

- Given a perfectly consistent $A$-matrix:

$$\begin{bmatrix} \frac{w_1}{w_1} & \frac{w_1}{w_2} & \cdots & \frac{w_1}{w_n} \\ \frac{w_2}{w_1} & \frac{w_2}{w_2} & \cdots & \frac{w_2}{w_n} \\ \vdots & \vdots & & \vdots \\ \frac{w_n}{w_1} & \frac{w_n}{w_2} & \cdots & \frac{w_n}{w_n} \end{bmatrix}$$

- By normalizing each column, we obtain the following matrix

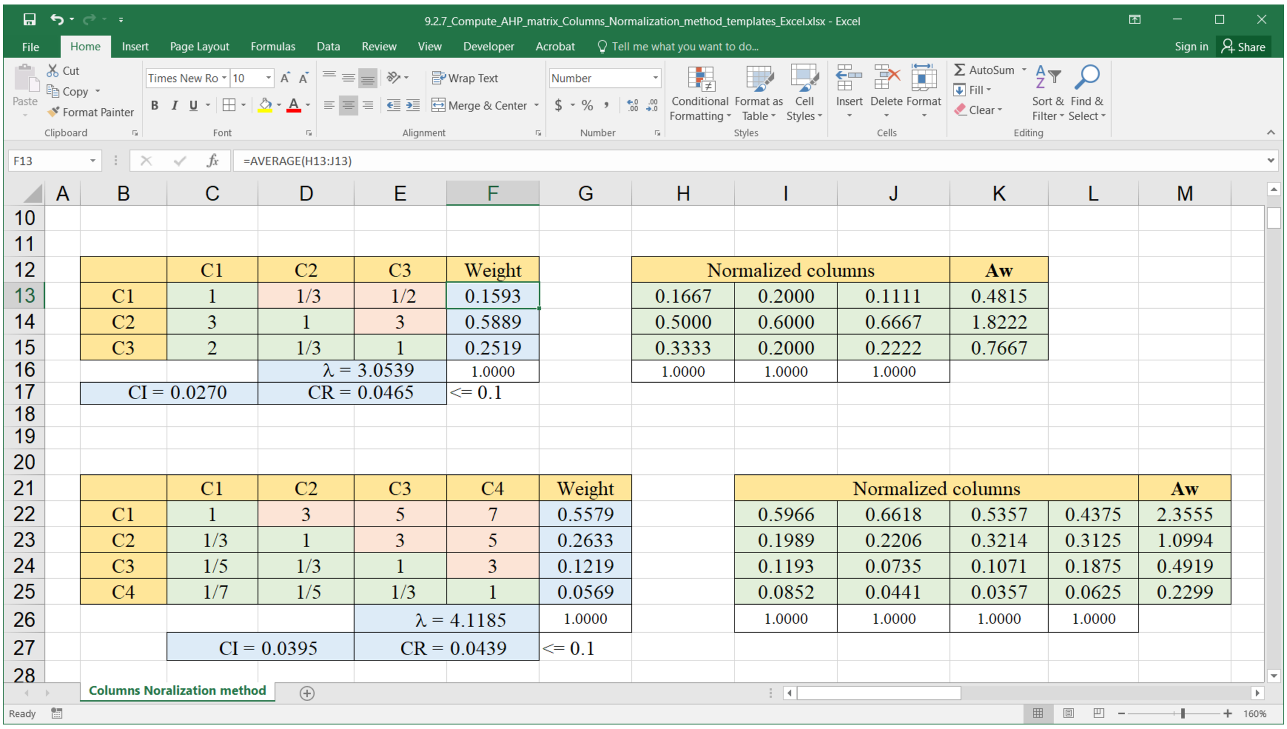
$$\begin{bmatrix} w_1 & w_1 & \cdots & w_1 \\ w_2 & w_2 & \cdots & w_2 \\ \vdots & \vdots & & \vdots \\ w_n & w_n & \cdots & w_n \end{bmatrix}$$

- Hence if $\boldsymbol{A}$ is perfectly consistent, the exact weights may be obtained by normalizing any columns of $\boldsymbol{A}$.

- However, when A is not perfectly consistent, each normalized columns will be slightly different, and by averaging across the rows, we obtain a good approximation for the weights.

- The step for computing an approximate value of λ is the same as in the RGM method.

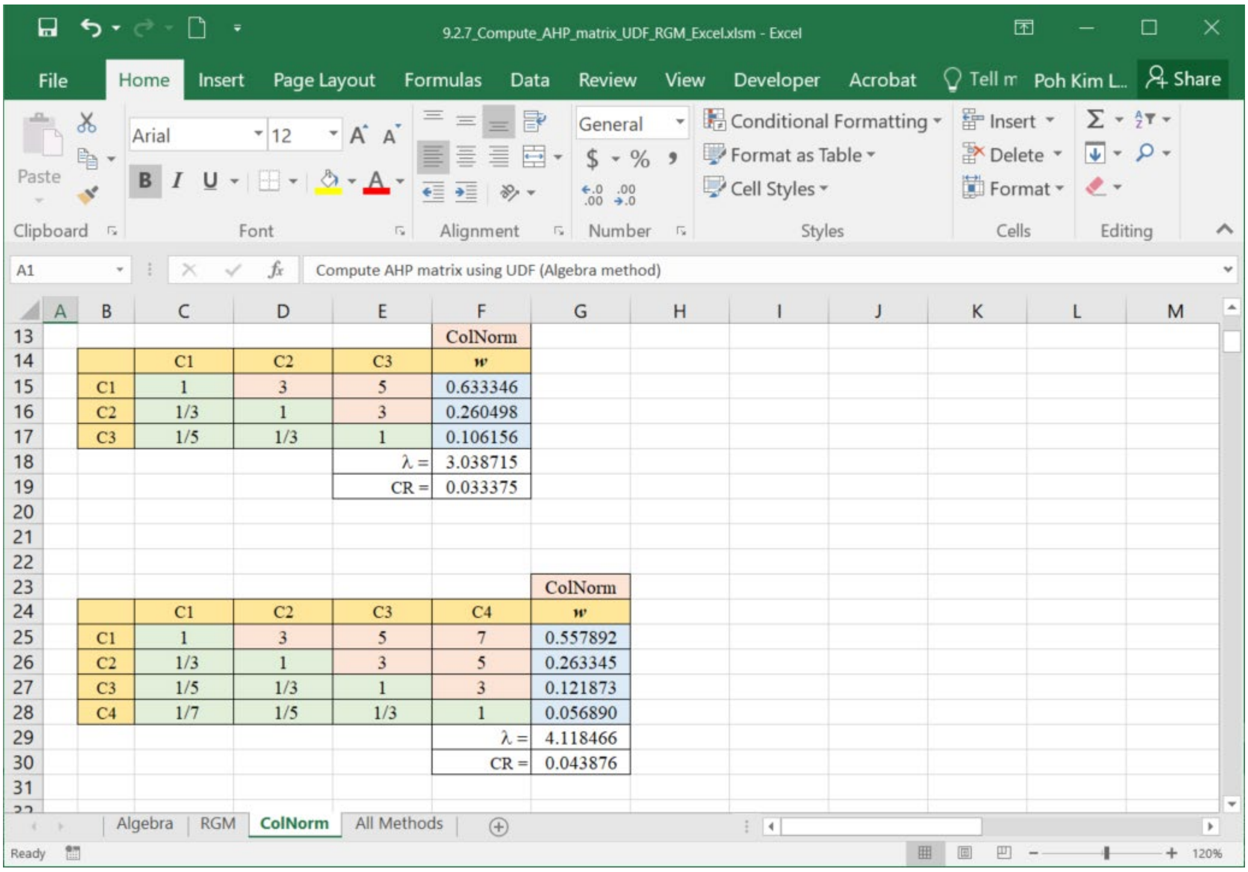
## Using only Excel Worksheet Functions




| | C1 | C2 | C3 | Weight | | Normalized columns | | | Aw |
|---|---|---|---|---|---|---|---|---|---|
| C1 | 1 | 1/3 | 1/2 | 0.1593 | | 0.1667 | 0.2000 | 0.1111 | 0.4815 |
| C2 | 3 | 1 | 3 | 0.5889 | | 0.5000 | 0.6000 | 0.6667 | 1.8222 |
| C3 | 2 | 1/3 | 1 | 0.2519 | | 0.3333 | 0.2000 | 0.2222 | 0.7667 |
| | | | λ = 3.0539 | 1.0000 | | 1.0000 | 1.0000 | 1.0000 | |
| CI = 0.0270 | | CR = 0.0465 | | <= 0.1 | | | | | |

| | C1 | C2 | C3 | C4 | Weight | Normalized columns | | | | Aw |
|---|---|---|---|---|---|---|---|---|---|---|
| C1 | 1 | 3 | 5 | 7 | 0.5579 | 0.5966 | 0.6618 | 0.5357 | 0.4375 | 2.3555 |
| C2 | 1/3 | 1 | 3 | 5 | 0.2633 | 0.1989 | 0.2206 | 0.3214 | 0.3125 | 1.0994 |
| C3 | 1/5 | 1/3 | 1 | 3 | 0.1219 | 0.1193 | 0.0735 | 0.1071 | 0.1875 | 0.4919 |
| C4 | 1/7 | 1/5 | 1/3 | 1 | 0.0569 | 0.0852 | 0.0441 | 0.0357 | 0.0625 | 0.2299 |
| | | | | λ = 4.1185 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | |
| | CI = 0.0395 | | CR = 0.0439 | | <= 0.1 | | | | | |

# Using Excel User Defined Function




| | | | | ColNorm |
|---|---|---|---|---|
| | C1 | C2 | C3 | w |
| C1 | 1 | 3 | 5 | 0.633346 |
| C2 | 1/3 | 1 | 3 | 0.260498 |
| C3 | 1/5 | 1/3 | 1 | 0.106156 |
| | | | λ = | 3.038715 |
| | | | CR = | 0.033375 |

| | | | | | ColNorm |
|---|---|---|---|---|---|
| | C1 | C2 | C3 | C4 | w |
| C1 | 1 | 3 | 5 | 7 | 0.557892 |
| C2 | 1/3 | 1 | 3 | 5 | 0.263345 |
| C3 | 1/5 | 1/3 | 1 | 3 | 0.121873 |
| C4 | 1/7 | 1/5 | 1/3 | 1 | 0.056890 |
| | | | | λ = | 4.118466 |
| | | | | CR = | 0.043876 |

# Using Python

In [1]:
```python
""" Compute AHP matrix using Column Normalization approximation method """
""" This method is fast but not very accurate; Use RGM method instead  """
import numpy as np
```

In [2]:
```python
A = np.array([[1, 1/3, 1/2],
              [3,  1,   3 ],
              [2, 1/3,  1 ]])
```

In [3]:
```python
# Normalise each column of A, then average across each row.
w = (A/A.sum(axis=0)).mean(axis=1)
print(f"w = {w}")
```

```
w = [0.15925926 0.58888889 0.25185185]
```

In [4]:
```python
# Estimate lambda_max using all rows
lambda_max = (np.dot(A,w)/w).mean()
n, _ = A.shape
CI = (lambda_max-n)/(n-1)
CR = CI/0.58
print(f"lambda_max= {lambda_max:.6f}, CI = {CI:.6f}, CR= {CR:.6f}")
```

```
lambda_max= 3.053904, CI = 0.026952, CR= 0.046469
```

### Comparison of Approximation Methods

| | Exact Analytical Method | Geometric Mean Method | Column Normalization Method |
|---|---|---|---|
| $w_1$ | 0.15705579 | 0.15705579 | 0.15925926 |
| $w_2$ | 0.59363369 | 0.59363369 | 0.58888889 |
| $w_3$ | 0.24931053 | 0.24931053 | 0.25185185 |
| $\lambda_{max}$ | 3.053622 | 3.053622 | 3.053904 |

**Notes:**

- Row Geometric Mean Method provides better results than the Column Normalization Method.
- RGM method is very good for small matrix size, but the errors will increase as the size increases.
- Use the RGM method if you have to use an approximation method.
- Use the Column Normalization method only if you do not have a scientific calculator to find the $n^{th}$ roots.

## 2.8 Numerical Method for Computing $w$ and $\lambda_{max}$

### Power Iterations method

- The Power Iterations method can be used to find the dominant eigenvalue and its's corresponding eigenvector by numerical iterations until the solution converges to within some tolerance limits.

**Algorithm:**

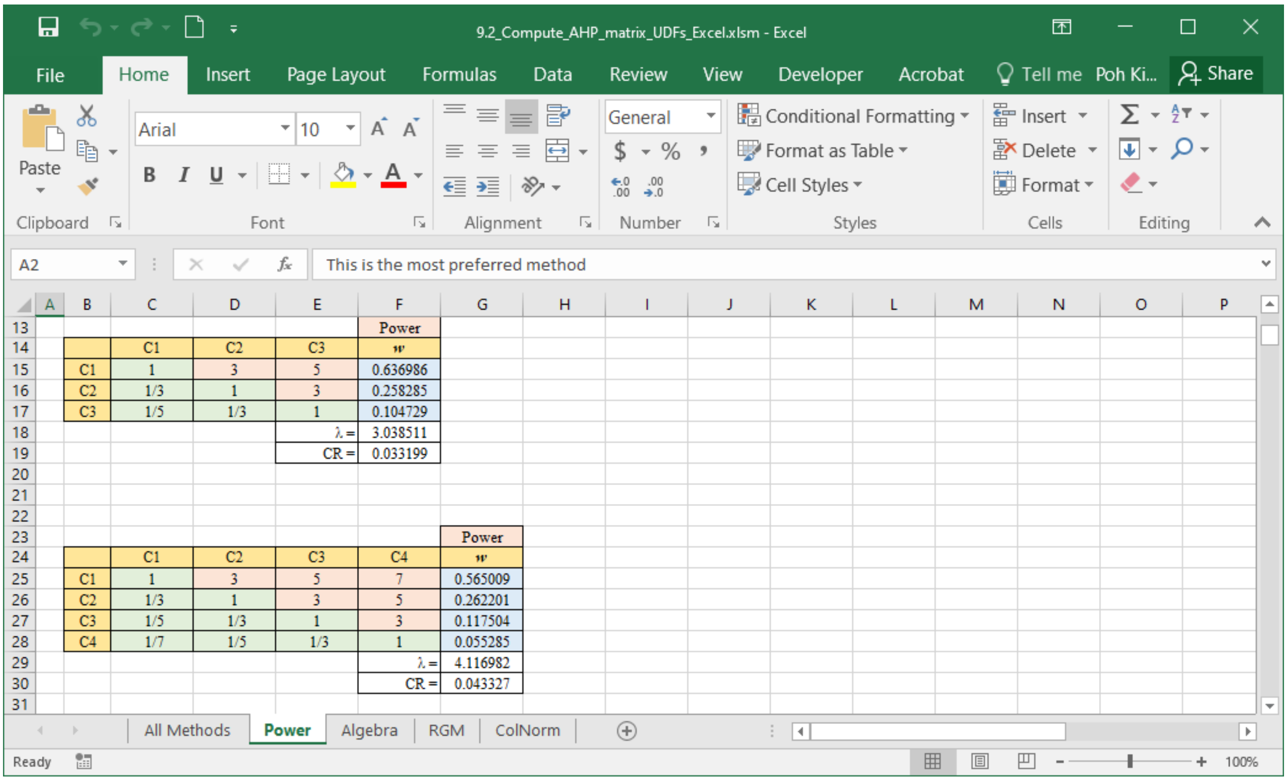
Given a valid matrix $\boldsymbol{A}$.

Let $\boldsymbol{w}_0$ = initial guess by an approximation method.
Iteration $k = 0$

**While** $k \leq$ max_iterations
$\quad \boldsymbol{w}_{k+1} = \boldsymbol{A}\, \boldsymbol{w}_k$
$\quad \boldsymbol{w}_{k+1} = \boldsymbol{w}_{k+1}$ / sum($\boldsymbol{w}_{k+1}$) // normalize it
$\quad$ **If** $|\, \boldsymbol{w}_{k+1} - \boldsymbol{w}_k \,| <$ tolerance:
$\quad\quad \boldsymbol{w}_k = \boldsymbol{w}_{k+1}$ // take the last value before breaking out
$\quad\quad$ **break**
$\quad$ **End If**
$\quad \boldsymbol{w}_k = \boldsymbol{w}_{k+1}$
$\quad k = k+1$
**End While**

$\boldsymbol{\lambda} = \boldsymbol{A}\boldsymbol{w}_k / \boldsymbol{w}_k$ // component wise
$\lambda_{max}$ = mean($\boldsymbol{\lambda}$)
$CI = (\lambda_{max} - n) / (n - 1)$
$CR = CI/RI_n$

### <u>Using Excel User-Defined Function</u>




| | C1 | C2 | C3 | Power $w$ |
|---|---|---|---|---|
| C1 | 1 | 3 | 5 | 0.636986 |
| C2 | 1/3 | 1 | 3 | 0.258285 |
| C3 | 1/5 | 1/3 | 1 | 0.104729 |
| | | | λ = | 3.038511 |
| | | | CR = | 0.033199 |

| | C1 | C2 | C3 | C4 | Power $w$ |
|---|---|---|---|---|---|
| C1 | 1 | 3 | 5 | 7 | 0.565009 |
| C2 | 1/3 | 1 | 3 | 5 | 0.262201 |
| C3 | 1/5 | 1/3 | 1 | 3 | 0.117504 |
| C4 | 1/7 | 1/5 | 1/3 | 1 | 0.055285 |
| | | | | λ = | 4.116982 |
| | | | | CR = | 0.043327 |

## Using Python

In [1]:
```python
""" Compute AHP matrix using Power Iterations method """
import numpy as np
from scipy.stats import gmean
```

In [2]:
```python
A = np.array([[1, 1/3, 1/2],
              [3,   1,   3 ],
              [2, 1/3,   1 ]])
```

In [3]:
```python
# Initial solution:
#   Use RGM approximation. Faster convergence
gm = gmean(A, axis=1)
w = gm/gm.sum()
#   Use column normalization. More iterations needed
# w = (A/A. sum(axis=0)).mean(axis=1)
```

In [4]:
```python
# Perform Power Iterations
max_iter= 1000000
epsilon = 1.E-16
for iter in range(max_iter):
    w1 = np.dot(A,w)     # w(k+1) = A w(k)
    w1 = w1/w1.sum()     # normalize w(k+1)
    if all(np.absolute(w1-w) < epsilon):
        w = w1
        print(f"Tolerance {epsilon} achieved at iter #{iter}")
        break
    w = w1
print(f"w = {w}")
```

```
Tolerance 1e-16 achieved at iter #2
w = [0.15705579 0.59363369 0.24931053]
```

In [5]:
```python
# Estimate lambda_max using all rows
lambda_max = (np.dot(A,w)/w).mean()
n, _ = A.shape
CI = (lambda_max-n)/(n-1)
CR =  CI/0.58
print(f"lambda_max= {lambda_max:.6f}, CI= {CI:.6f}, CR= {CR:.6f}")
```

```
lambda_max= 3.053622, CI= 0.026811, CR= 0.046225
```

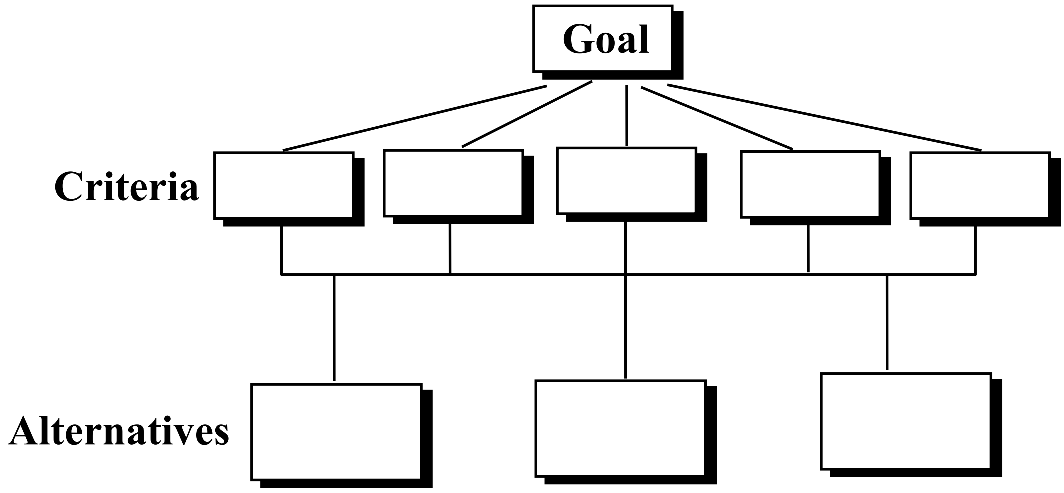
## 3 Modeling and Solving a AHP Model

### 3.1 Case Study: Job Selection Problem

- We illustrate the rest of the steps for AHP via a case study on job selection.

- Suppose a recent NUS graduate has three job offers. How should she make her choice using AHP?

**Constructing the Hierarchy:**



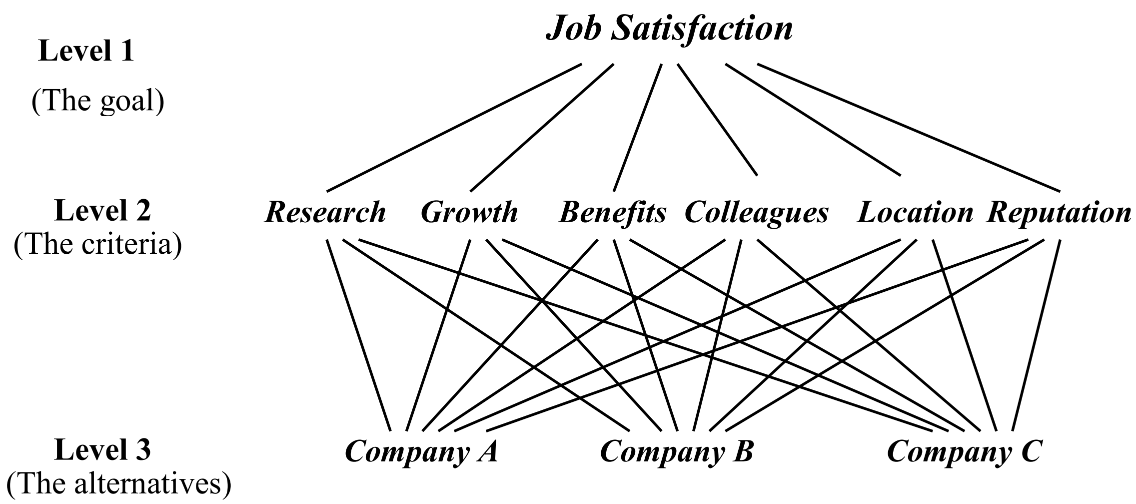

**Goal**: Job Satisfaction

**The Hierarchy** consists of three levels:

Level 1: Goal

Level 2: Criteria that contribute towards achievement of the goal in level 1.

Level 3: The alternatives under consideration.



### Performing Judgments and Computing the Local Weights

- Perform pairwise comparison of level-2 elements with respect to level 1, i.e., evaluate the contributions of each of the six criteria towards achieving the goal "Job Satisfaction".

| | Research | Growth | Benefits | Colleagues | Location | Reputation |
|---|---|---|---|---|---|---|
| **Research** | 1 | 1 | 1 | 4 | 1 | 1/2 |
| **Growth** | | 1 | 2 | 4 | 1 | 1/2 |
| **Benefits** | | | 1 | 5 | 3 | 1/2 |
| **Colleagues** | | | | 1 | 1/3 | 1/3 |
| **Location** | | | | | 1 | 1 |
| **Reputation** | | | | | | 1 |

- Solve for $\lambda_{max}$ and $\boldsymbol{w}$.
  $\lambda_{max}$= 6.4203  CI =0.08407  CR=0.06780 < 10%
  $\boldsymbol{w}$ = [0.158408 0.189247 0.197997 0.048310 0.150245 0.255792]

- That is with respect to Job satisfaction, the contributions from the various criteria are:

| | Criterion | Weight |
|---|---|---|
| 1 | Research | 15.8 % |
| 2 | Growth | 18.9 % |
| 3 | Benefits | 19.8 % |
| 4 | Colleagues | 4.8 % |
| 5 | Location | 15.0 % |
| 6 | Reputation | 25.6 % |

- Next, we perform pairwise comparison of level-3 elements (i.e., the alternative jobs) with respect to each of the six criteria in level 2. This will result in six 3×3 *A*-matrices.

- Comparison of alternatives w.r.t. "Research":

| | A | B | C |
|---|---|---|---|
| A | 1 | 1/4 | 1/2 |
| B | | 1 | 3 |
| C | | | 1 |

$\lambda_{max}$ = 3.01829, CI =0.00915, CR =0.01577 < 0.1, $\boldsymbol{w}$ = [0.13650, 0.62501, 0.23849]

- Comparison of alternatives w.r.t. "Growth":

| | A | B | C |
|---|---|---|---|
| A | 1 | 1/4 | 1/5 |
| B | | 1 | 1/2 |
| C | | | 1 |

$\lambda_{max}$ = 3.0246, CI =0.012298, CR=0.0212 < 0.1, $\boldsymbol{w}$ = [0.09739, 0.33307, 0.56954]

- Comparison of alternatives w.r.t. “Benefits”:

| | A | B | C |
|---|---|---|---|
| A | 1 | 3 | 1/3 |
| B | | 1 | 1/7 |
| C | | | 1 |

$\lambda_{max}$ = 3.0070, CI = 0.003511, CR =0.00605 < 0.1, ***w*** = [0.2426 , 0.08794, 0.6694]

- Comparison of alternatives w.r.t. “Colleagues”:

| | A | B | C |
|---|---|---|---|
| A | 1 | 1/3 | 5 |
| B | | 1 | 7 |
| C | | | 1 |

$\lambda_{max}$ = 3.0649, CI = 0.032444, CR = 0.05594 < 0.1, ***w*** = [0.27895, 0.64912, 0.07193]

- Comparison of alternatives w.r.t. “Location”:

| | A | B | C |
|---|---|---|---|
| A | 1 | 1 | 7 |
| B | | 1 | 7 |
| C | | | 1 |

$\lambda_{max}$ = 3, CI = 0, CR = 0 < 0.1, ***w***= [0.46667, 0.46667, 0.06667]

- Comparison of alternatives w.r.t. “Reputation”:

| | A | B | C |
|---|---|---|---|
| A | 1 | 7 | 9 |
| B | | 1 | 2 |
| C | | | 1 |

$\lambda_{max}$= 3.0217, CI = 0.01086, CR = 0.01873 < 0.1, ***w*** = [0.79276, 0.13122, 0.07602]

## Computing the Global Weights of the Alternatives

- The results so far may be summarized as follows:

| | Criterion | Criterion's weight | Alternative | Alt's local weights w.r.t. criterion |
|---|---|---|---|---|
| 1 | Research | **0.158408** | | |
| | | | Company A | 0.13650 |
| | | | Company B | 0.62501 |
| | | | Company C | 0.23849 |
| 2 | Growth | **0.189247** | | |
| | | | Company A | 0.09739 |
| | | | Company B | 0.33307 |
| | | | Company C | 0.56954 |
| 3 | Benefits | **0.197997** | | |
| | | | Company A | 0.24264 |
| | | | Company B | 0.08795 |
| | | | Company C | 0.66942 |
| 4 | Colleagues | **0.048310** | | |
| | | | Company A | 0.27905 |
| | | | Company B | 0.64912 |
| | | | Company C | 0.07193 |
| 5 | Location | **0.150245** | | |
| | | | Company A | 0.46667 |
| | | | Company B | 0.46667 |
| | | | Company C | 0.06667 |
| 6 | Reputation | **0.255792** | | |
| | | | Company A | 0.79276 |
| | | | Company B | 0.13122 |
| | | | Company C | 0.07602 |

- The Global Weight of an alternative is equal to criterion-weighted sum of its local weights.

- In Matrix notations:

$$\begin{bmatrix} w_A \\ w_B \\ w_C \end{bmatrix} = \begin{bmatrix} 0.13650 & 0.09739 & 0.24264 & 0.27895 & 0.46667 & 0.79276 \\ 0.62501 & 0.33307 & 0.08795 & 0.64912 & 0.46667 & 0.13122 \\ 0.23849 & 0.56954 & 0.66942 & 0.07193 & 0.06667 & 0.07602 \end{bmatrix} \begin{bmatrix} 0.158408 \\ 0.189247 \\ 0.197997 \\ 0.048310 \\ 0.150245 \\ 0.255792 \end{bmatrix} = \begin{bmatrix} 0.3745 \\ 0.3145 \\ 0.3110 \end{bmatrix}$$

- The Global weights of the three alternatives are:

| Alternative | Global weight |
|---|---|
| Company A | 0.3745 |
| Company B | 0.3145 |
| Company C | 0.3110 |

## Final Decision

- Choose Company A which has the highest global weight of 0.3745.
- Notice that Company B and C are close behind.

**Algorithm to evaluate a 3-Level AHP Model**

# 3-Level AHP Model

Given a 3-Level AHP Model:

- Goal $G$.
- $\boldsymbol{C} = \{c_1, c_2 \ldots, c_n\}$ = set of criteria.
- $\boldsymbol{A} = \{A_1, A_2, \ldots, A_p\}$ = set of alternatives.

Let $pwc(\boldsymbol{X}, y)$ denotes the normalized vector of weights obtained by pairwise compariing the elements of $\boldsymbol{X}$ w.r.t. element $y$.

Evaluate 3-Level AHP Model

**procedure** 3L-AHP$(G, \boldsymbol{C}, \boldsymbol{A})$
  $\boldsymbol{u} \leftarrow pwc(\boldsymbol{C}, G)$ // criteria's global weights
  **for** $i \leftarrow 1, n$ **do**
    $\boldsymbol{w}_i \leftarrow pwc(\boldsymbol{A}, c_i)$ // alternatives' local weights
  **end for**
  $\boldsymbol{W} \leftarrow [\boldsymbol{w}_1 \boldsymbol{w}_2 \ldots \boldsymbol{w}_n]$ // $(p \times n)$ matrix
  $\boldsymbol{w}^G \leftarrow \boldsymbol{W}\boldsymbol{u}$
  **return** $\boldsymbol{w}^G$ // alternatives' global weights
**end procedure**

### 3.2 Sensitivity Analysis

- Sensitivity Analysis can be performed to determine the impact on the global weights of the alternatives and hence their rankings if the weight of a criterion (i.e. its priority) is changed while keeping the weights of all the other criteria in the same relative proportion to their respective base values.

**Job Selection Problem:**

- Suppose we denote the weights of the 6 criteria by $w_1, w_2, \ldots, w_6$, respectively.

- The base values of the weights are:

$w_1 = 0.158408$
$w_2 = 0.189247$
$w_3 = 0.197997$
$w_4 = 0.048310$
$w_5 = 0.150245$
$w_6 = 0.255792$

- The current value of $w_1$ is 0.158408. Suppose we wish to investigate the impact of $w_1$ by allowing it to vary from $p = 0$ to 1, then the other five criterion weights are adjusted as all the weights must always add up to one.

- The weights are computed as follows:

Let $S = \sum_{i=2}^{6} w_i$ = Sum( 0.189247, 0.197997, 0.048310, 0.150245, 0.255792 ) = 0.841592

$w'_1 = p$
$w'_2 = (1-p)\,0.189247 / S = 0.22487\,(1-p)$
$w'_3 = (1-p)\,0.197997 / S = 0.23527\,(1-p)$
$w'_4 = (1-p)\,0.048310 / S = 0.05740\,(1-p)$
$w'_5 = (1-p)\,0.150245 / S = 0.17852\,(1-p)$
$w'_6 = (1-p)\,0.255792 / S = 0.30394\,(1-p)$

- The alternatives' global weights are then computed using the adjusted criteria weights:

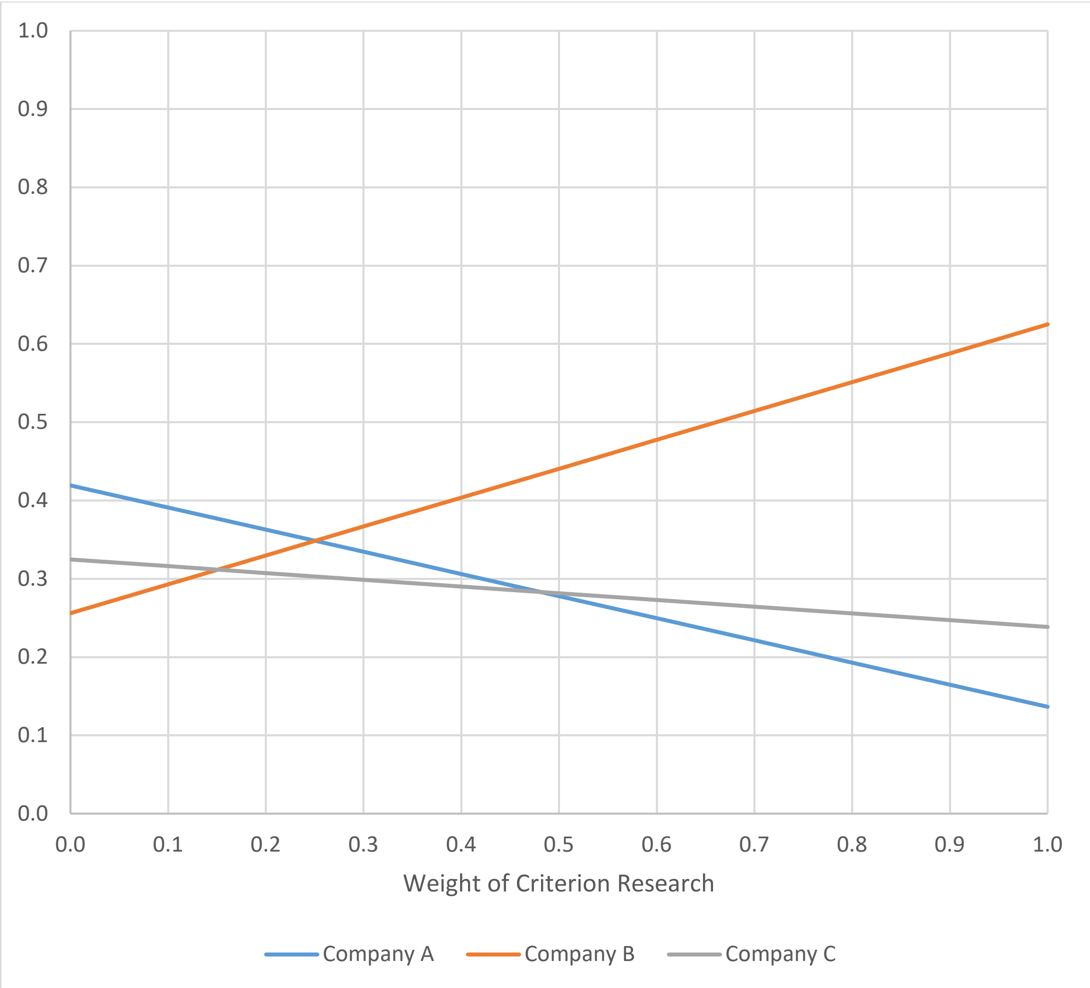
$$\begin{bmatrix} w_A \\ w_B \\ w_C \end{bmatrix} = \begin{bmatrix} 0.1365 & 0.0974 & 0.2426 & 0.2790 & 0.4667 & 0.7928 \\ 0.6250 & 0.3331 & 0.0880 & 0.6491 & 0.4667 & 0.1312 \\ 0.2385 & 0.5695 & 0.6694 & 0.0719 & 0.0666 & 0.0760 \end{bmatrix} \begin{bmatrix} w'_1 \\ w'_2 \\ w'_3 \\ w'_4 \\ w'_5 \\ w'_6 \end{bmatrix}$$

- The results of varying the value of each of the criterion weight one-at-a-time from 0 to 1 (step 0.1) are given below.

**Impact of changing weight of criterion Research**

| Weight for Criterion Research | Global Weight of Alternative | | |
|---|---|---|---|
| | Company A | Company B | Company C |
| 0.0 | **0.4193** | 0.2561 | 0.3247 |
| 0.1 | **0.3910** | 0.2930 | 0.3160 |
| 0.2 | **0.3627** | 0.3298 | 0.3074 |
| 0.3 | 0.3344 | **0.3667** | 0.2988 |
| 0.4 | 0.3062 | **0.4036** | 0.2902 |
| 0.5 | 0.2779 | **0.4405** | 0.2816 |
| 0.6 | 0.2496 | **0.4774** | 0.2730 |
| 0.7 | 0.2213 | **0.5143** | 0.2643 |
| 0.8 | 0.1931 | **0.5512** | 0.2557 |
| 0.9 | 0.1648 | **0.5881** | 0.2471 |
| 1.0 | 0.1365 | **0.625** | 0.2385 |

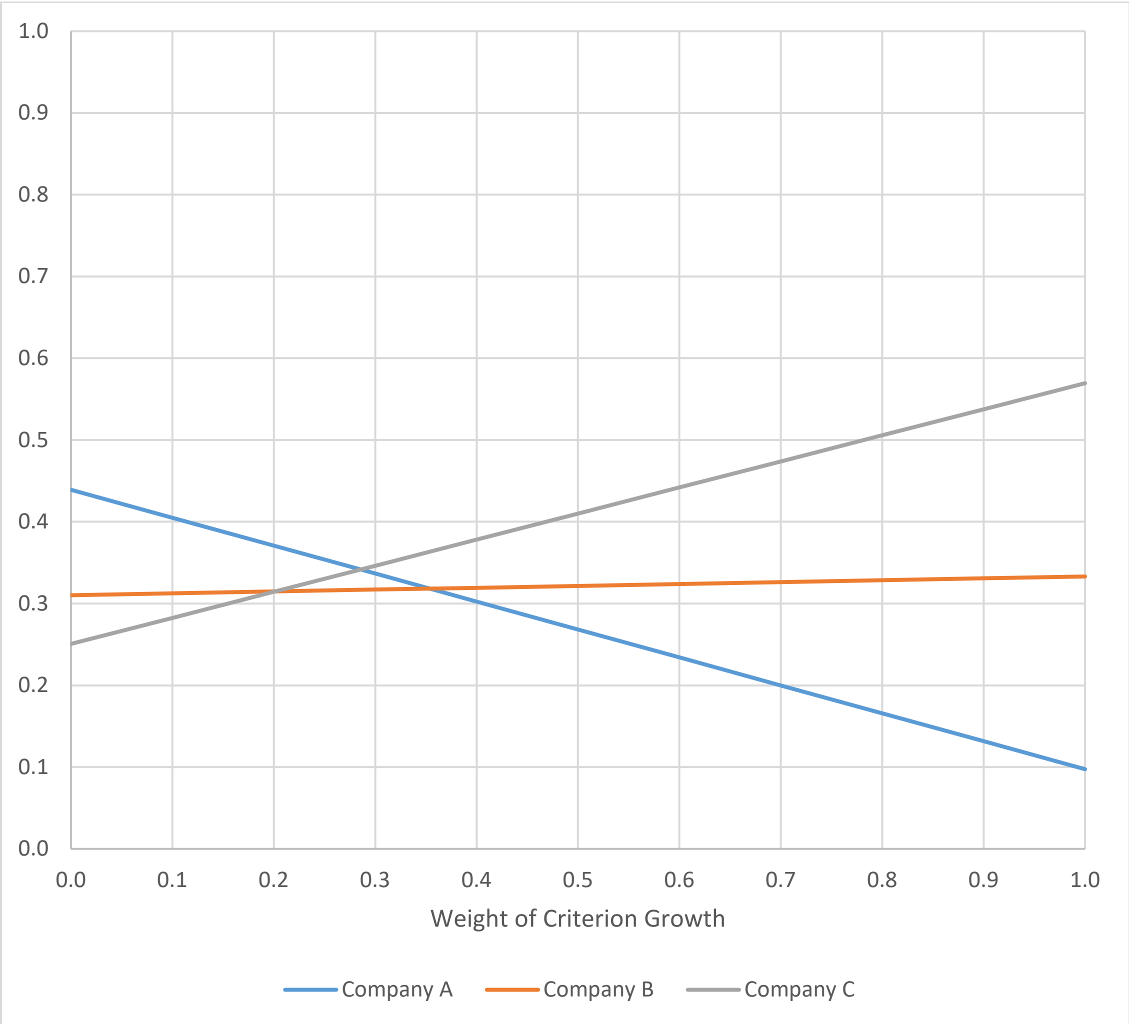
**Rainbow Diagram on the Impact of Changing weight of Criterion Research**



**Impact of changing weight of criterion Growth**

| Weight for Criterion Growth | Global Weight of Alternative | | |
|---|---|---|---|
| | Company A | Company B | Company C |
| 0.0 | **0.4392** | 0.3102 | 0.2507 |
| 0.1 | **0.4050** | 0.3125 | 0.2826 |
| 0.2 | **0.3708** | 0.3148 | 0.3144 |
| 0.3 | 0.3366 | 0.3170 | **0.3463** |
| 0.4 | 0.3025 | 0.3193 | **0.3782** |
| 0.5 | 0.2683 | 0.3216 | **0.4101** |
| 0.6 | 0.2341 | 0.3239 | **0.4420** |
| 0.7 | 0.1999 | 0.3262 | **0.4739** |
| 0.8 | 0.1658 | 0.3285 | **0.5057** |
| 0.9 | 0.1316 | 0.3308 | **0.5376** |
| 1.0 | 0.0974 | 0.3331 | **0.5695** |

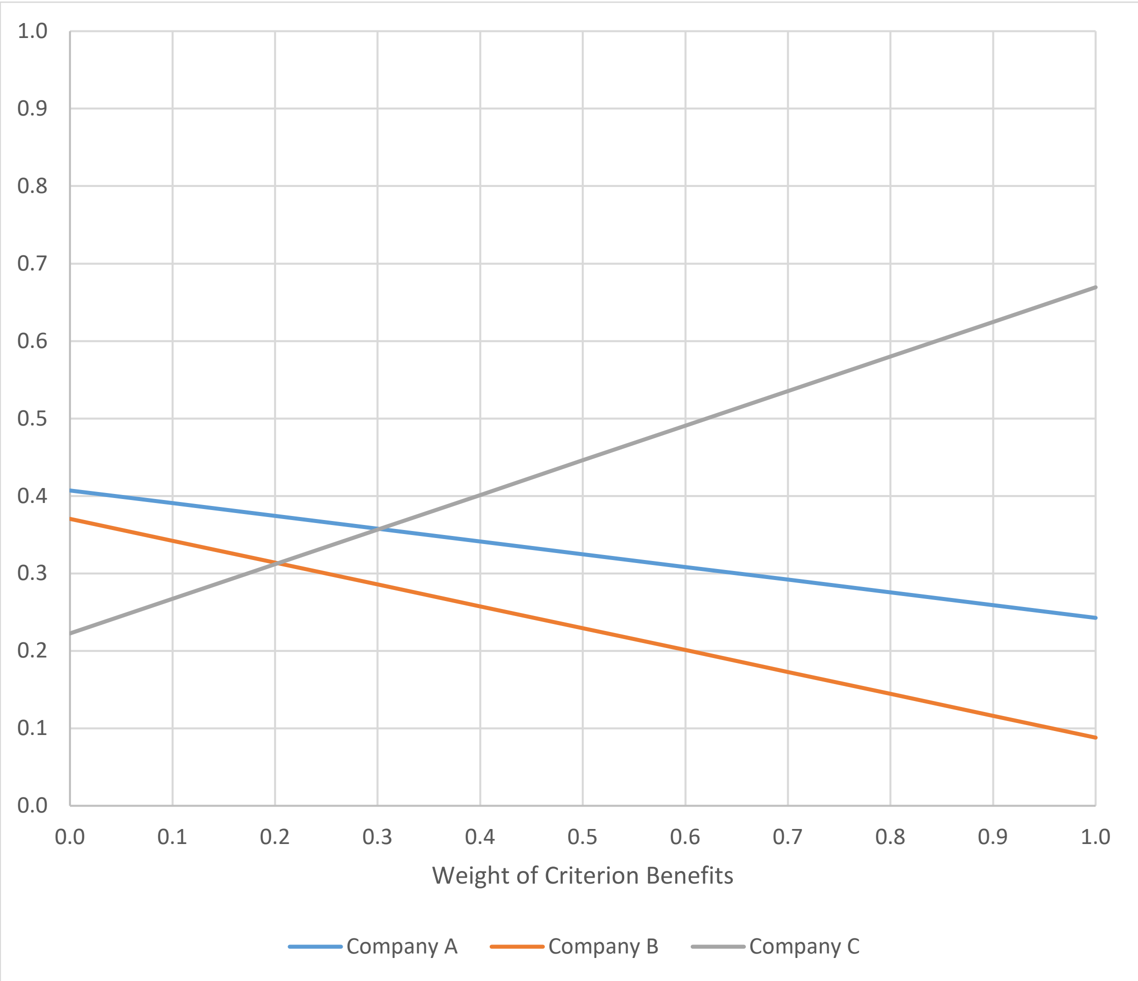
**Rainbow Diagram on the Impact of Changing weight of Criterion Growth**



### Impact of changing weight of criterion Benefits

| Weight for Criterion Benefits | Global Weight of Alternative | | |
|---|---|---|---|
| | Company A | Company B | Company C |
| 0.0 | **0.4070** | 0.3704 | 0.2225 |
| 0.1 | **0.3906** | 0.3422 | 0.2672 |
| 0.2 | **0.3742** | 0.3139 | 0.3119 |
| 0.3 | **0.3577** | 0.2857 | 0.3566 |
| 0.4 | 0.3413 | 0.2575 | **0.4013** |
| 0.5 | 0.3248 | 0.2292 | **0.4460** |
| 0.6 | 0.3084 | 0.2010 | **0.4907** |
| 0.7 | 0.2919 | 0.1727 | **0.5353** |
| 0.8 | 0.2755 | 0.1445 | **0.5800** |
| 0.9 | 0.2590 | 0.1162 | **0.6247** |
| 1.0 | 0.2426 | 0.0880 | **0.6694** |

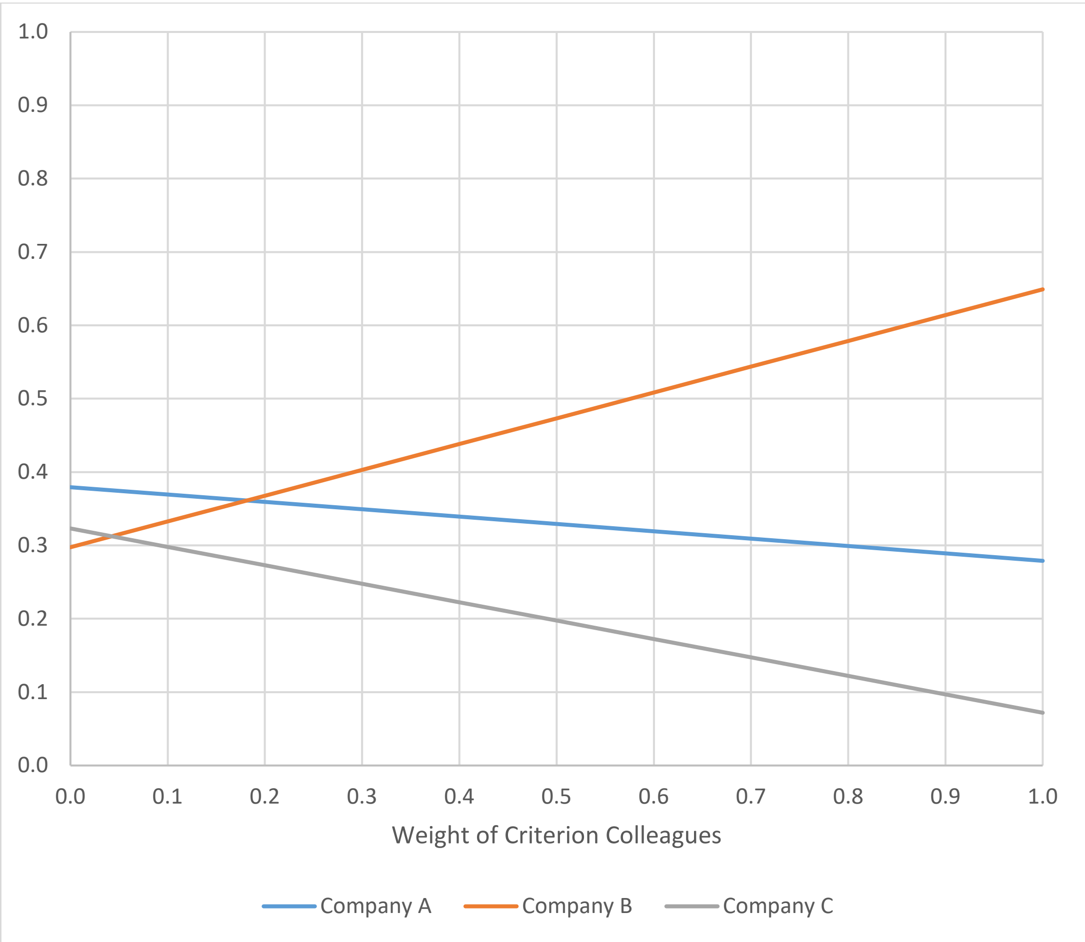
### Rainbow Diagram on the Impact of Changing weight of Criterion Benefits



### Impact of changing weight of criterion Colleagues

| Weight for Criterion Colleagues | Global Weight of Alternative | | |
|---|---|---|---|
| | Company A | Company B | Company C |
| 0.0 | **0.3793** | 0.2975 | 0.3232 |
| 0.1 | **0.3693** | 0.3327 | 0.2980 |
| 0.2 | 0.3593 | **0.3678** | 0.2729 |
| 0.3 | 0.3492 | **0.4030** | 0.2478 |
| 0.4 | 0.3392 | **0.4382** | 0.2227 |
| 0.5 | 0.3292 | **0.4733** | 0.1975 |
| 0.6 | 0.3191 | **0.5085** | 0.1724 |
| 0.7 | 0.3091 | **0.5436** | 0.1473 |
| 0.8 | 0.2991 | **0.5788** | 0.1222 |
| 0.9 | 0.2890 | **0.6139** | 0.0970 |
| 1.0 | 0.2790 | **0.6491** | 0.0719 |

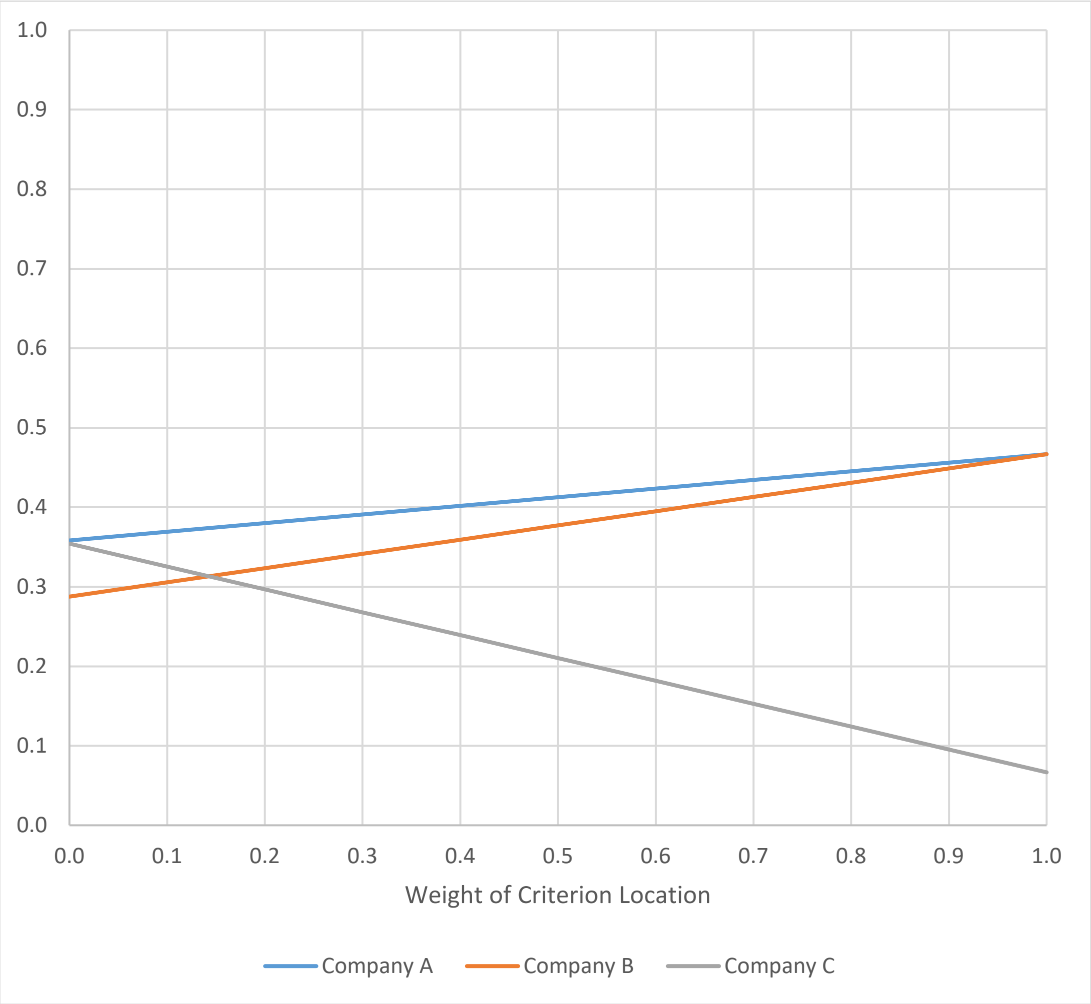
### Rainbow Diagram on the Impact of Changing weight of Criterion Colleagues



**Impact of changing weight of criterion Location**

| Weight for Criterion Location | Global Weight of Alternative | | |
|---|---|---|---|
| | Company A | Company B | Company C |
| 0.0 | **0.3582** | 0.2876 | 0.3542 |
| 0.1 | **0.3690** | 0.3055 | 0.3255 |
| 0.2 | **0.3799** | 0.3234 | 0.2967 |
| 0.3 | **0.3907** | 0.3413 | 0.2679 |
| 0.4 | **0.4016** | 0.3592 | 0.2392 |
| 0.5 | **0.4124** | 0.3771 | 0.2104 |
| 0.6 | **0.4233** | 0.3951 | 0.1817 |
| 0.7 | **0.4341** | 0.4130 | 0.1529 |
| 0.8 | **0.4450** | 0.4309 | 0.1241 |
| 0.9 | **0.4558** | 0.4488 | 0.0954 |
| 1.0 | **0.4667** | 0.4667 | 0.0666 |

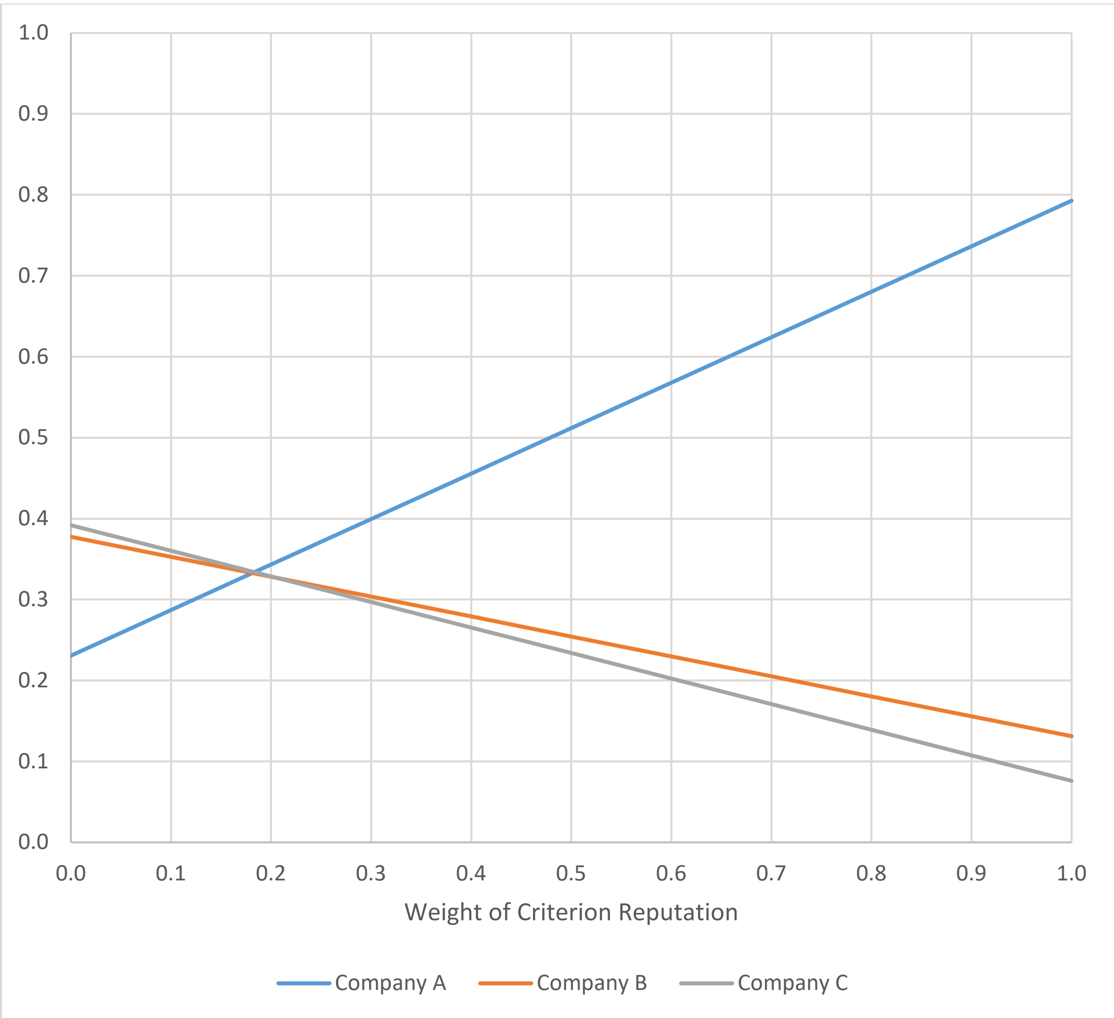
**Rainbow Diagram on the Impact of Changing weight of Criterion Location**



**Impact of changing weight of criterion Reputation**

| Weight for Criterion Reputation | Global Weight of Alternative | | |
|---|---|---|---|
| | Company A | Company B | Company C |
| 0.0 | 0.2307 | 0.3775 | **0.3918** |
| 0.1 | 0.2869 | 0.3529 | **0.3602** |
| 0.2 | 0.3431 | 0.3282 | **0.3286** |
| 0.3 | **0.3993** | 0.3036 | 0.2971 |
| 0.4 | **0.4555** | 0.2790 | 0.2655 |
| 0.5 | **0.5117** | 0.2544 | 0.2339 |
| 0.6 | **0.5680** | 0.2297 | 0.2023 |
| 0.7 | **0.6242** | 0.2051 | 0.1707 |
| 0.8 | **0.6804** | 0.1805 | 0.1392 |
| 0.9 | **0.7366** | 0.1558 | 0.1076 |
| 1.0 | **0.7928** | 0.1312 | 0.0760 |

**Rainbow Diagram on the Impact of Changing weight of Criterion Reputation**



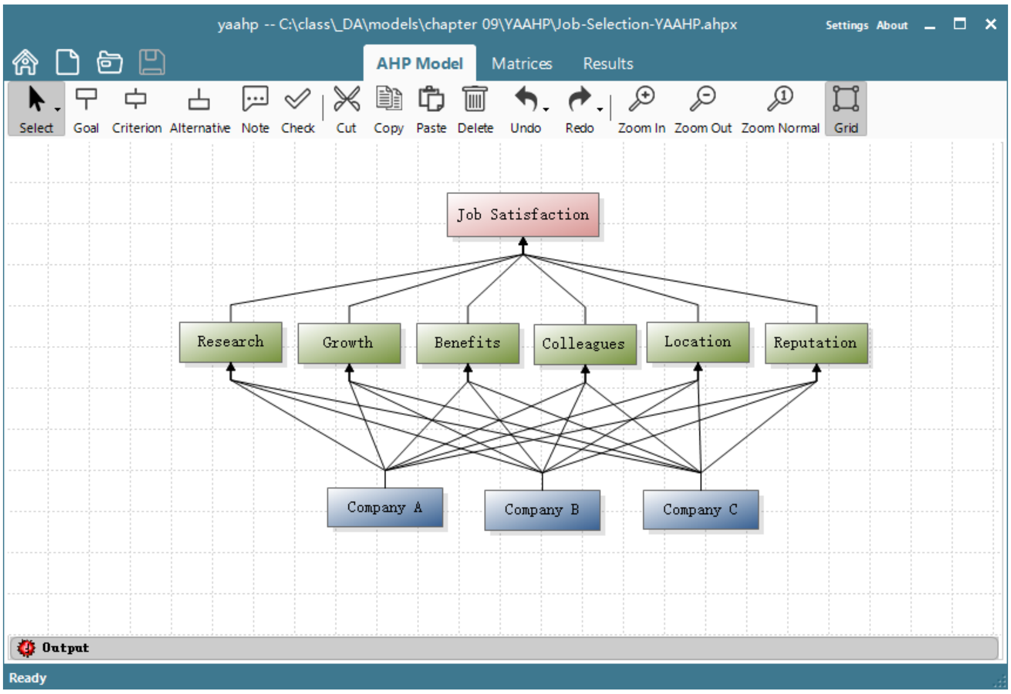
# 4 Using Computer Software to Solve AHP Models

## 4.1 YAAHP (Yet Another AHP)

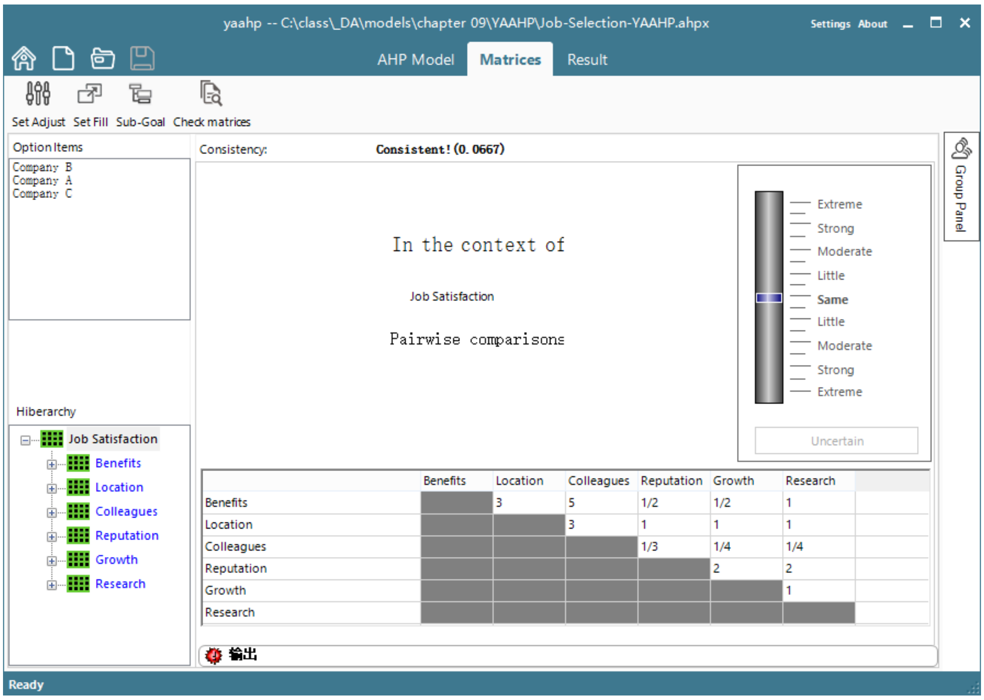
- YAAHP solves standard AHP models and Fuzzy Comprehensive Evaluation Models.
  - Chinese version: [http://www.metadecsn.com/download/](http://www.metadecsn.com/download/)

- Note: Trial versions from 11 onward do not support sensitivity analysis.

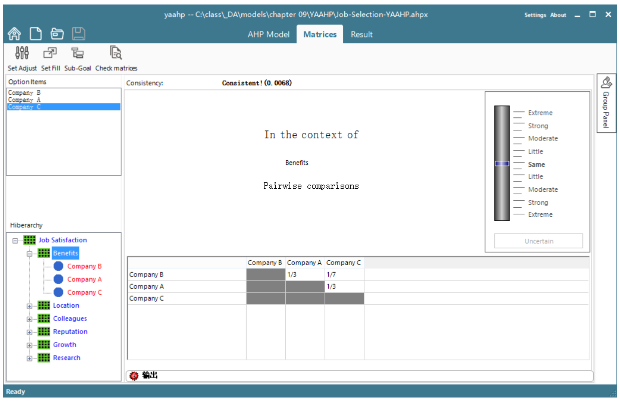
- **<u>Hierarchy Modeling Interface</u>**

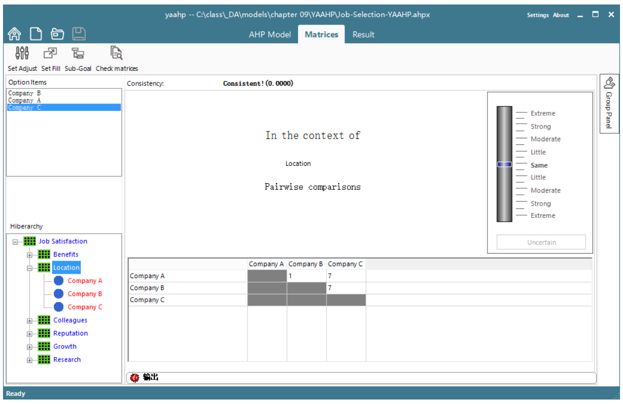


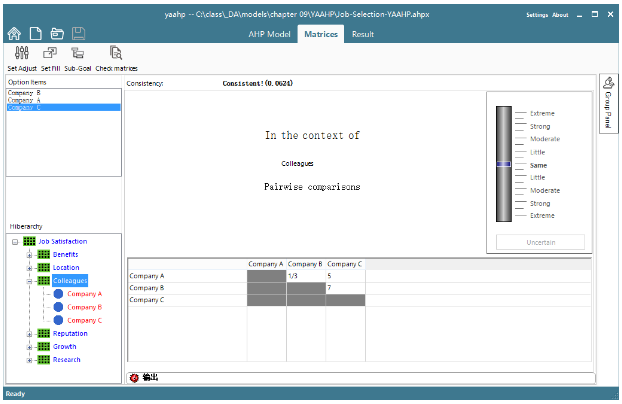

**<u>Pairwise Comprison of the Criteria</u>**

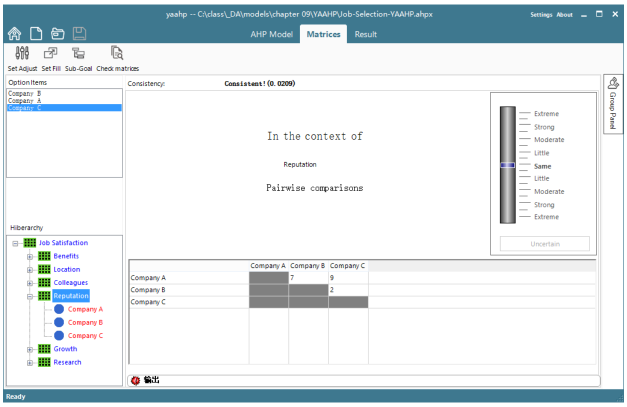


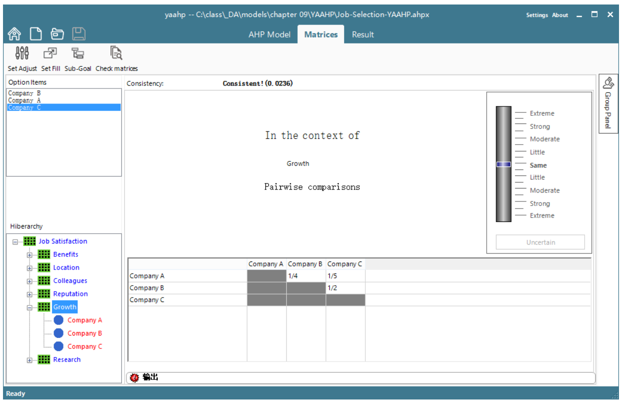
# Pairwise comprision of alternatives w.r.t. each criterion

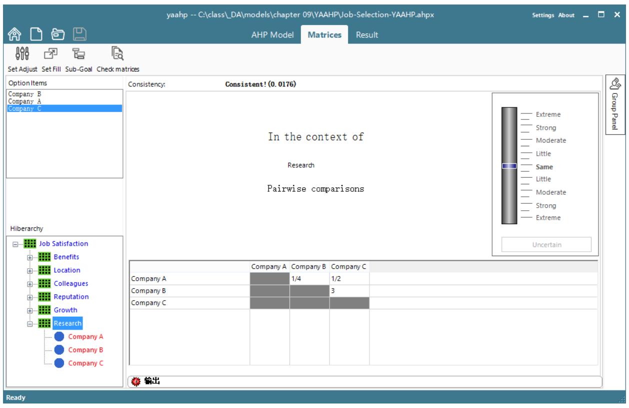


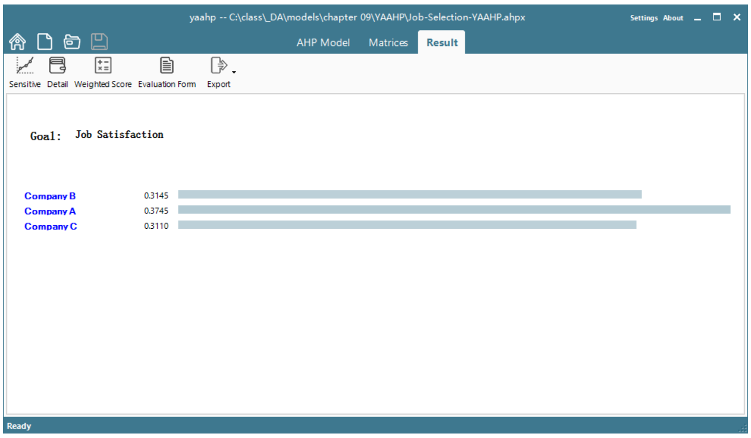



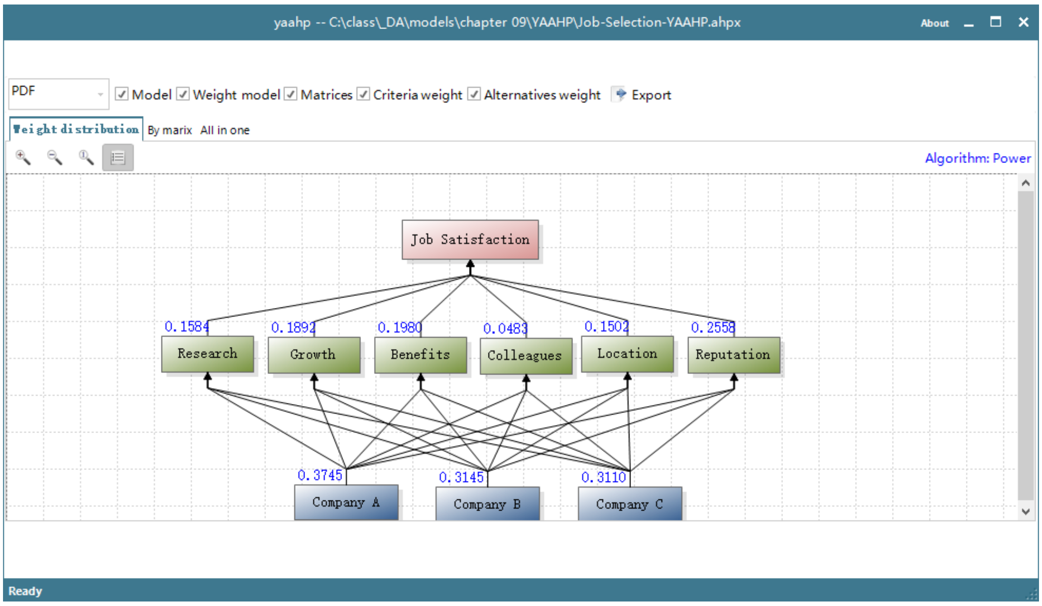




# Results



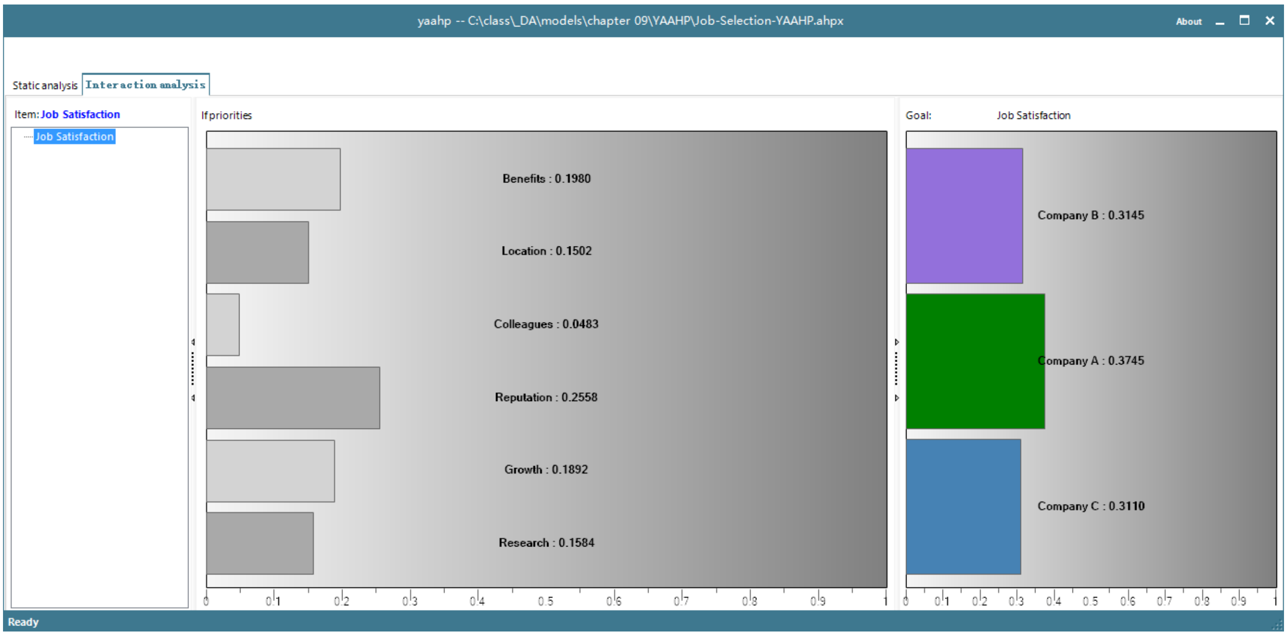
## Summary of Results




## Sensitivity Analysis with YAAHP

### Interactive Dynamic Sensitivity

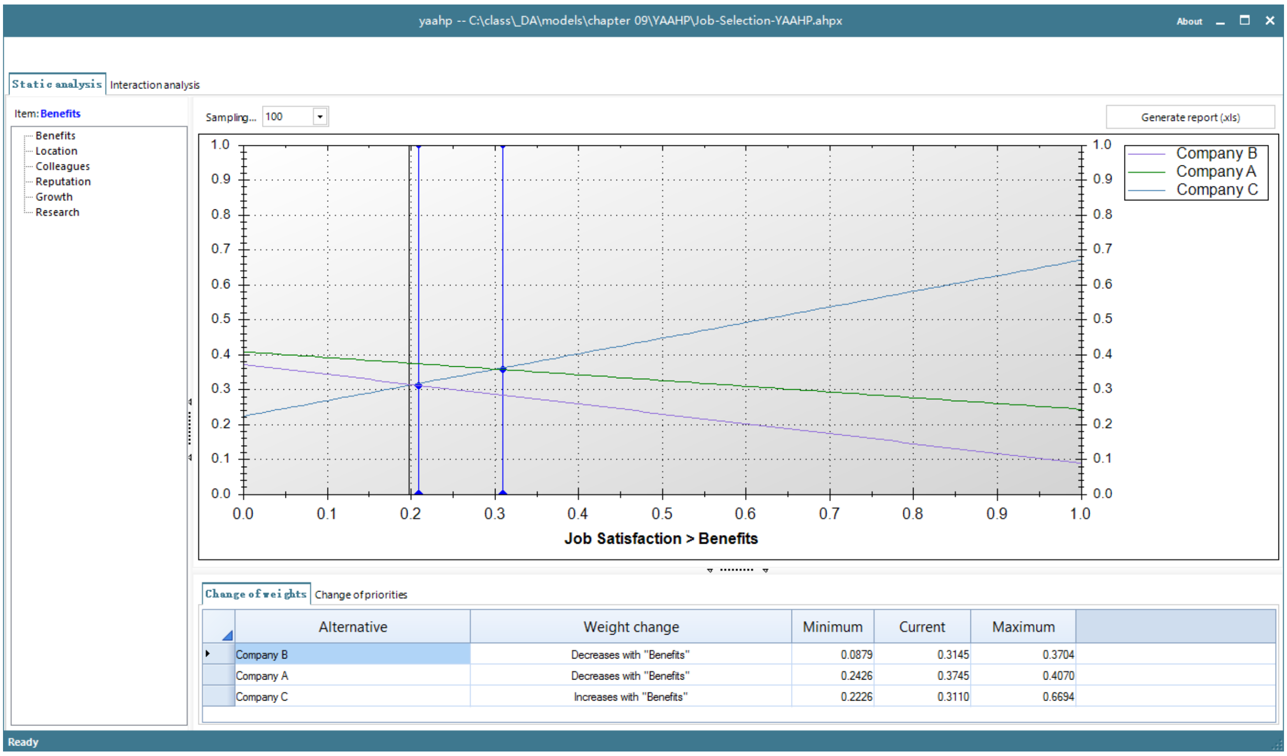
- Dynamic sensitivity enables us to see the change in global weight of the alternatives as we increase or decrease the priority weight of the criteria.



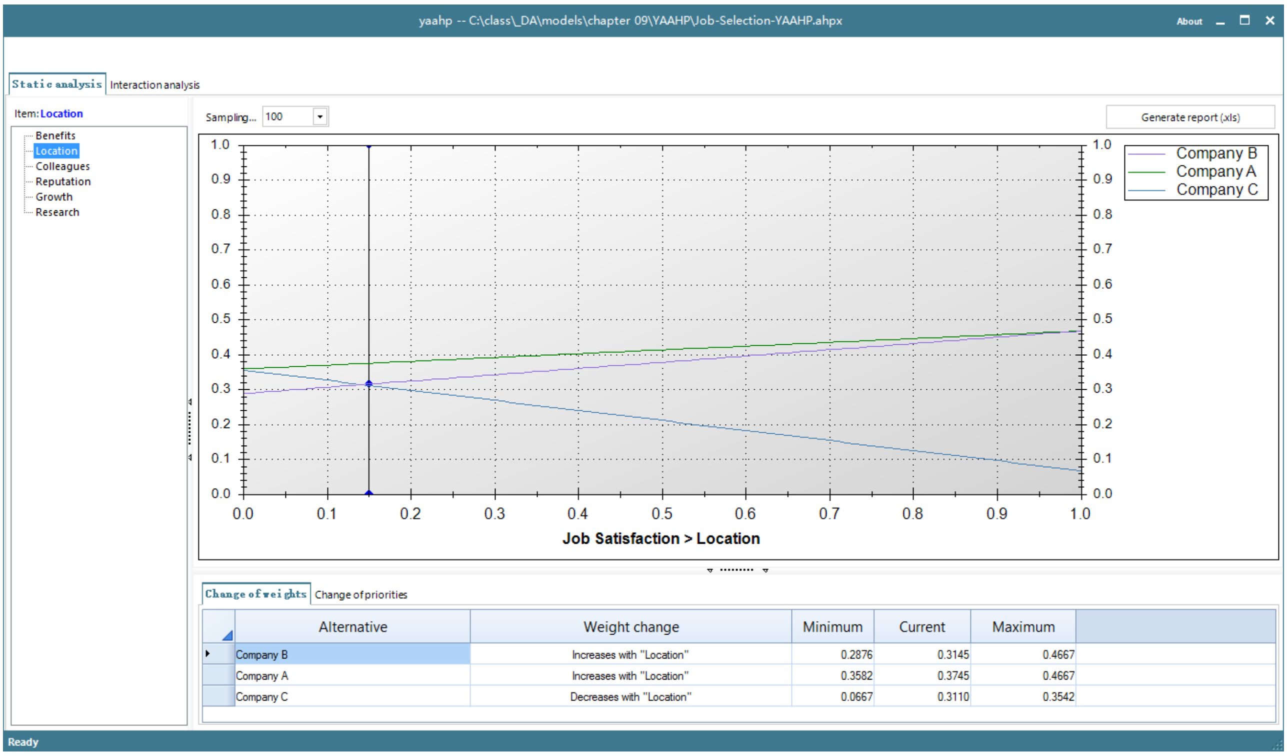
## Rainbow Diagrams on Changes in Criterion Weight

- These diagrams how the global weights of the alternatives change as the weight of a criterion changes from 0 to 1.0, while keeping the weights of the other criteria in the same relative proportion as their base values.

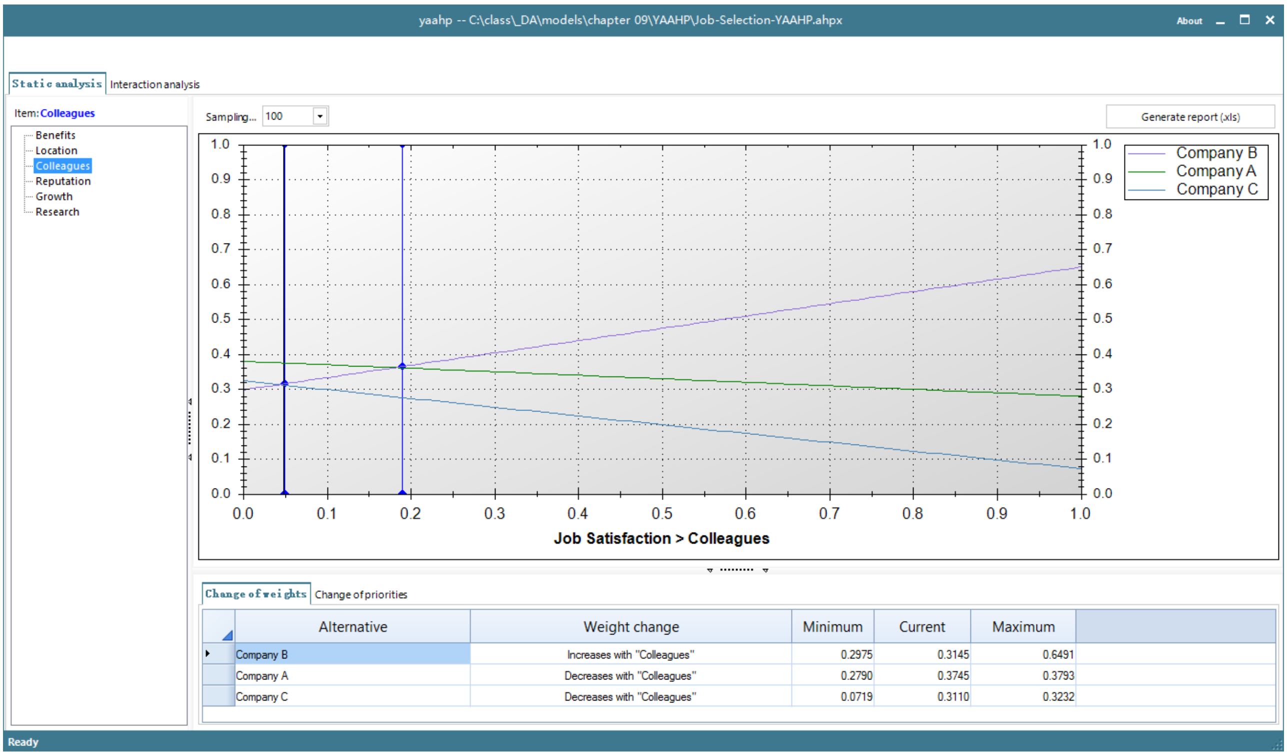
### Rainbow Diagram for alternative global weights when Benefit weight is varied from 0 to 1



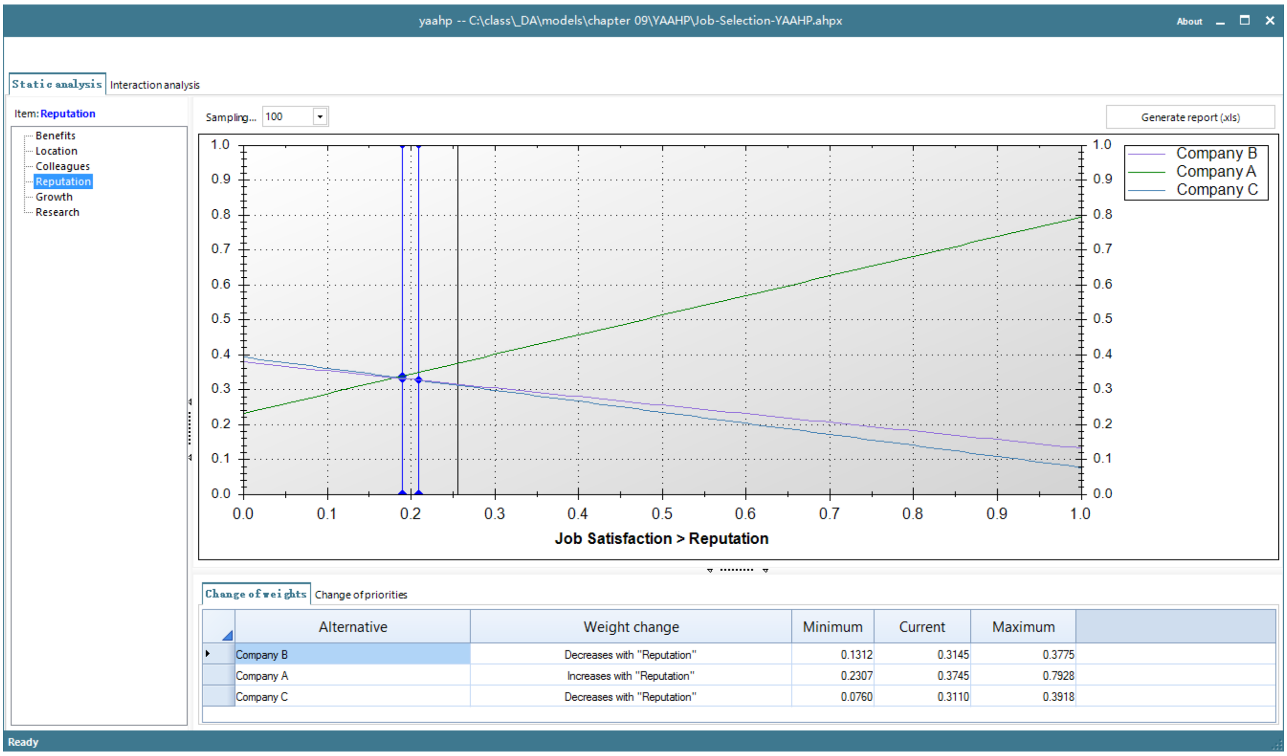

### Rainbow Diagram or alternative global weights when Location weight is varied from 0 to 1



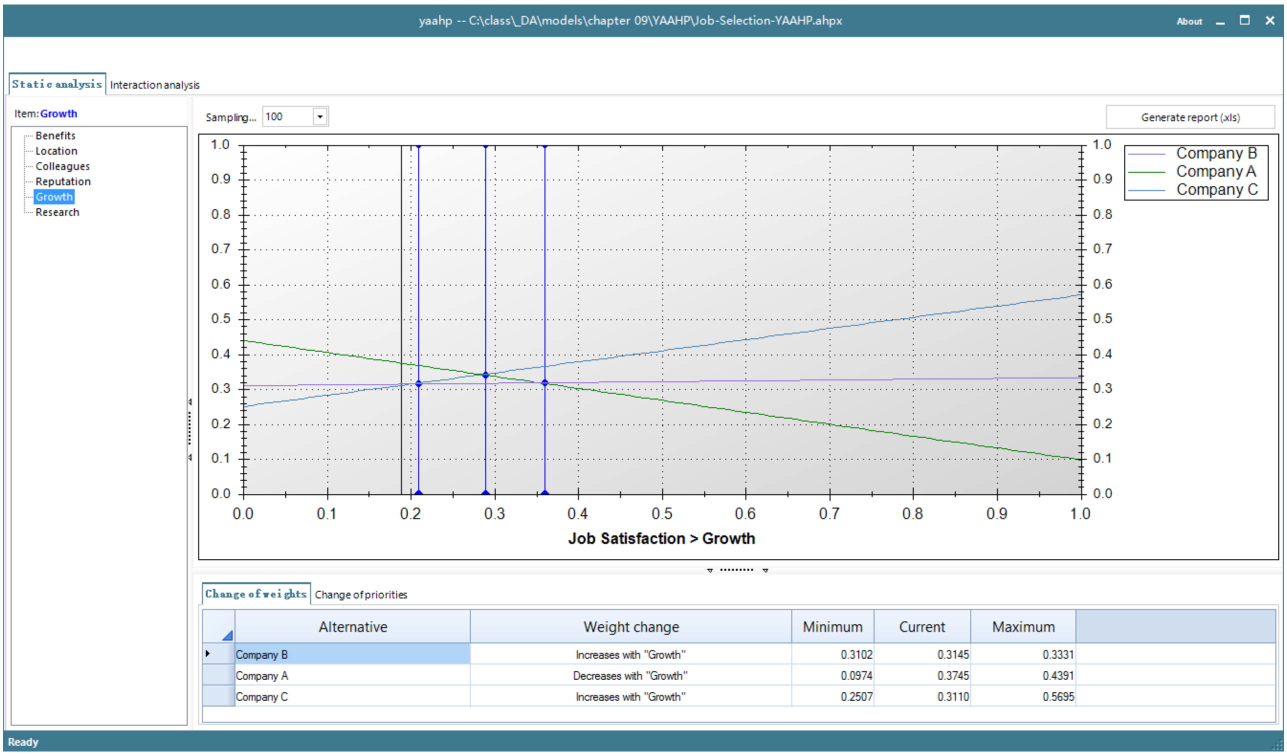
## Rainbow Diagram for alternative global weights when Colleagues weight is varied from 0 to 1




## Rainbow Diagram for alternative global weights when Reputation wt is varied from 0 to 1

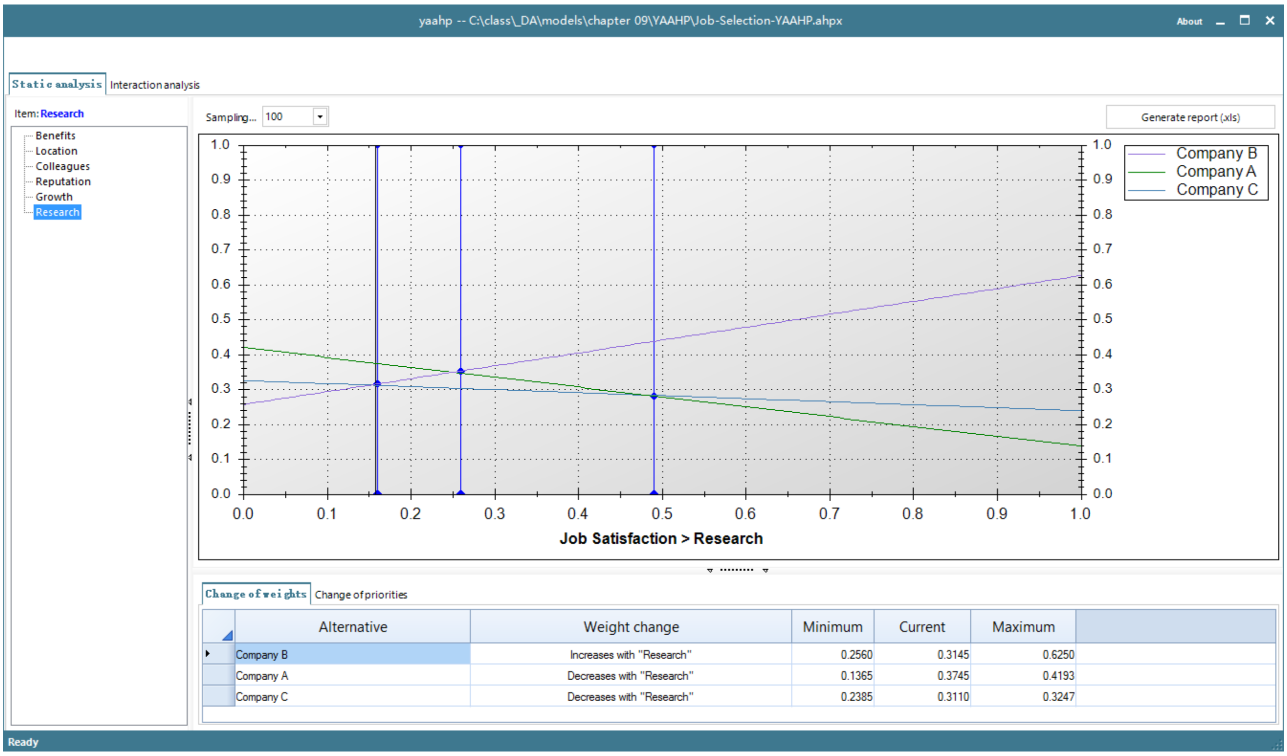


## Rainbow Diagram for or alternative global weights when Growth weight is varied from 0 to 1




| Alternative | Weight change | Minimum | Current | Maximum |
|---|---|---|---|---|
| Company B | Increases with "Growth" | 0.3102 | 0.3145 | 0.3331 |
| Company A | Decreases with "Growth" | 0.0974 | 0.3745 | 0.4391 |
| Company C | Increases with "Growth" | 0.2507 | 0.3110 | 0.5695 |

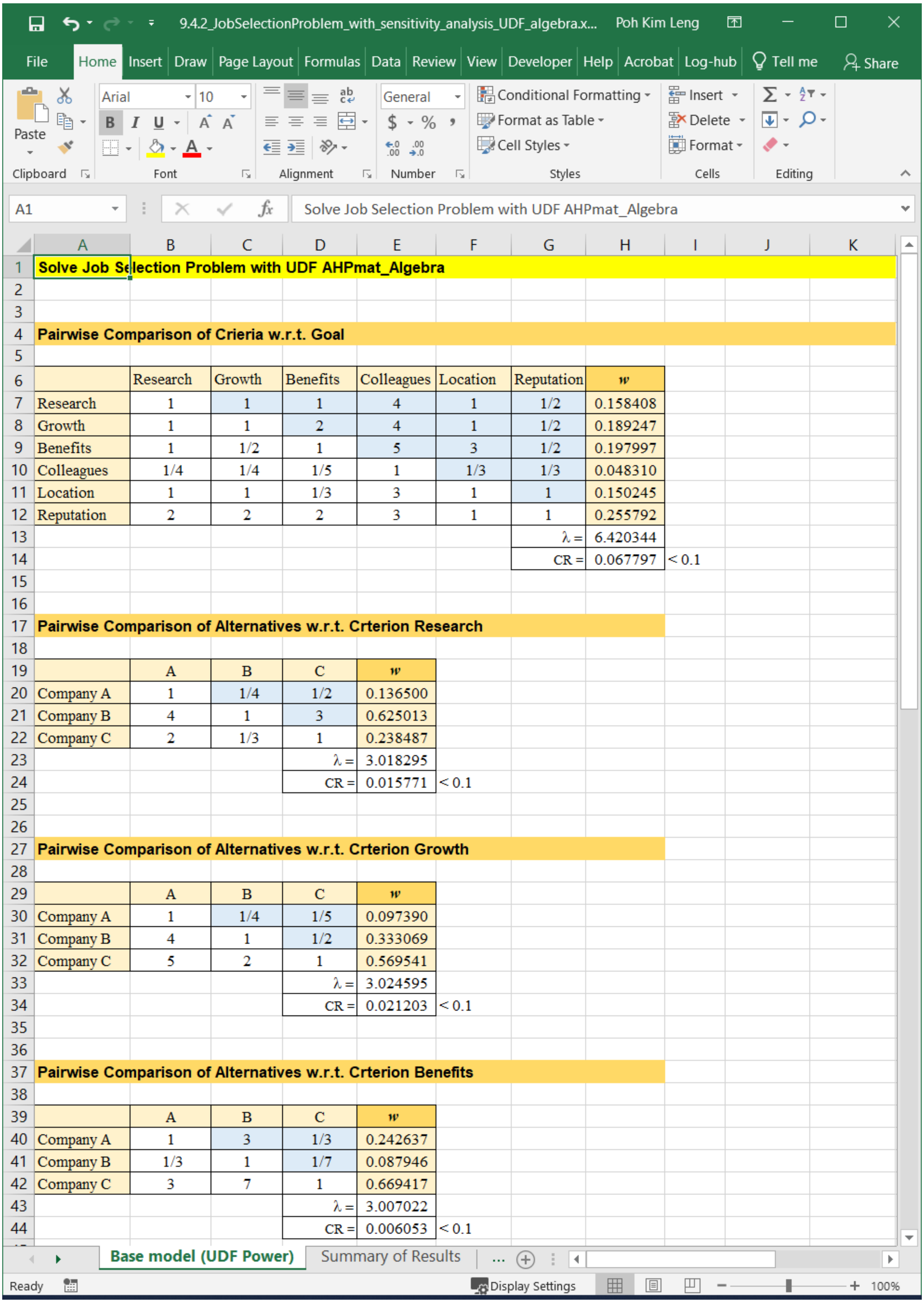
## Rainbow Diagram for or alternative global weights hen Research weight is varied from 0 to 1




| Alternative | Weight change | Minimum | Current | Maximum |
|---|---|---|---|---|
| Company B | Increases with "Research" | 0.2560 | 0.3145 | 0.6250 |
| Company A | Decreases with "Research" | 0.1365 | 0.3745 | 0.4193 |
| Company C | Decreases with "Research" | 0.2385 | 0.3110 | 0.3247 |

## 4.2 Using Excel with User-Defined Functions

- You can use Excel UDFs to compute the weights, λ, CI and CR for each matrices, and then compute the global weights using worksheet functions. Sensitivity analysis can also be performed.




**Solve Job Selection Problem with UDF AHPmat_Algebra**

**Pairwise Comparison of Crieria w.r.t. Goal**

| | Research | Growth | Benefits | Colleagues | Location | Reputation | *w* | |
|---|---|---|---|---|---|---|---|---|
| Research | 1 | 1 | 1 | 4 | 1 | 1/2 | 0.158408 | |
| Growth | 1 | 1 | 2 | 4 | 1 | 1/2 | 0.189247 | |
| Benefits | 1 | 1/2 | 1 | 5 | 3 | 1/2 | 0.197997 | |
| Colleagues | 1/4 | 1/4 | 1/5 | 1 | 1/3 | 1/3 | 0.048310 | |
| Location | 1 | 1 | 1/3 | 3 | 1 | 1 | 0.150245 | |
| Reputation | 2 | 2 | 2 | 3 | 1 | 1 | 0.255792 | |
| | | | | | | λ = | 6.420344 | |
| | | | | | | CR = | 0.067797 | < 0.1 |

**Pairwise Comparison of Alternatives w.r.t. Crterion Research**

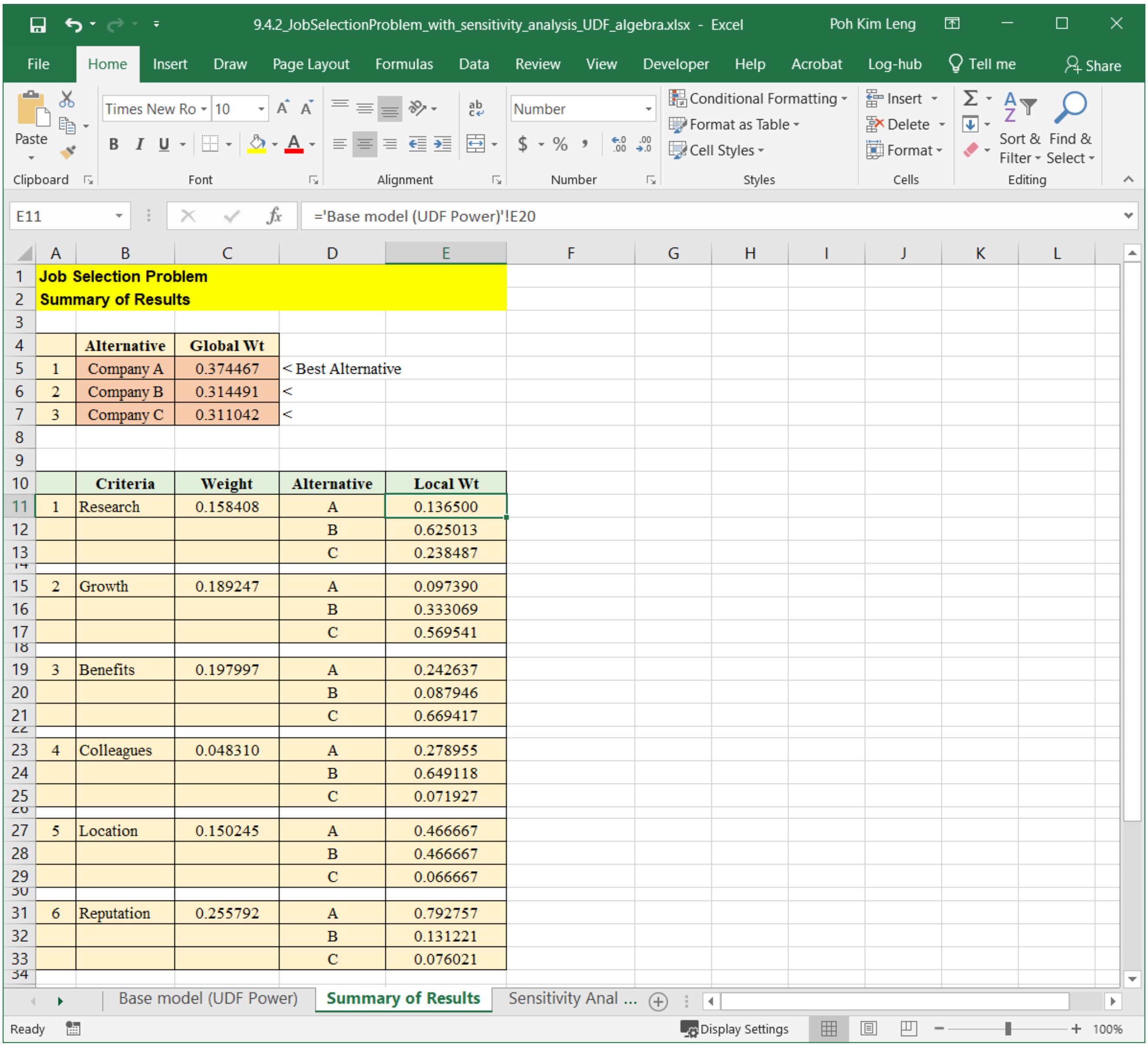
| | A | B | C | *w* | |
|---|---|---|---|---|---|
| Company A | 1 | 1/4 | 1/2 | 0.136500 | |
| Company B | 4 | 1 | 3 | 0.625013 | |
| Company C | 2 | 1/3 | 1 | 0.238487 | |
| | | | λ = | 3.018295 | |
| | | | CR = | 0.015771 | < 0.1 |

**Pairwise Comparison of Alternatives w.r.t. Crterion Growth**

| | A | B | C | *w* | |
|---|---|---|---|---|---|
| Company A | 1 | 1/4 | 1/5 | 0.097390 | |
| Company B | 4 | 1 | 1/2 | 0.333069 | |
| Company C | 5 | 2 | 1 | 0.569541 | |
| | | | λ = | 3.024595 | |
| | | | CR = | 0.021203 | < 0.1 |

**Pairwise Comparison of Alternatives w.r.t. Crterion Benefits**

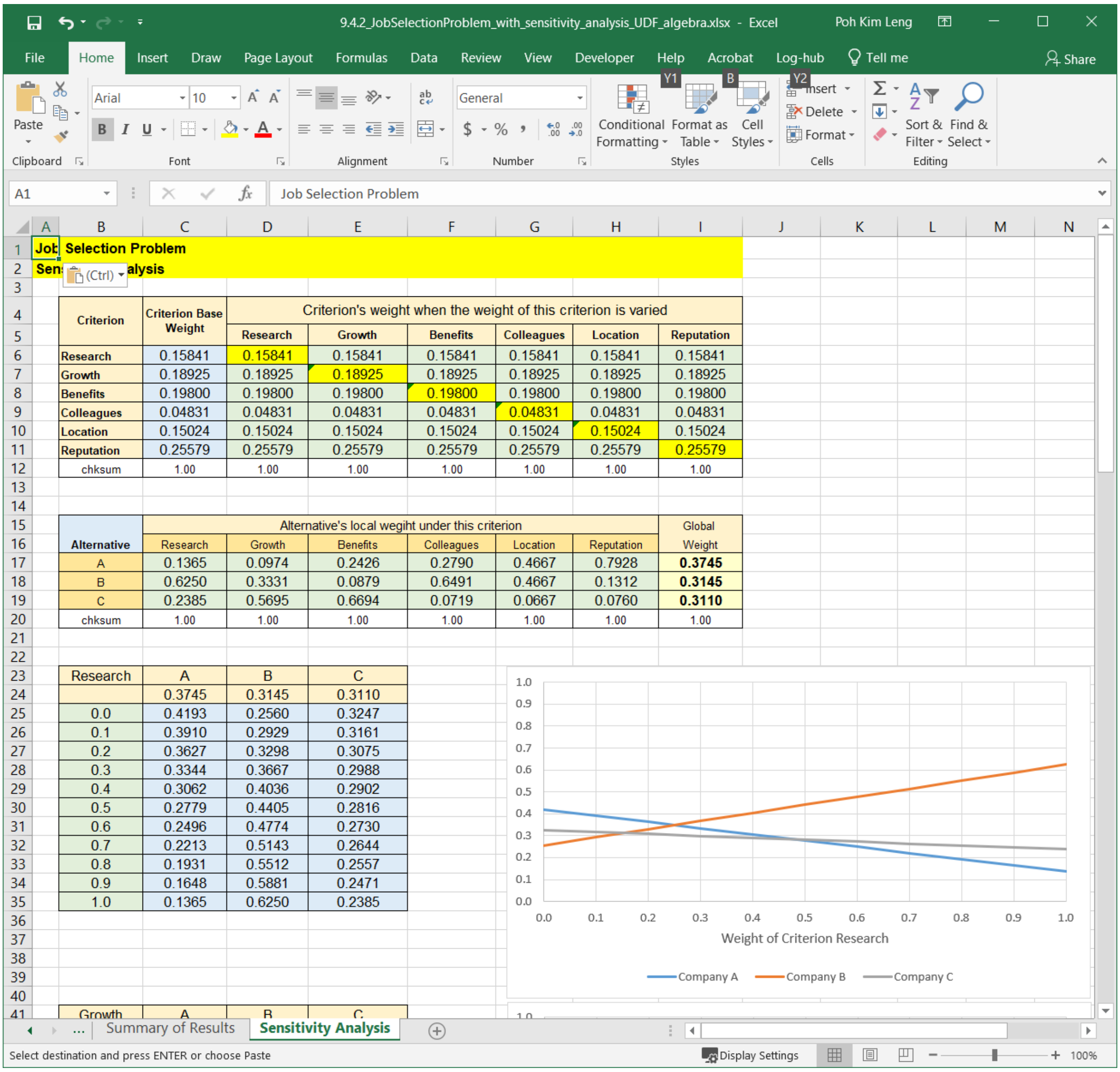
| | A | B | C | *w* | |
|---|---|---|---|---|---|
| Company A | 1 | 3 | 1/3 | 0.242637 | |
| Company B | 1/3 | 1 | 1/7 | 0.087946 | |
| Company C | 3 | 7 | 1 | 0.669417 | |
| | | | λ = | 3.007022 | |
| | | | CR = | 0.006053 | < 0.1 |




**Job Selection Problem**
**Summary of Results**

| | Alternative | Global Wt | |
|---|---|---|---|
| 1 | Company A | 0.374467 | < Best Alternative |
| 2 | Company B | 0.314491 | < |
| 3 | Company C | 0.311042 | < |

| | Criteria | Weight | Alternative | Local Wt |
|---|---|---|---|---|
| 1 | Research | 0.158408 | A | 0.136500 |
| | | | B | 0.625013 |
| | | | C | 0.238487 |
| 2 | Growth | 0.189247 | A | 0.097390 |
| | | | B | 0.333069 |
| | | | C | 0.569541 |
| 3 | Benefits | 0.197997 | A | 0.242637 |
| | | | B | 0.087946 |
| | | | C | 0.669417 |
| 4 | Colleagues | 0.048310 | A | 0.278955 |
| | | | B | 0.649118 |
| | | | C | 0.071927 |
| 5 | Location | 0.150245 | A | 0.466667 |
| | | | B | 0.466667 |
| | | | C | 0.066667 |
| 6 | Reputation | 0.255792 | A | 0.792757 |
| | | | B | 0.131221 |
| | | | C | 0.076021 |



Job Selection Problem

Sensitivity Analysis

| Criterion | Criterion Base Weight | Criterion's weight when the weight of this criterion is varied | | | | | |
|---|---|---|---|---|---|---|---|
| | | Research | Growth | Benefits | Colleagues | Location | Reputation |
| Research | 0.15841 | 0.15841 | 0.15841 | 0.15841 | 0.15841 | 0.15841 | 0.15841 |
| Growth | 0.18925 | 0.18925 | 0.18925 | 0.18925 | 0.18925 | 0.18925 | 0.18925 |
| Benefits | 0.19800 | 0.19800 | 0.19800 | 0.19800 | 0.19800 | 0.19800 | 0.19800 |
| Colleagues | 0.04831 | 0.04831 | 0.04831 | 0.04831 | 0.04831 | 0.04831 | 0.04831 |
| Location | 0.15024 | 0.15024 | 0.15024 | 0.15024 | 0.15024 | 0.15024 | 0.15024 |
| Reputation | 0.25579 | 0.25579 | 0.25579 | 0.25579 | 0.25579 | 0.25579 | 0.25579 |
| chksum | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |

| Alternative | Alternative's local wegiht under this criterion | | | | | | Global Weight |
|---|---|---|---|---|---|---|---|
| | Research | Growth | Benefits | Colleagues | Location | Reputation | |
| A | 0.1365 | 0.0974 | 0.2426 | 0.2790 | 0.4667 | 0.7928 | **0.3745** |
| B | 0.6250 | 0.3331 | 0.0879 | 0.6491 | 0.4667 | 0.1312 | **0.3145** |
| C | 0.2385 | 0.5695 | 0.6694 | 0.0719 | 0.0667 | 0.0760 | **0.3110** |
| chksum | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |

| Research | A | B | C |
|---|---|---|---|
| | 0.3745 | 0.3145 | 0.3110 |
| 0.0 | 0.4193 | 0.2560 | 0.3247 |
| 0.1 | 0.3910 | 0.2929 | 0.3161 |
| 0.2 | 0.3627 | 0.3298 | 0.3075 |
| 0.3 | 0.3344 | 0.3667 | 0.2988 |
| 0.4 | 0.3062 | 0.4036 | 0.2902 |
| 0.5 | 0.2779 | 0.4405 | 0.2816 |
| 0.6 | 0.2496 | 0.4774 | 0.2730 |
| 0.7 | 0.2213 | 0.5143 | 0.2644 |
| 0.8 | 0.1931 | 0.5512 | 0.2557 |
| 0.9 | 0.1648 | 0.5881 | 0.2471 |
| 1.0 | 0.1365 | 0.6250 | 0.2385 |

## 4.3 Using Python to Solve a 3-Level AHP Model

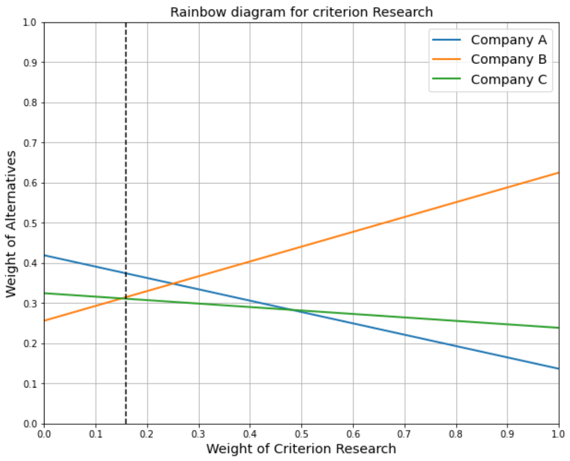
```
# -*- coding: utf-8 -*-
""" AHP_3L_with_sensit_analysis_JobSelectionProblem.py """
import numpy as np
from scipy import stats
from scipy.optimize import root
import matplotlib.pyplot as plt

def main():

    """ Solve the 3-Level AHP Job Selection Problem """

    G = 'Job Satisfaction'
    C = ['Research','Growth','Benefits','Colleages',
         'Location','Reputation']
    AL = ['Company A', 'Company B', 'Company C']

    # Pairwise compare criteria w.r.t Goal
    A0 = np.array([[ 1,   1 ,  1,  4,  1,  1/2 ],
                   [ 1,   1,   2,  4,  1,  1/2 ],
                   [ 1, 1/2 , 1,   5,  3,  1/2 ],
                   [1/4,1/4, 1/5, 1, 1/3, 1/3 ],
                   [ 1,   1, 1/3, 3,  1,   1  ],
                   [ 2,   2,  2,  3,  1,   1  ]] )

    # Pairwise compare alternatives wrt Research
    A1 = np.array([[ 1, 1/4, 1/2 ],
                   [ 4,  1,    3 ],
                   [ 2, 1/3,   1 ]])

    # Pairwise compare alternatives wrt Growth
    A2 =  np.array([[1, 1/4, 1/5 ],
                    [4,  1,  1/2 ],
                    [5,  2,   1  ]])

    # Pairwise compare alternatives wrt Benefits
    A3 = np.array([[ 1,  3,  1/3 ],
                   [1/3, 1,  1/7 ],
                   [ 3,  7,   1  ]])

    # Pairwise compare alternatives wrt Colleages
    A4 = np.array([[ 1,  1/3,  5 ],
                   [ 3,   1,   7 ],
                   [1/5, 1/7,  1 ]])

    # Pairwise compare alternatives wrt Location
    A5 = np.array([[ 1,   1,   7 ],
                   [ 1,   1,   7 ],
                   [1/7, 1/7,  1 ]])

    # Pairwise compare alternatives wrt Reputation
    A6 = np.array([[ 1,   7,   9 ],
                   [1/7,  1,   2 ],
                   [1/9, 1/2,  1 ]])


    # Compute Criteria weights
    method = 'Algebra'
    u = AHPmat(A0, method=method)
    print("Criteria's Weights")
    for i, cr in enumerate(C):
        print(f"  {cr:10}: {u[i]:.6f}")

    # Compute alternatives' weights wrt each criterion
    W = np.array([ AHPmat(A, method=method)
                  for A in [A1, A2, A3, A4, A5, A6] ]).T
    print("\nAlternatives' local Weights")
    print(W)
```

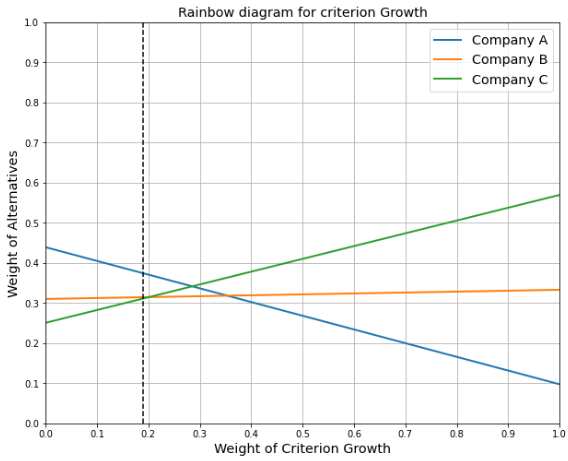
```python
    # Compute alternative's global weights
    wG = np.dot(W, u)
    print(f"\nAlternatives' global weight wrt {G}")
    for i, coy in enumerate(AL):
        print(f"  {coy}: {wG[i]:.6f}")

    # Perform sensitivity analysis on Criteria
    for k, cr in enumerate(C):
        WG_dict = {}
        for p in np.linspace(0,1,11):
            adj_u = renorm_wt(p, k, u)
            WG_dict[p] = np.dot(W, adj_u)
        rainbow_diagram(WG_dict, AL, cr, base_value=u[k])


def rainbow_diagram(w_dict, alternatives, criterion, base_value=None):
    """ Plot the rainbow diagram
    Parameters:
      w = dictionary of array of alternative weights of the form
          { p : [w1, w2, ..., wn ]  where 0 <= p <= 1.
      alternatives = list of alternatives
      criterion = criterion name
      base_value = base value of criterion being varied
    """
    fig, ax = plt.subplots(figsize=(10,8))
    ax.plot(w_dict.keys(), w_dict.values(), label=alternatives, lw='2')
    if base_value is not None:
        ax.plot([base_value, base_value], [0, 1], '--', color='black')
    ax.set_title(f"Rainbow diagram for criterion {criterion}", fontsize='x-large')
    ax.set_xlim(0,1)
    ax.set_ylim(0,1)
    ax.set_xticks(np.linspace(0,1,11))
    ax.set_yticks(np.linspace(0,1,11))
    ax.set_xlabel(f'Weight of Criterion {criterion}', fontsize='x-large')
    ax.set_ylabel('Weight of Alternatives', fontsize='x-large')
    ax.legend(fontsize='x-large')
    ax.grid()
    plt.show()


def renorm_wt(p, k, base_wt):
    """ Renormalize the weights when one weight is change
        while keeping all the other weights in their original
        proportions
    Parameters:
        p = new value between 0 and 1
        k = index between 0 and n-1
        base_wt = base weights
    Returns:
        a renormalized weight vector
    """

    new_wt = base_wt.copy()
    bal_wt = base_wt.sum() - base_wt[k]
    for i, w in enumerate(base_wt):
        if i != k:
            new_wt[i] = (1-p)*base_wt[i]/bal_wt
    new_wt[k] = p
    return new_wt



def AHPmat(A, method='Power'):
    """ Comput AHP matrix A using chosen method
    Parameter:  A = matrix to evaluate
    Returns:    w, lambda_max, CI, CR
    """

    RI=(0.58,0.90,1.12,1.24,1.32,1.41,1.45,1.49,1.51,1.54,1.56,1.57,1.58)
```

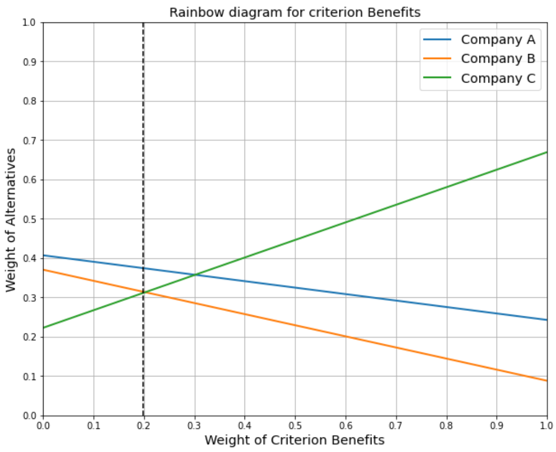
```python
    def Power(A):
        """ Compute the AHP matrix A using Power Iterations method.
            Parameter:  A = matrix to evaluate
            Returns:    w, lambda_max, CI, CR
        """
        gm = stats.gmean(A, axis=1)   # Use RGM method as initial value
        w = gm/gm.sum()
        max_iter= 1000000
        epsilon = 1.E-12
        for iter in range(max_iter):
            w1 = np.dot(A,w)     # w(k+1) = A w(k)
            w1 = w1/w1.sum()     # normalize w(k+1)
            if all(np.absolute(w1-w) < epsilon):
                w = w1
                break
            w = w1
        lambda_max = (np.dot(A,w)/w).mean()
        n, _ = A.shape
        CI = (lambda_max-n)/(n-1)
        CR = 0 if n==2 else CI/RI[n-3]
        return w, lambda_max, CI, CR

    def Algebra(A):
        """ Compute the AHP matrix A using Power Iterations method.
            Parameter:  A = matrix to evaluate
            Returns:    w, lambda_max, CI, CR
        """
        n, _ = A.shape
        # Solve for lambda such that Det(A - lambda*I) = 0
        sol = root(lambda x: np.linalg.det(A-np.eye(n)*x), n)
        lambda_max = sol.x[0]
        # Find w by solving a set of linear equations M w = b
        # M = A - lambda_max I for first n-1 rows
        M = A - np.eye(n)*lambda_max
        # Replace the last row with [1, 1..., 1]
        M[n-1] = np.ones(n)
        b = np.append(np.zeros(n-1), [1])  # b = [0, 0, ..., 1]
        w = np.linalg.solve(M,b)
        CI = (lambda_max-n)/(n-1)
        CR = 0 if n == 2 else CI/RI[n-3]
        return w, lambda_max, CI, CR


    """ The RGM method is not recommended as you can do better with
        Algebra or Power method.  You can use it to compare results """
    def RGM(A):
        """ Compute the AHP matrix A using the RGM approximation method.
            Parameter:  A = matrix to evaluate
            Returns:    w, lambda_max, CI, CR
        """
        n, _ = A.shape
        gm = stats.gmean(A, axis=1)
        w = gm/gm.sum()
        lambda_max = (np.dot(A,w)/w).mean()
        CI = (lambda_max-n)/(n-1)
        CR = 0 if n==2 else CI/RI[n-3]
        return w, lambda_max, CI, CR

    # We just need the w vector
    if method=='Power':
        return Power(A)[0]
    elif method=='Algebra':
        return Algebra(A)[0]
    elif method=='RGM':
        return RGM(A)[0]
    else:
        print("Invalid method chosen")
        exit()

if __name__=="__main__":
    main()
```

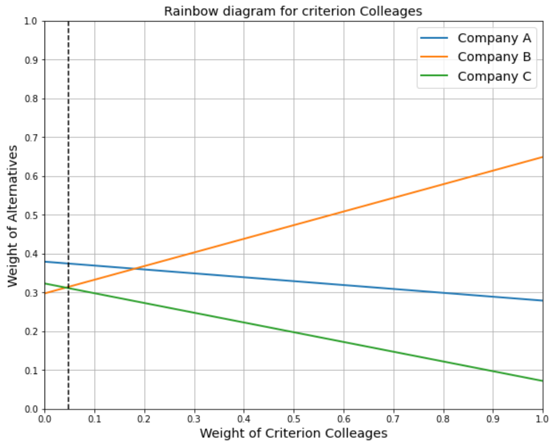
## Output

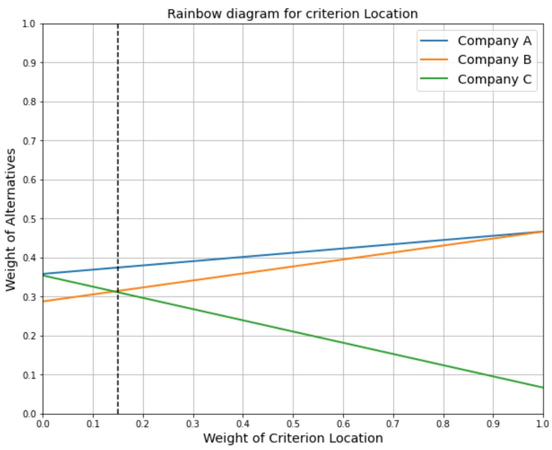
```
Criteria's Weights
  Research  : 0.158408
  Growth    : 0.189247
  Benefits  : 0.197997
  Colleages : 0.048310
  Location  : 0.150245
  Reputation: 0.255792

Alternatives' local Weights
[[0.1364998  0.09739007 0.24263692 0.27895457 0.46666667 0.79275736]
 [0.62501307 0.33306935 0.08794621 0.649118   0.46666667 0.13122122]
 [0.23848712 0.56954058 0.66941687 0.07192743 0.06666667 0.07602141]]

Alternatives' global weights wrt Job Satisfaction
  Company A: 0.374467
  Company B: 0.314491
  Company C: 0.311042
```

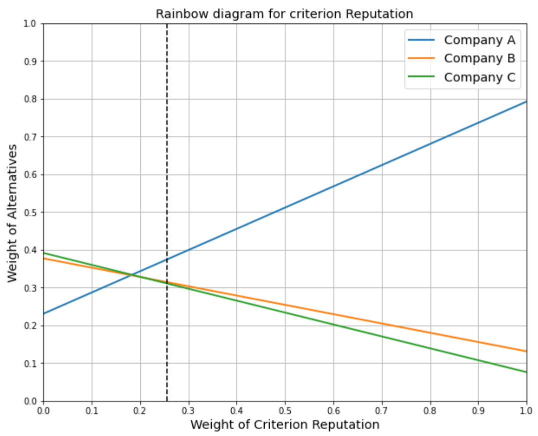


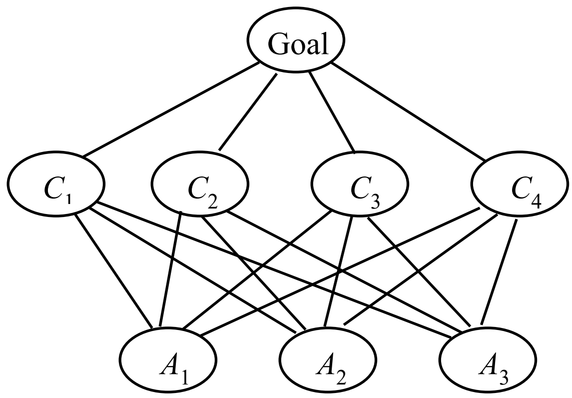



## 5 AHP Models with Complex Hierarchies

### 5.1 Models with more than 3 levels

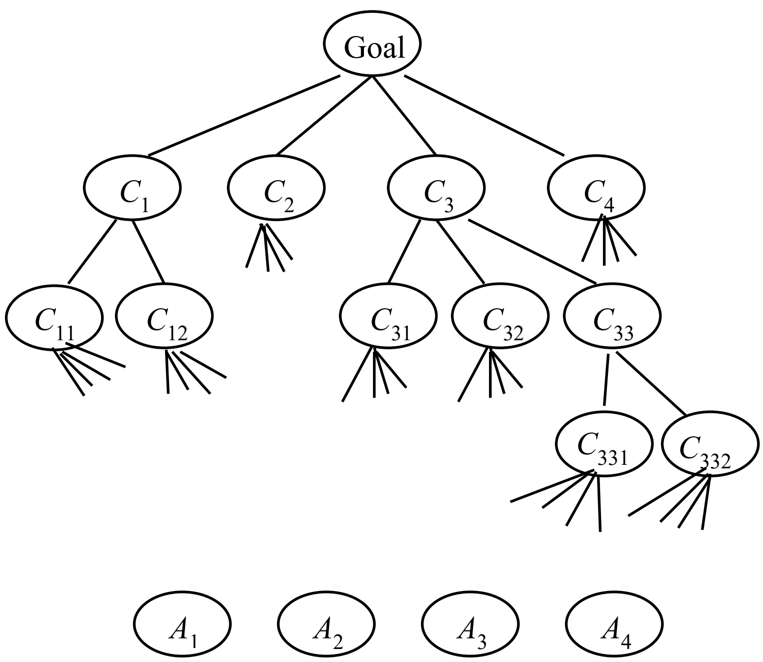
- So far, we have seen the use of a 3-level complete hierarchy where the second level signifies the criteria for evaluation, and the third or lowest level denotes the alternative to be evaluated.




- We call such a hierarchy a **Simple 3-Level Hierarchy.** It is the simplest form of hierarchy for performing multiple criteria evaluation of alternatives.

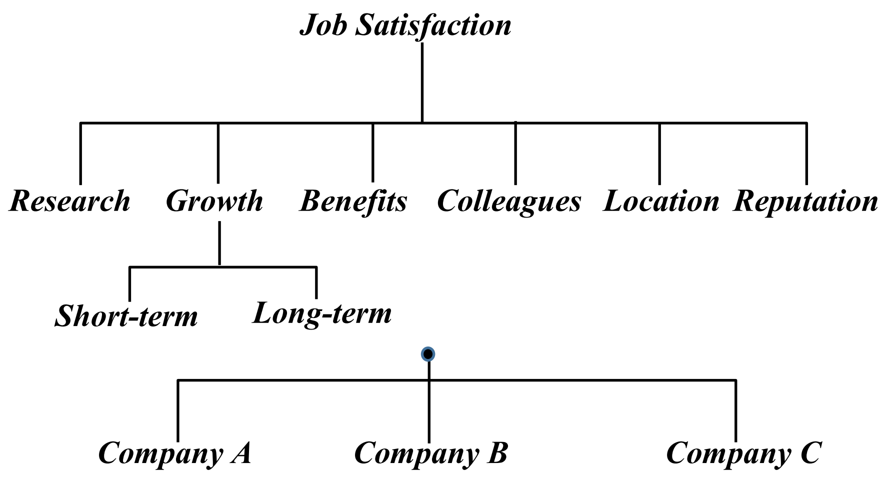
**Multi-Level Tree Hierarchies**

- An AHP model may have main criteria which may be decomposed into sub-criteria, and some of the sub-criteria may have further decomposed, etc.
- The hierarchy does not need to be balanced in the depth of sub-trees.
- The criteria that are directly above the alternatives are called leaf-criteria.



## 5.2 Case Study: Job Selection Problem with Growth Sub-criteria

- In the Job Selection Problem, suppose that the Growth criterion has two sub-criteria: short-term growth and long-term growth:




- We pairwise compare the 6 criteria w.r.t. the Goal and obtain the same weights as before.

- The two sub-criteria are judged w.r.t. their parent (Growth):

| | Short-term growth | Long-term growth |
|---|---|---|
| Short-term growth | 1 | 1/3 |
| Long-term growth | 3 | 1 |

$\boldsymbol{w}$ = [0.25, 0.75]

- We pairwise compare the 3 alternatives w.r.t. the seven leaf criteria and/or sub-criteria:

- The results for the criteria Research, Benefits, Colleagues, Location, and Reputations are as before.

- The results for the sub-criteria short-term growth and long-term growth are as follows:

Comparison of alternatives w.r.t. "Short-Term Growth":

| | A | B | C |
|---|---|---|---|
| A | 1 | 1/3 | 1/7 |
| B | | 1 | 1/3 |
| C | | | 1 |

$\lambda_{max}$ = 3.0070, CI= 0.003511, CR.=0.006053 < 0.1, $\boldsymbol{w}$ = [0.08795, 0.24264, 0.66942]

Comparison of alternatives w.r.t. "Long-Term Growth":

| | A | B | C |
|---|---|---|---|
| A | 1 | 3 | 5 |
| B | | 1 | 2 |
| C | | | 1 |

$\lambda_{max}$ = 3.0037, CI =0.001847, CR =0.003815 < 0.1, $\boldsymbol{w}$ = [0.64833, 0.22965, 0.12202]

- The global weights for the alternatives are computed as follows:

| | Criteria | Sub-criteria (if any) | Global weights of leaf criteria | Alt's local weight w.r.t Leaf Criterion |
|---|---|---|---|---|
| 1 | Research (0.158408) | | 0.158408 | |
| | | | | Job A (0.13650) |
| | | | | Job B (0.62501) |
| | | | | Job C (0.23849) |
| 2 | Growth (0.189247) | | | |
| 2.1 | | Short-term (0.25) | 0.047312 | |
| | | | | Job A (0.08795) |
| | | | | Job B (0.24264) |
| | | | | Job C (0.66942) |
| 2.1 | | Long-term (0.75) | 0.141935 | |
| | | | | Job A (0.64833) |
| | | | | Job B (0.22965) |
| | | | | Job C (0.12201) |
| 3 | Benefits (0.197997) | | 0.197997 | |
| | | | | Job A (0.24264) |
| | | | | Job B (0.08795) |
| | | | | Job C (0.66942) |
| 4 | Colleagues (0.048310) | | 0.048310 | |
| | | | | Job A (0.27895) |
| | | | | Job B (0.64912) |
| | | | | Job C (0.07193) |
| 5 | Location (0.150245) | | 0.150245 | |
| | | | | Job A (0.46667) |
| | | | | Job B (0.46667) |
| | | | | Job C (0.06667) |
| 6 | Reputation (0.255792) | | 0.255792 | |
| | | | | Job A (0.79276) |
| | | | | Job B (0.13122) |
| | | | | Job C (0.07602) |

- Global weight for criterion "Short-Term Growth" = $0.25 \times 0.189247 = 0.047312$
- Global weight for criterion "Long-Term Growth" = $0.75 \times 0.189247 = 0.141935$

- The Global Weight of an alternative is equal to the leaf criterion-weighted sum of its local weights. In Matrix notations:

$$\begin{bmatrix} w_A \\ w_B \\ w_C \end{bmatrix} = \begin{bmatrix} 0.13650 & 0.08795 & 0.64833 & 0.24264 & 0.27895 & 0.46667 & 0.79276 \\ 0.62501 & 0.24264 & 0.22965 & 0.08795 & 0.64912 & 0.46667 & 0.13122 \\ 0.23849 & 0.66942 & 0.12202 & 0.66942 & 0.07193 & 0.06667 & 0.07602 \end{bmatrix} \begin{bmatrix} 0.158408 \\ 0.047312 \\ 0.141935 \\ 0.197997 \\ 0.048310 \\ 0.150245 \\ 0.255792 \end{bmatrix} = \begin{bmatrix} 0.4522 \\ 0.2955 \\ 0.2523 \end{bmatrix}$$

- The results are as follows:

| Alternative | Global weight |
|---|---|
| Job A | **0.4522** |
| Job B | 0.2955 |
| Job C | 0.2523 |

- Hence choose Job A since it has the highest global weight.

# General 4-Level AHP Model

Given a General 4-Level AHP Model where not all the main criteria have sub-criteria.

- Goal $G$.
- $\boldsymbol{C} = \{c_1, c_2 \ldots, c_n\}$ = set of main criteria.
- $\boldsymbol{s}_i = \{s_{i1}, s_{i2}, \ldots, s_{im_i}\}$ = set of sub-criteria of $c_i, \forall c_i \in \boldsymbol{C}$.
- $\boldsymbol{S} = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_n\}$.
- $\boldsymbol{A} = \{A_1, A_2, \ldots, A_p\}$ = set of alternatives.

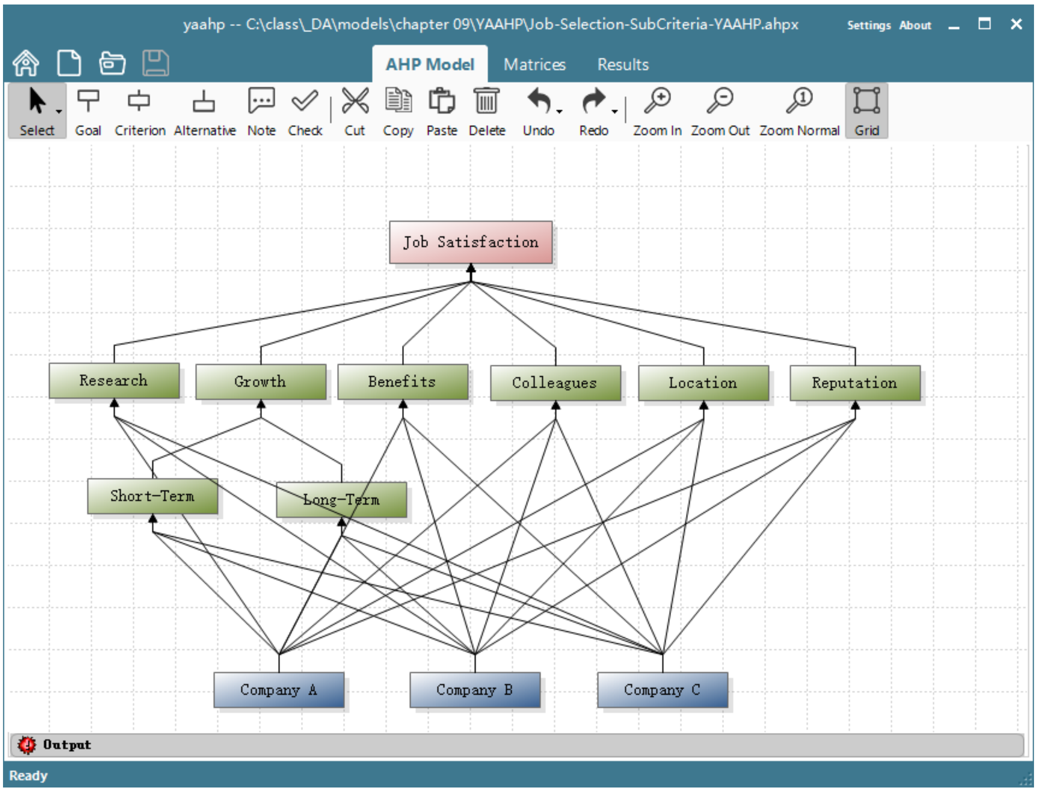
Let $pwc(\boldsymbol{X}, y)$ denotes the normalized vector of weights obtained by pairwise compariing the elements of $\boldsymbol{X}$ w.r.t. element $y$.

Evaluate General 4-Level AHP Model

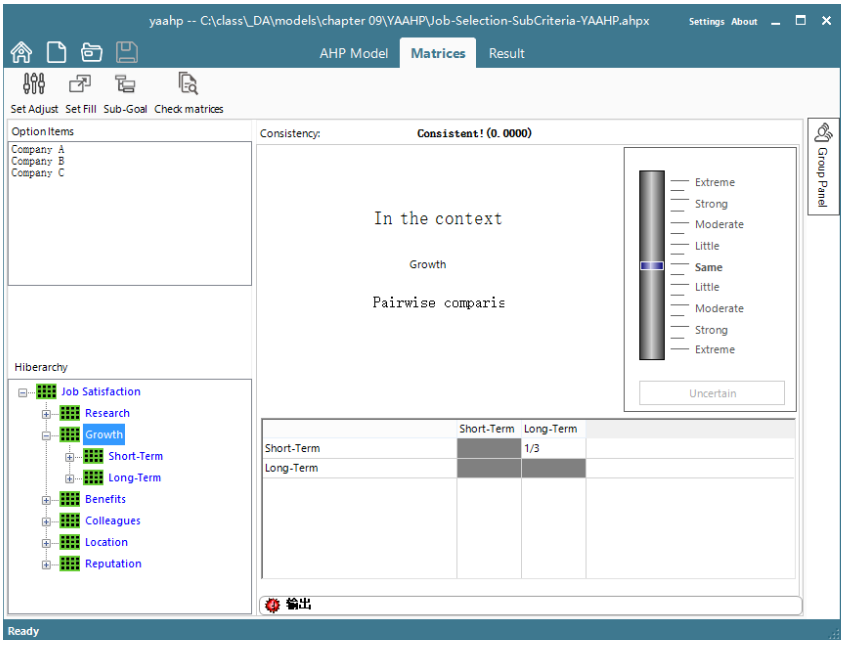
**procedure** G4L-AHP($G, \boldsymbol{C}, \boldsymbol{S}, \boldsymbol{A}$)
    $\boldsymbol{u} \leftarrow pwc(\boldsymbol{C}, G)$ // criteria's global weights
    **for** $i \leftarrow 1, n$ **do**
        **if** $\boldsymbol{s}_i \neq \varnothing$ **then** // there are sub-criteria
            $\boldsymbol{v}_i \leftarrow pwc(\boldsymbol{s}_i, c_i)$ // sub-criteria's local weights
            $\boldsymbol{v}_i^G \leftarrow u_i \boldsymbol{v}_i$ // sub-criteria's global weights
            **for** $j \leftarrow 1, m_i$ **do**
                $\boldsymbol{w}_{ij} \leftarrow pwc(\boldsymbol{A}, s_{ij})$ // alternative's local weights
            **end for**
            $\boldsymbol{W}_i \leftarrow [\boldsymbol{w}_{i1} \boldsymbol{w}_{i2} \ldots \boldsymbol{w}_{im_i}]$ // form a $(p \times m_i)$ matrix
        **else** // there is no sub-criteria
            $\boldsymbol{v}_i^G \leftarrow [u_i]$ // form a $(1 \times 1)$ matrix
            $\boldsymbol{W}_i \leftarrow pwc(\boldsymbol{A}, c_i)$ // alternatives' local weights
        **end if**
    **end for**
    $\boldsymbol{w}^G \leftarrow \sum_i^n \boldsymbol{W}_i \boldsymbol{v}_i^G$ // alternatives's global weights
    **return** $\boldsymbol{w}^G$
**end procedure**

## 5.2 Using YAAHP Application Software

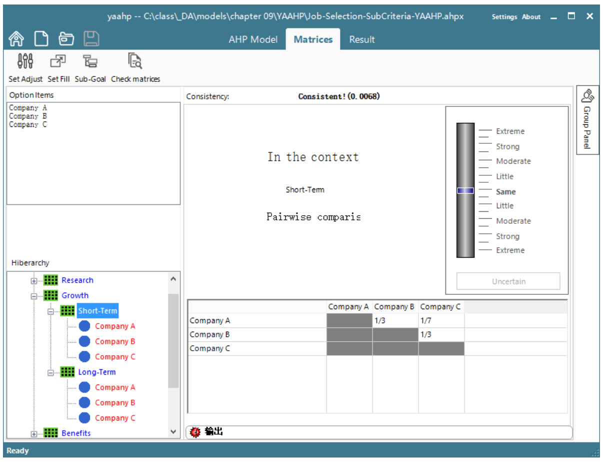
**The Hierarchy with Sub Criteria**

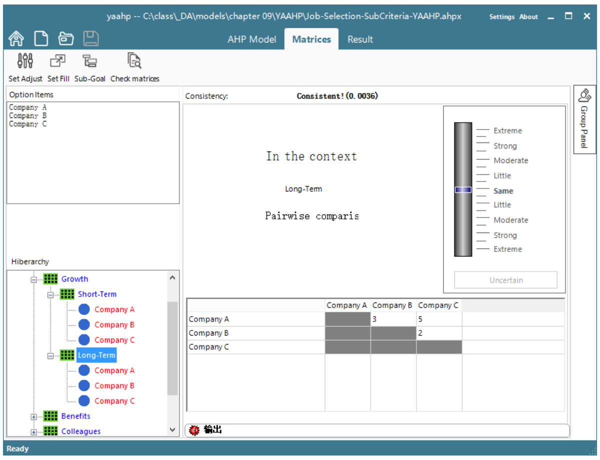



**Pairwise comparison of sub-criteria with respective Growth**

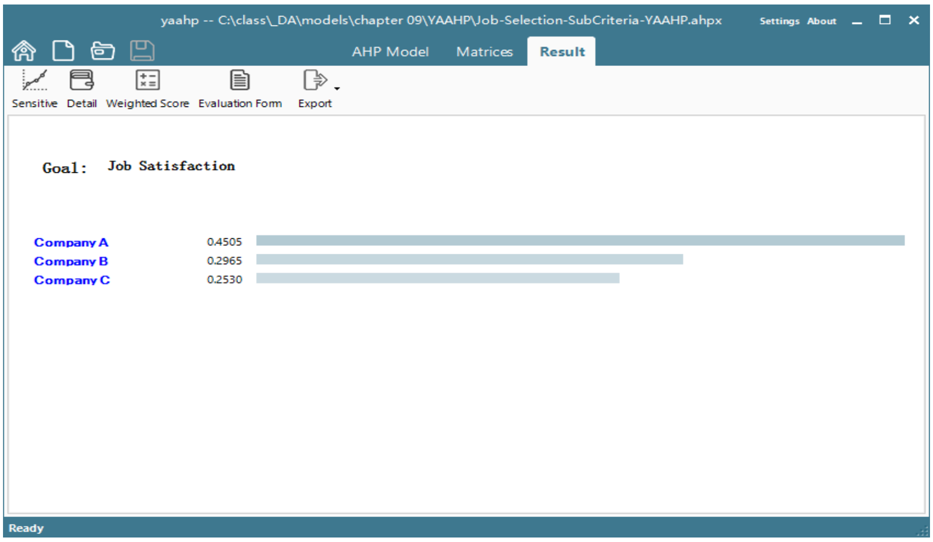


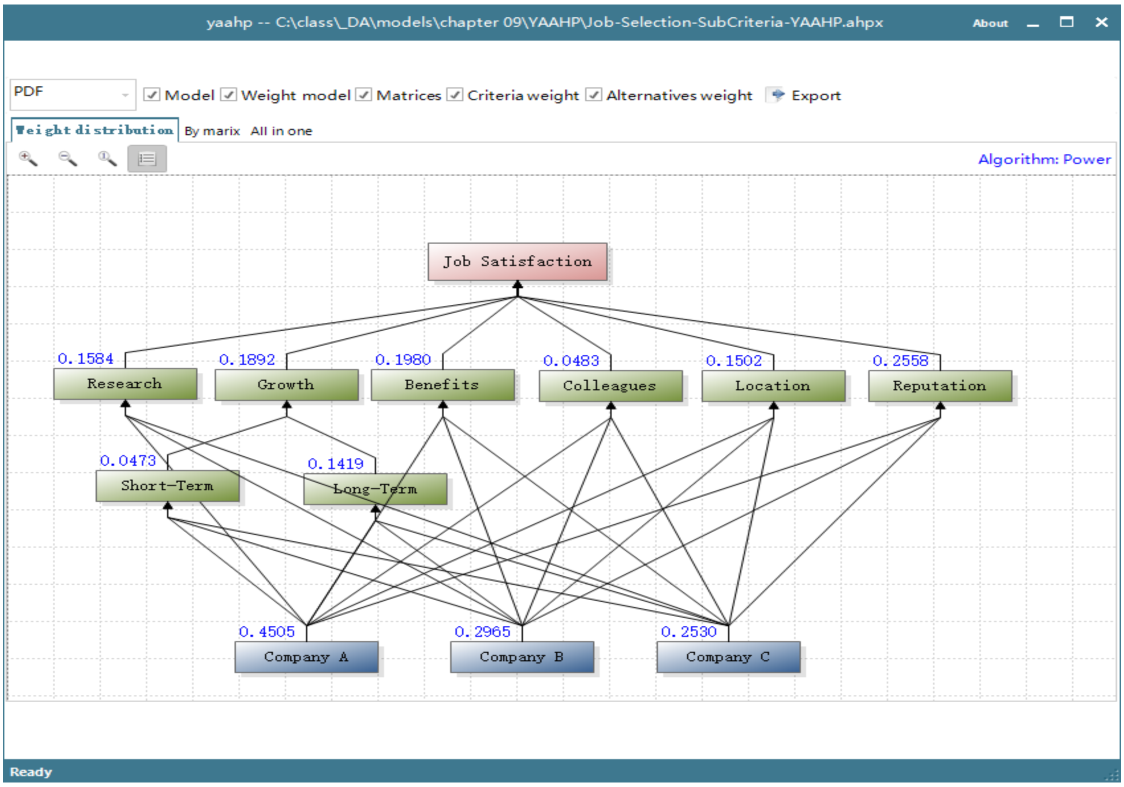
# Pairwise Comparison of Alternatives w.r.t. Sub-criteria



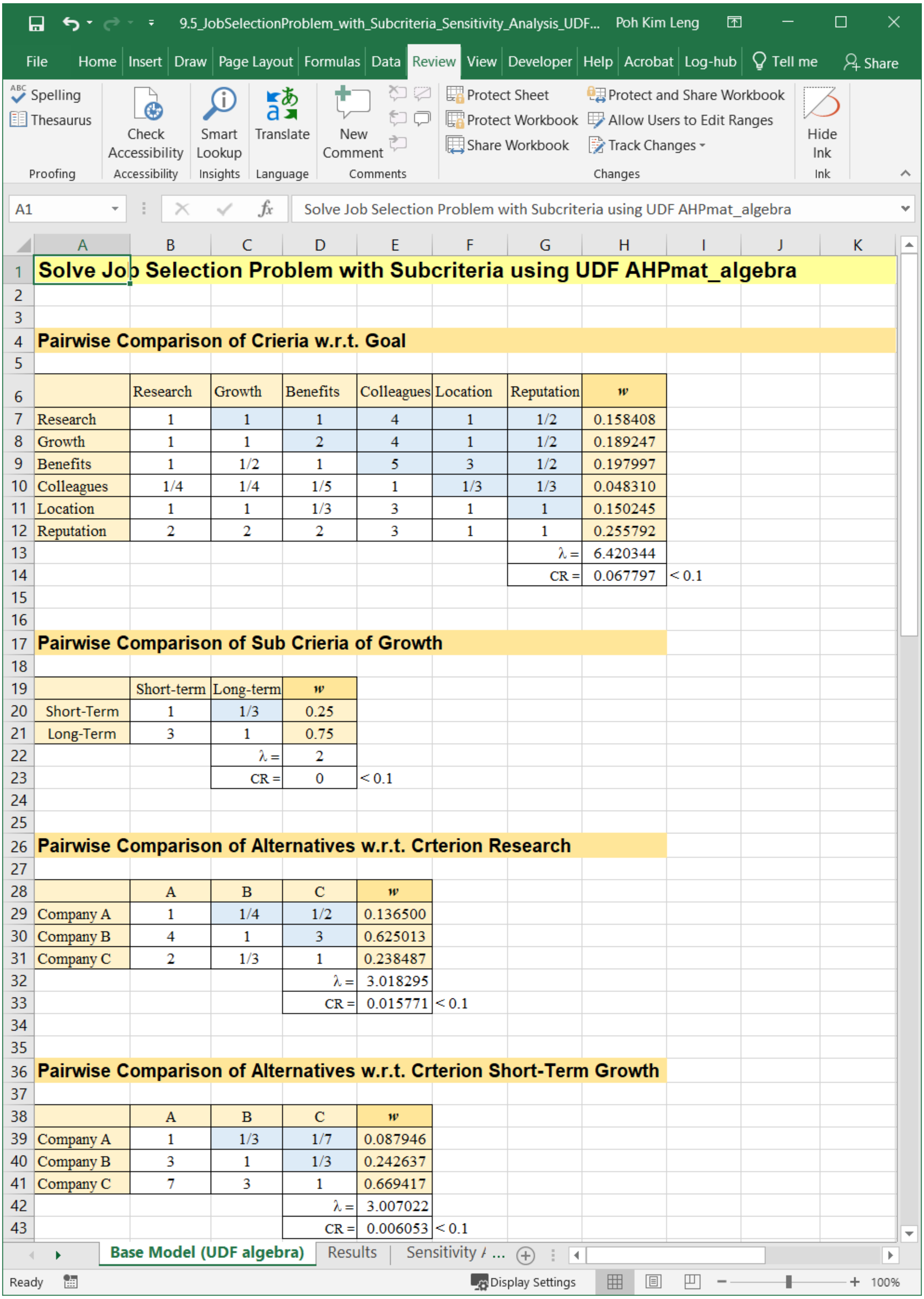




# Results



## 5.3 Using Excel with User-Defined Functions




**Solve Job Selection Problem with Subcriteria using UDF AHPmat_algebra**

**Pairwise Comparison of Crieria w.r.t. Goal**

| | Research | Growth | Benefits | Colleagues | Location | Reputation | *w* | |
|---|---|---|---|---|---|---|---|---|
| Research | 1 | 1 | 1 | 4 | 1 | 1/2 | 0.158408 | |
| Growth | 1 | 1 | 2 | 4 | 1 | 1/2 | 0.189247 | |
| Benefits | 1 | 1/2 | 1 | 5 | 3 | 1/2 | 0.197997 | |
| Colleagues | 1/4 | 1/4 | 1/5 | 1 | 1/3 | 1/3 | 0.048310 | |
| Location | 1 | 1 | 1/3 | 3 | 1 | 1 | 0.150245 | |
| Reputation | 2 | 2 | 2 | 3 | 1 | 1 | 0.255792 | |
| | | | | | | λ = | 6.420344 | |
| | | | | | | CR = | 0.067797 | < 0.1 |

**Pairwise Comparison of Sub Crieria of Growth**

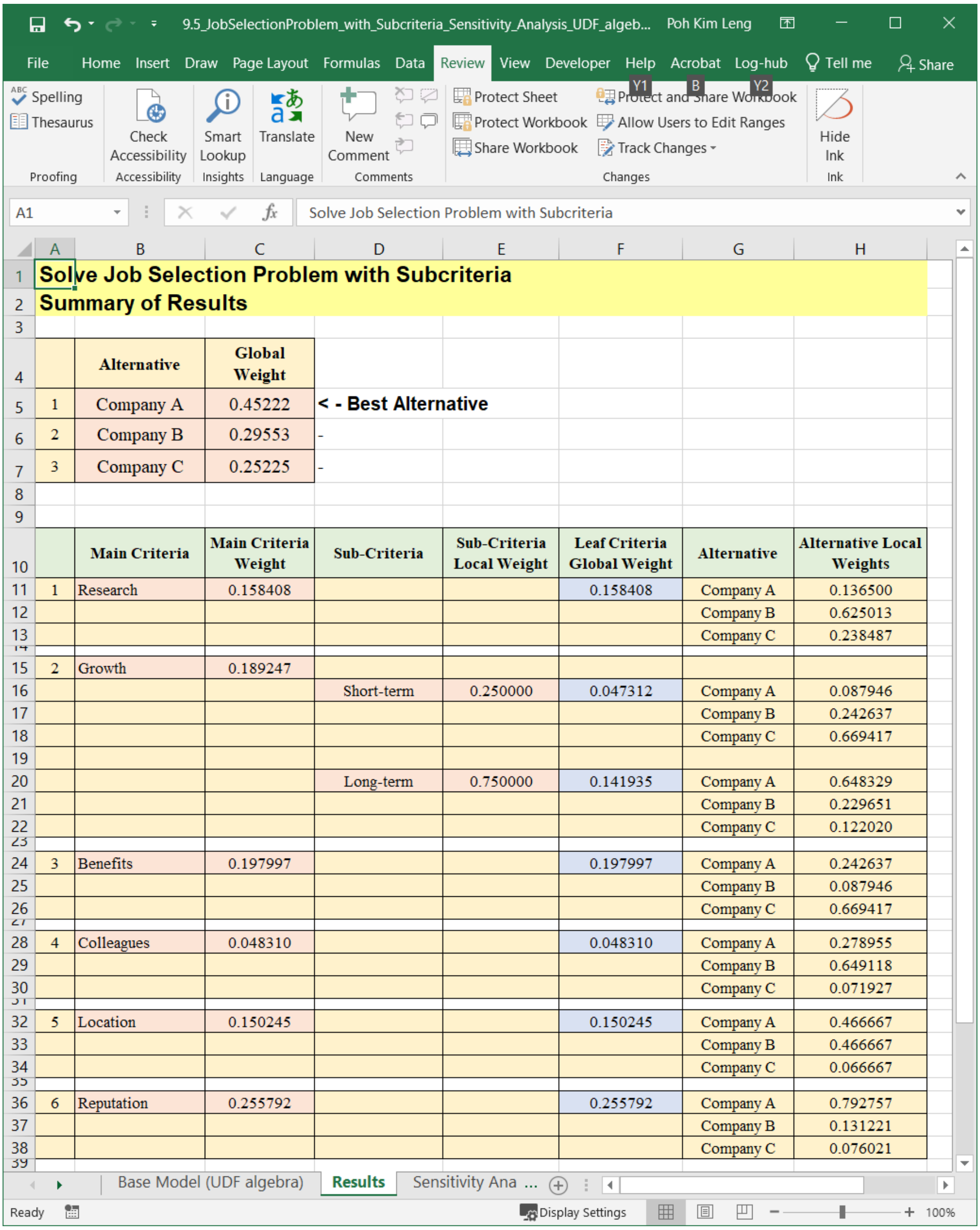
| | Short-term | Long-term | *w* | |
|---|---|---|---|---|
| Short-Term | 1 | 1/3 | 0.25 | |
| Long-Term | 3 | 1 | 0.75 | |
| | | λ = | 2 | |
| | | CR = | 0 | < 0.1 |

**Pairwise Comparison of Alternatives w.r.t. Crterion Research**

| | A | B | C | *w* | |
|---|---|---|---|---|---|
| Company A | 1 | 1/4 | 1/2 | 0.136500 | |
| Company B | 4 | 1 | 3 | 0.625013 | |
| Company C | 2 | 1/3 | 1 | 0.238487 | |
| | | | λ = | 3.018295 | |
| | | | CR = | 0.015771 | < 0.1 |

**Pairwise Comparison of Alternatives w.r.t. Crterion Short-Term Growth**

| | A | B | C | *w* | |
|---|---|---|---|---|---|
| Company A | 1 | 1/3 | 1/7 | 0.087946 | |
| Company B | 3 | 1 | 1/3 | 0.242637 | |
| Company C | 7 | 3 | 1 | 0.669417 | |
| | | | λ = | 3.007022 | |
| | | | CR = | 0.006053 | < 0.1 |



**Solve Job Selection Problem with Subcriteria**

**Summary of Results**

| | Alternative | Global Weight | |
|---|---|---|---|
| 1 | Company A | 0.45222 | < - Best Alternative |
| 2 | Company B | 0.29553 | - |
| 3 | Company C | 0.25225 | - |

| | Main Criteria | Main Criteria Weight | Sub-Criteria | Sub-Criteria Local Weight | Leaf Criteria Global Weight | Alternative | Alternative Local Weights |
|---|---|---|---|---|---|---|---|
| 1 | Research | 0.158408 | | | 0.158408 | Company A | 0.136500 |
| | | | | | | Company B | 0.625013 |
| | | | | | | Company C | 0.238487 |
| 2 | Growth | 0.189247 | | | | | |
| | | | Short-term | 0.250000 | 0.047312 | Company A | 0.087946 |
| | | | | | | Company B | 0.242637 |
| | | | | | | Company C | 0.669417 |
| | | | Long-term | 0.750000 | 0.141935 | Company A | 0.648329 |
| | | | | | | Company B | 0.229651 |
| | | | | | | Company C | 0.122020 |
| 3 | Benefits | 0.197997 | | | 0.197997 | Company A | 0.242637 |
| | | | | | | Company B | 0.087946 |
| | | | | | | Company C | 0.669417 |
| 4 | Colleagues | 0.048310 | | | 0.048310 | Company A | 0.278955 |
| | | | | | | Company B | 0.649118 |
| | | | | | | Company C | 0.071927 |
| 5 | Location | 0.150245 | | | 0.150245 | Company A | 0.466667 |
| | | | | | | Company B | 0.466667 |
| | | | | | | Company C | 0.066667 |
| 6 | Reputation | 0.255792 | | | 0.255792 | Company A | 0.792757 |
| | | | | | | Company B | 0.131221 |
| | | | | | | Company C | 0.076021 |

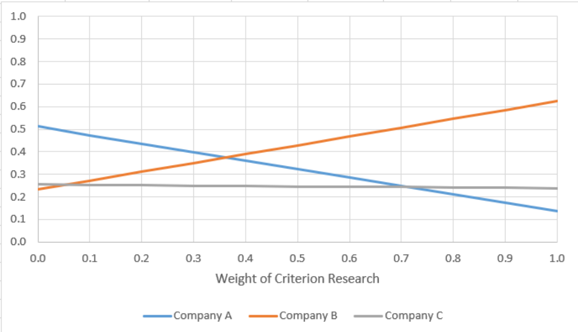
9.5_JobSelectionProblem_with_Subcriteria_Sensitivity_Analysis_UDF_algebra.xlsx - Excel

A1 | Solve Job Selection Problem with Subcriteria

**Rainbow Diagram for Criterion Research**

| Leaf weights | 0.158408 | 0.047312 | 0.141935 | 0.197997 | 0.048310 | 0.150245 | 0.255792 | 1.00 |
|---|---|---|---|---|---|---|---|---|

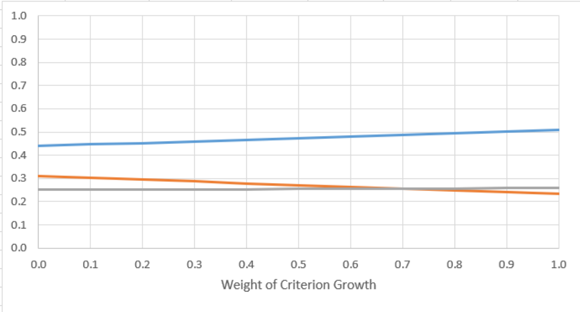
| | A | B | C |
|---|---|---|---|
| | 0.4522 | 0.2955 | 0.2522 |
| 0.0 | 0.5116 | 0.2335 | 0.2548 |
| 0.1 | 0.4741 | 0.2727 | 0.2532 |
| 0.2 | 0.4366 | 0.3118 | 0.2516 |
| 0.3 | 0.3991 | 0.3510 | 0.2499 |
| 0.4 | 0.3616 | 0.3901 | 0.2483 |
| 0.5 | 0.3241 | 0.4293 | 0.2467 |
| 0.6 | 0.2866 | 0.4684 | 0.2450 |
| 0.7 | 0.2490 | 0.5076 | 0.2434 |
| 0.8 | 0.2115 | 0.5467 | 0.2418 |
| 0.9 | 0.1740 | 0.5859 | 0.2401 |
| 1.0 | 0.1365 | 0.6250 | 0.2385 |




**Rainbow Diagram for Criterion Growth**

| Leaf weights | 0.158408 | 0.047312 | 0.141935 | 0.197997 | 0.048310 | 0.150245 | 0.255792 | 1.00 |
|---|---|---|---|---|---|---|---|---|

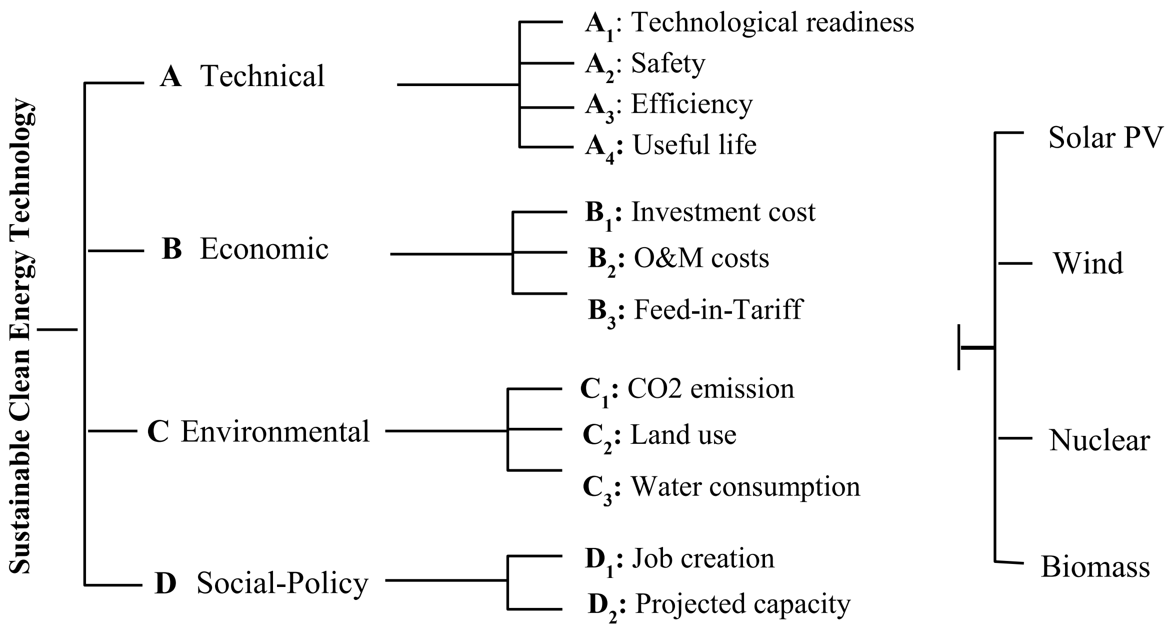
| | A | B | C |
|---|---|---|---|
| | 0.4522 | 0.2955 | 0.2522 |
| 0.0 | 0.4391 | 0.3102 | 0.2507 |
| 0.1 | 0.4461 | 0.3024 | 0.2515 |
| 0.2 | 0.4530 | 0.2947 | 0.2523 |
| 0.3 | 0.4599 | 0.2870 | 0.2532 |
| 0.4 | 0.4668 | 0.2793 | 0.2540 |
| 0.5 | 0.4737 | 0.2715 | 0.2548 |
| 0.6 | 0.4806 | 0.2638 | 0.2556 |
| 0.7 | 0.4875 | 0.2561 | 0.2564 |
| 0.8 | 0.4944 | 0.2483 | 0.2572 |
| 0.9 | 0.5013 | 0.2406 | 0.2581 |
| 1.0 | 0.5082 | 0.2329 | 0.2589 |




Results | Sensitivity Analysis

# 6 Case Study: Evaluation of Sustainable New Energy Technologies

## 6.1 The AHP Hierarchy




## 6.2 Prioritization of Criteria and Sub-Criteria

- Pairwise comparison of main criteria with respect to Goal

| | Technical | Economic | Environmental | Social-Policy | Weight |
|---|---|---|---|---|---|
| Technical | 1 | 1 | 1/5 | 1/2 | 0.109270 |
| Economic | 1 | 1 | 1/5 | 1/2 | 0.109270 |
| Environmental | 5 | 5 | 1 | 3 | 0.572450 |
| Social-Policy | 2 | 2 | 1/3 | 1 | 0.209009 |

$\lambda_{max} = 4.004158$, $CR = 0.001540 < 0.1$

- Pairwise comparison of sub-criteria under "Technical"

| | Technology readiness | Safety | Efficiency | Useful life | Weight |
|---|---|---|---|---|---|
| Technology readiness | 1 | 1 | 3 | 1 | 0.304999 |
| Safety | 1 | 1 | 3 | 1 | 0.304999 |
| Efficiency | 1/3 | 1/3 | 1 | 1/2 | 0.113143 |
| Useful life | 1 | 1 | 2 | 1 | 0.276859 |

$\lambda_{max} = 4.020620$ $CR = 0.007637 < 0.1$

- Pairwise comparison of sub-criteria under "Economic"

| | Investment cost | O&M costs | Feed-in-Tariff | Weight |
|---|---|---|---|---|
| Investment cost | 1 | 1/9 | 1/5 | 0.060328 |
| O&M costs | 9 | 1 | 4 | 0.708524 |
| Feed-in-Tariff | 5 | 1/4 | 1 | 0.231148 |

$\lambda_{max} = 3.071265$ $\quad CR = 0.061436 < 0.1$

- Pairwise comparison of sub-criteria under "Environmental"

| | CO2 emission | Land use | Water consumption | Weight |
|---|---|---|---|---|
| CO2 emission | 1 | 1/4 | 1/3 | 0.126005 |
| Land use | 4 | 1 | 1 | 0.457934 |
| Water consumption | 3 | 1 | 1 | 0.416061 |

$\lambda_{max} = 3.009203$ $\quad CR = 0.007933 < 0.1$

- Pairwise comparison of sub-criteria under "Social & Policy"

| | Job creation | Projected capacity | Weight |
|---|---|---|---|
| Job creation | 1 | 2 | 0.666667 |
| Projected capacity | 1/2 | 1 | 0.333333 |

$\lambda_{max} = 2$ $\quad CR = 0$

### 6.3 Assessment of Alternatives' under Leaf Criteria

- Pairwise comparison of Alternatives w.r.t. sub-criterion "Technology readiness"

| | Solar PV | Wind | Nuclear | Biomass | Weight |
|---|---|---|---|---|---|
| Solar PV | 1 | 1 | 3 | 2 | 0.350913 |
| Wind | 1 | 1 | 3 | 2 | 0.350913 |
| Nuclear | 1/3 | 1/3 | 1 | 1/2 | 0.109114 |
| Biomass | 1/2 | 1/2 | 2 | 1 | 0.189060 |

$\lambda_{max} = 4.010363$ $CR = 0.003838 < 0.1$

Pairwise comparison of Alternatives w.r.t. sub-criterion "Safety"

| | Solar PV | Wind | Nuclear | Biomass | Weight |
|---|---|---|---|---|---|
| Solar PV | 1 | 2 | 3 | 1/2 | 0.286323 |
| Wind | 1/2 | 1 | 2 | 1/2 | 0.182003 |
| Nuclear | 1/3 | 1/2 | 1 | 1/4 | 0.096899 |
| Biomass | 2 | 2 | 4 | 1 | 0.434775 |

$\lambda_{max} = 4.045819$ $CR = 0.016970 < 0.1$

Pairwise comparison of Alternatives w.r.t. sub-criterion "Efficiency"

| | Solar PV | Wind | Nuclear | Biomass | Weight |
|---|---|---|---|---|---|
| Solar PV | 1 | 1/2 | 1/2 | 1/2 | 0.140424 |
| Wind | 2 | 1 | 1 | 2 | 0.339710 |
| Nuclear | 2 | 1 | 1 | 1 | 0.280848 |
| Biomass | 2 | 1/2 | 1 | 1 | 0.239018 |

$\lambda_{max} = 4.060647$ $CR = 0.022462 < 0.1$

Pairwise comparison of Alternatives w.r.t. sub-criterion "Useful life"

| | Solar PV | Wind | Nuclear | Biomass | Weight |
|---|---|---|---|---|---|
| Solar PV | 1 | 1 | 1/2 | 1 | 0.188380 |
| Wind | 1 | 1 | 1/3 | 1/2 | 0.144399 |
| Nuclear | 2 | 3 | 1 | 3 | 0.462341 |
| Biomass | 1 | 2 | 1/3 | 1 | 0.204880 |

$\lambda_{max} = 4.081273$ $CR = 0.030101 < 0.1$

Pairwise comparison of Alternatives w.r.t. sub-criterion "Investment cost"

| | Solar PV | Wind | Nuclear | Biomass | Weight |
|---|---|---|---|---|---|
| Solar PV | 1 | 1/2 | 1 | 1 | 0.200000 |
| Wind | 2 | 1 | 2 | 2 | 0.400000 |
| Nuclear | 1 | 1/2 | 1 | 1 | 0.200000 |
| Biomass | 1 | 1/2 | 1 | 1 | 0.200000 |

$\lambda_{max} = 4$ $\quad CR = 0 < 0.1$

Pairwise comparison of Alternatives w.r.t. sub-criterion "O&M costs"

| | Solar PV | Wind | Nuclear | Biomass | Weight |
|---|---|---|---|---|---|
| Solar PV | 1 | 2 | 1/3 | 1/7 | 0.094454 |
| Wind | 1/2 | 1 | 1/3 | 1/9 | 0.062475 |
| Nuclear | 3 | 3 | 1 | 1/3 | 0.222439 |
| Biomass | 7 | 9 | 3 | 1 | 0.620632 |

$\lambda_{max} = 4.046530$ $\quad CR = 0.017233 < 0.1$

Pairwise comparison of Alternatives w.r.t. sub-criterion "Feed-in-Tariff (FIT)"

| | Solar PV | Wind | Nuclear | Biomass | Weight |
|---|---|---|---|---|---|
| Solar PV | 1 | 1/2 | 1/2 | 1/2 | 0.140424 |
| Wind | 2 | 1 | 1 | 2 | 0.339710 |
| Nuclear | 2 | 1 | 1 | 1 | 0.280848 |
| Biomass | 2 | 1/2 | 1 | 1 | 0.239018 |

$\lambda_{max} = 4.060647$ $\quad CR = 0.022462 < 0.1$

Pairwise comparison of Alternatives w.r.t. sub-criterion "CO2 emission"

| | Solar PV | Wind | Nuclear | Biomass | Weight |
|---|---|---|---|---|---|
| Solar PV | 1 | 1/2 | 1/2 | 3 | 0.188624 |
| Wind | 2 | 1 | 1/2 | 5 | 0.306684 |
| Nuclear | 2 | 2 | 1 | 5 | 0.435730 |
| Biomass | 1/3 | 1/5 | 1/5 | 1 | 0.068963 |

$\lambda_{max} = 4.064806$ $\quad CR = 0.024002 < 0.1$

Pairwise comparison of Alternatives w.r.t. sub-criterion "Land use"

| | Solar PV | Wind | Nuclear | Biomass | Weight |
|---|---|---|---|---|---|
| Solar PV | 1 | 8 | 7 | 9 | 0.707809 |
| Wind | 1/8 | 1 | 1 | 4 | 0.122522 |
| Nuclear | 1/7 | 1 | 1 | 4 | 0.125540 |
| Biomass | 1/9 | 1/4 | 1/4 | 1 | 0.044129 |

$\lambda_{max} = 4.187467$ $\quad CR = 0.069432 < 0.1$

Pairwise comparison of Alternatives w.r.t. sub-criterion "Water consumption"

| | Solar PV | Wind | Nuclear | Biomass | Weight |
|---|---|---|---|---|---|
| Solar PV | 1 | 1/2 | 9 | 8 | 0.364338 |
| Wind | 2 | 1 | 9 | 9 | 0.532925 |
| Nuclear | 1/9 | 1/9 | 1 | 1/2 | 0.041935 |
| Biomass | 1/8 | 1/9 | 2 | 1 | 0.060802 |

$\lambda_{max} = 4.102325$ $\quad CR = 0.037898 < 0.1$

Pairwise comparison of Alternatives w.r.t. sub-criterion "Job creation"

| | Solar PV | Wind | Nuclear | Biomass | Weight |
|---|---|---|---|---|---|
| Solar PV | 1 | 5 | 6 | 4 | 0.614710 |
| Wind | 1/5 | 1 | 1 | 1 | 0.125778 |
| Nuclear | 1/6 | 1 | 1 | 1/2 | 0.101106 |
| Biomass | 1/4 | 1 | 2 | 1 | 0.158406 |

$\lambda_{max} = 4.040705$ $\quad CR = 0.015076 < 0.1$

Pairwise comparison of Alternatives w.r.t. sub-criterion "Projected capacity"

| | Solar PV | Wind | Nuclear | Biomass | Weight |
|---|---|---|---|---|---|
| Solar PV | 1 | 1/5 | 2 | 1/2 | 0.127130 |
| Wind | 5 | 1 | 7 | 2 | 0.548921 |
| Nuclear | 1/2 | 1/7 | 1 | 1/3 | 0.074592 |
| Biomass | 2 | 1/2 | 3 | 1 | 0.249357 |

$\lambda_{max} = 4.017755$ $\quad CR = 0.006576 < 0.1$

## 6.4 Results

| Main Criteria | | |
|---|---|---|
| | **w.r.t. Goal** | Global Wt |
| 1 | Technical | 0.109270 |
| 2 | Economic | 0.109270 |
| 3 | Environmental | 0.572450 |
| 4 | Social and Policy | 0.209009 |

| Sub Criteria | | | |
|---|---|---|---|
| | **Technical** | Local Wt | Global Wt |
| 1 | Technology readiness | 0.304999 | 0.033327 |
| 2 | Safety | 0.304999 | 0.033327 |
| 3 | Efficiency | 0.113143 | 0.012363 |
| 4 | Useful life | 0.276859 | 0.030252 |
| | **Economic** | Local Wt | Global Wt |
| 1 | Investment cost | 0.060328 | 0.006592 |
| 2 | O&M costs | 0.708524 | 0.077421 |
| 3 | Feed-in-Tariff | 0.231148 | 0.025258 |
| | **Environmental** | Local Wt | Global Wt |
| 1 | CO2 emission | 0.126005 | 0.072132 |
| 2 | Land use | 0.457934 | 0.262144 |
| 3 | Water consumption | 0.416061 | 0.238174 |
| | **Social** | Local Wt | Global Wt |
| 1 | Job creation | 0.666667 | 0.139340 |
| 2 | Projected capacity | 0.333333 | 0.069670 |

| Leaf criteria (global weight) | Technology readiness | Safety | Efficiency | Useful life |
|---|---|---|---|---|
| Alternative | 0.033327 | 0.033327 | 0.012363 | 0.030252 |
| Solar PV | 0.35091 | 0.28632 | 0.14042 | 0.18838 |
| Wind | 0.35091 | 0.18200 | 0.33971 | 0.14440 |
| Nuclear | 0.10911 | 0.09690 | 0.28085 | 0.46234 |
| Biomass | 0.18906 | 0.43478 | 0.23902 | 0.20488 |

| Leaf criteria (global weight) | Investment cost | O&M costs | Feed-in-Tariff |
|---|---|---|---|
| Alternative | 0.006592 | 0.077421 | 0.025258 |
| Solar PV | 0.20000 | 0.09445 | 0.14042 |
| Wind | 0.40000 | 0.06248 | 0.33971 |
| Nuclear | 0.20000 | 0.22244 | 0.28085 |
| Biomass | 0.20000 | 0.62063 | 0.23902 |

| Leaf criteria (global weight) | CO2 emission | Land use | Water consumption |
|---|---|---|---|
| Alternative | 0.072132 | 0.262144 | 0.238174 |
| Solar PV | 0.18862 | 0.70781 | 0.36434 |
| Wind | 0.30668 | 0.12252 | 0.53293 |
| Nuclear | 0.43573 | 0.12554 | 0.04194 |
| Biomass | 0.06896 | 0.04413 | 0.06080 |

| Leaf criteria (global weight) | Job creation | Projected capacity |
|---|---|---|
| Alternative | 0.139340 | 0.069670 |
| Solar PV | 0.61471 | 0.12713 |
| Wind | 0.12578 | 0.54892 |
| Nuclear | 0.10111 | 0.07459 |
| Biomass | 0.15841 | 0.24936 |

**Global Weights**

| | Alternative | Global Weight |
|---|---|---|
| 1 | Solar PV | 0.42129 |
| 2 | Wind | 0.27932 |
| 3 | Nuclear | 0.14357 |
| 4 | Biomass | 0.15582 |

## Computations in Matrix Notation

Sub-criteria global weights:

$$\boldsymbol{v}_1^G = 0.109270 \begin{bmatrix} 0.304999 \\ 0.304999 \\ 0.113143 \\ 0.276859 \end{bmatrix} = \begin{bmatrix} 0.033327 \\ 0.033327 \\ 0.012363 \\ 0.030252 \end{bmatrix} \qquad \boldsymbol{v}_2^G = 0.109270 \begin{bmatrix} 0.060328 \\ 0.708524 \\ 0.231148 \end{bmatrix} = \begin{bmatrix} 0.006592 \\ 0.077421 \\ 0.025258 \end{bmatrix}$$

$$\boldsymbol{v}_3^G = 0.572450 \begin{bmatrix} 0.126005 \\ 0.457934 \\ 0.416061 \end{bmatrix} = \begin{bmatrix} 0.072132 \\ 0.262144 \\ 0.238174 \end{bmatrix} \qquad \boldsymbol{v}_4^G = 0.209009 \begin{bmatrix} 0.666667 \\ 0.333333 \end{bmatrix} = \begin{bmatrix} 0.139340 \\ 0.069670 \end{bmatrix}$$

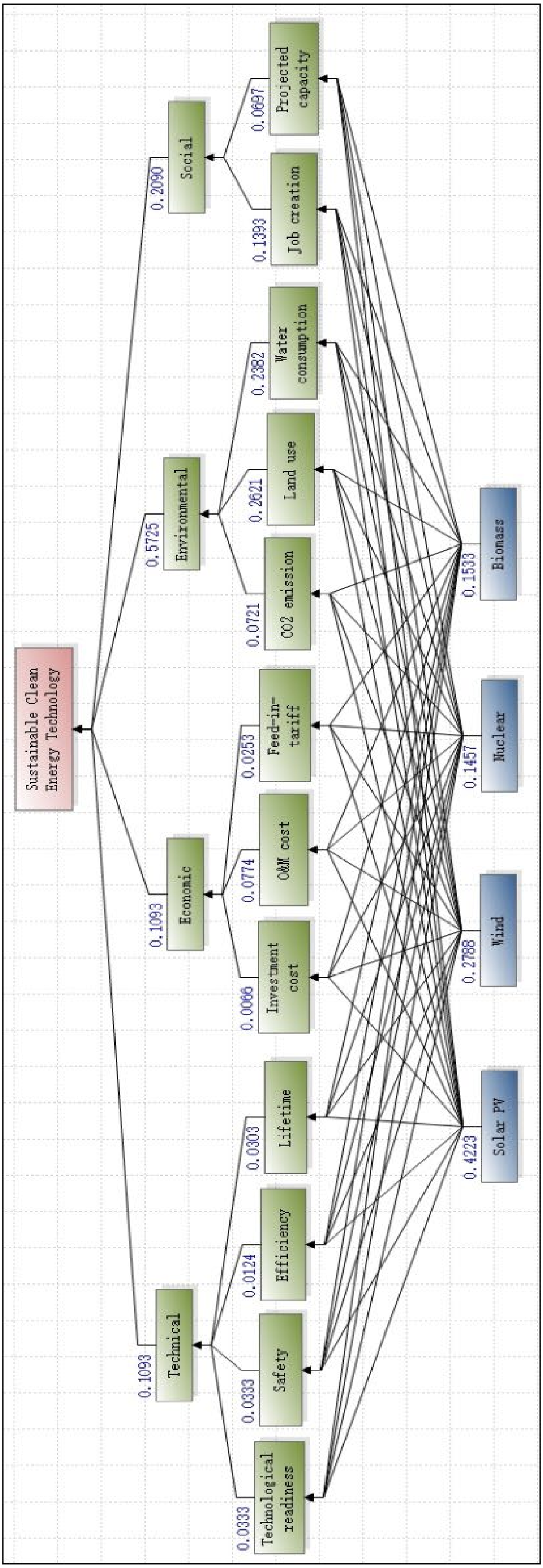
Alternatives weights w.r.t. each leaf-criterion:

$$\boldsymbol{W}_1 = \begin{bmatrix} 0.35091 & 0.28632 & 0.14042 & 0.18838 \\ 0.35091 & 0.18200 & 0.33971 & 0.14440 \\ 0.10911 & 0.09690 & 0.28085 & 0.46234 \\ 0.18906 & 0.43478 & 0.23902 & 0.20488 \end{bmatrix} \qquad \boldsymbol{W}_2 = \begin{bmatrix} 0.20000 & 0.09445 & 0.14042 \\ 0.40000 & 0.06248 & 0.33971 \\ 0.20000 & 0.22244 & 0.28085 \\ 0.20000 & 0.62063 & 0.23902 \end{bmatrix}$$

$$\boldsymbol{W}_3 = \begin{bmatrix} 0.18862 & 0.70781 & 0.36434 \\ 0.30668 & 0.12252 & 0.53293 \\ 0.43573 & 0.12554 & 0.04194 \\ 0.06896 & 0.04413 & 0.06080 \end{bmatrix} \qquad \boldsymbol{W}_4 = \begin{bmatrix} 0.61471 & 0.12713 \\ 0.12578 & 0.54892 \\ 0.10111 & 0.07459 \\ 0.15841 & 0.24936 \end{bmatrix}$$

Alternative global weights:

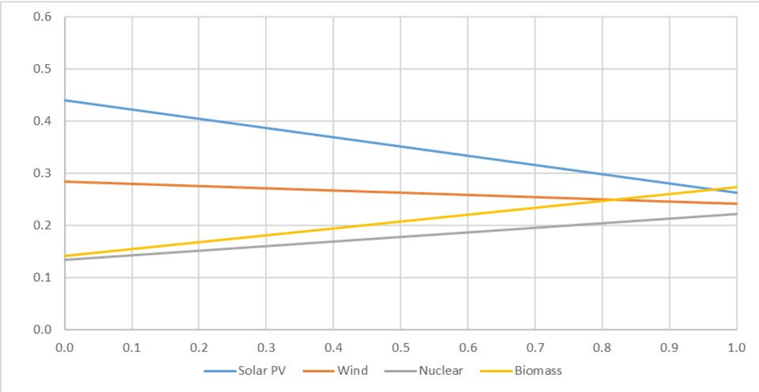
$$\boldsymbol{w}^G = \sum_{i=1}^{4} \boldsymbol{W}_i \boldsymbol{v}_i^G = \begin{bmatrix} 0.42129 \\ 0.27932 \\ 0.14357 \\ 0.15582 \end{bmatrix}$$


Sustainable Clean Energy Technology
0.1093
Technical
0.1093
Economic
0.5725
Environmental
0.2090
Social
0.0333
Technological readiness
0.0333
Safety
0.0124
Efficiency
0.0303
Lifetime
0.0066
Investment cost
0.0774
O&M cost
0.0253
Feed-in-tariff
0.0721
CO2 emission
0.2621
Land use
0.2382
Water consumption
0.1393
Job creation
0.0697
Projected capacity
0.4223
Solar PV
0.2788
Wind
0.1457
Nuclear
0.1533
Biomass

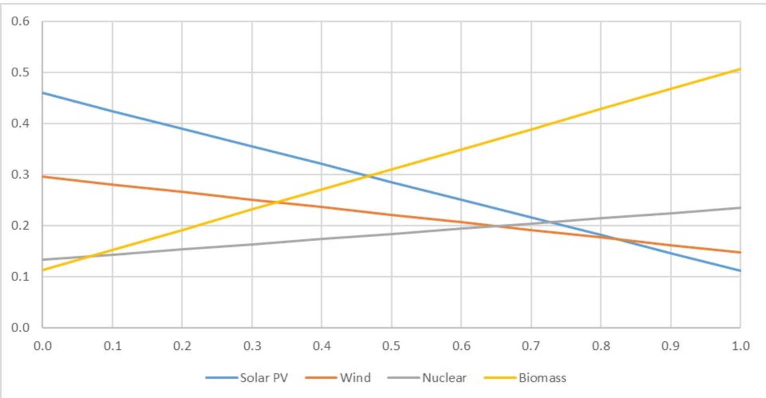
## 6.5 Sensitivity Analysis on Main Criteria Weights

Rainbow diagram for alternative's global weights when the weight of on main criterion is varied from 0 to 1 while keeping the weights of the other main criteria at the same proportion as their base model values.

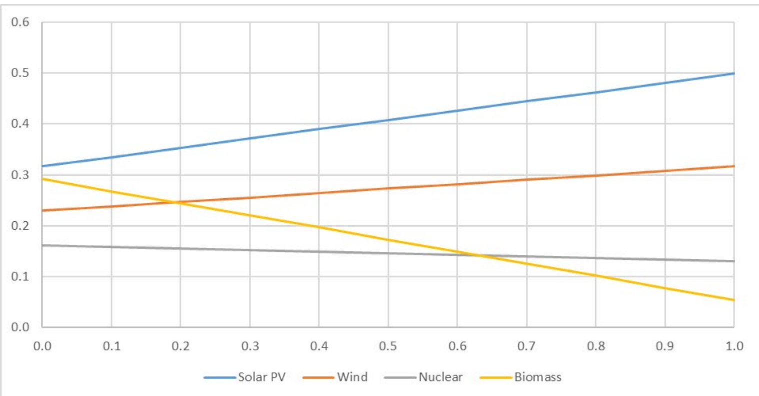
### Main Criterion: Technical



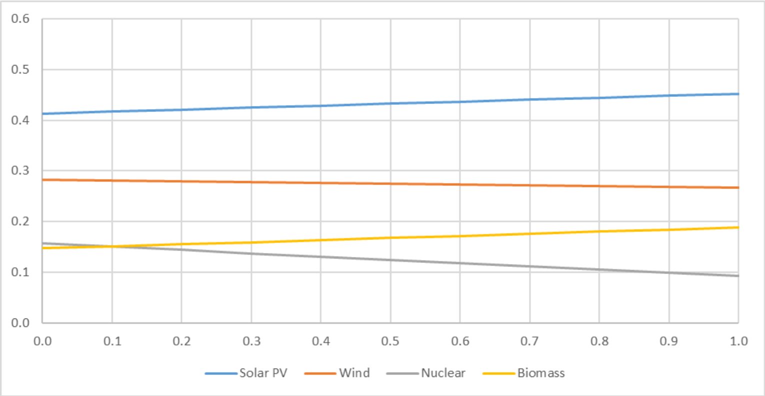

### Main Criterion: Economic




### Main Criterion: Environmental



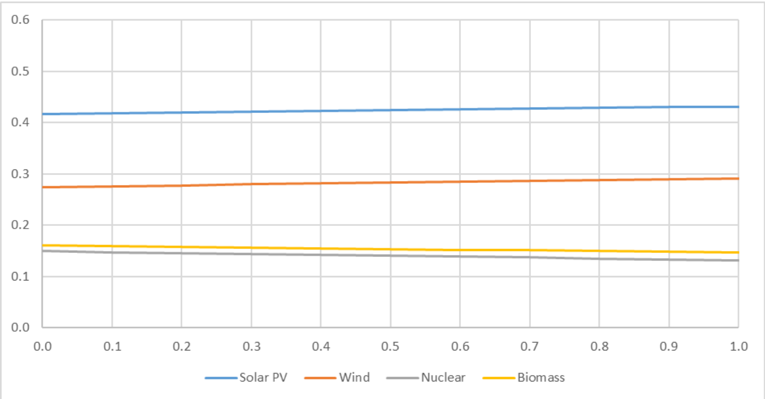
### Main Criterion: Social



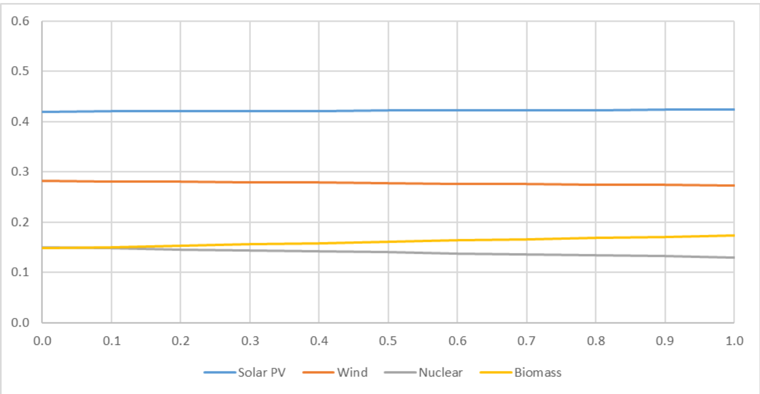

## 6.6 Sensitivity Analysis on Sub-Criteria Weights

Rainbow diagram for alternative's global weights when the weight of a sub-criterion is varied from 0 to 1 while keeping the weights of the other sub-criteria under the same main criterion at the same proportion as their base model values.

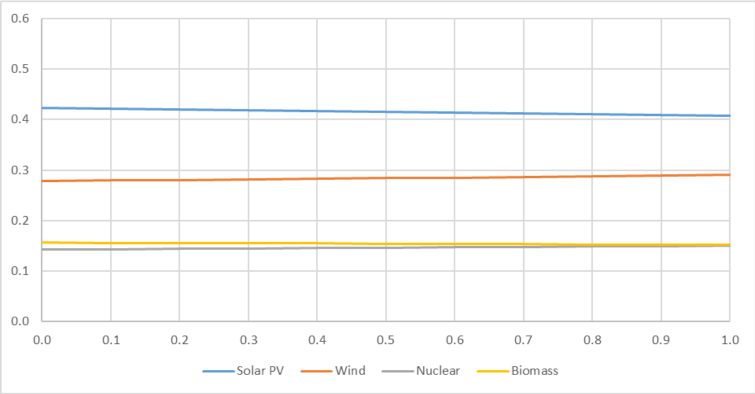
### Sub-Criterion: Technological readiness



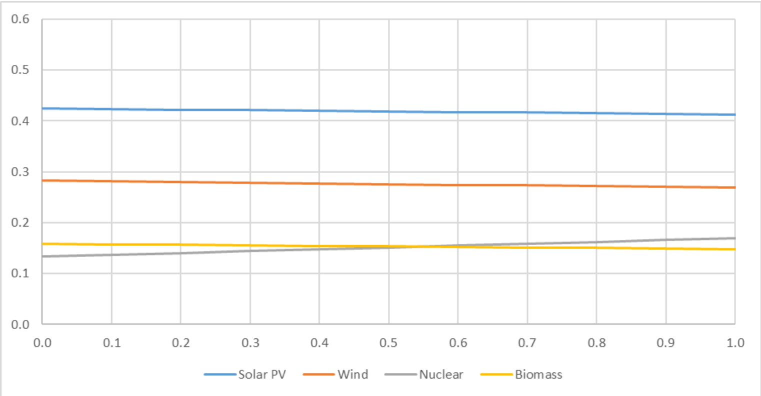

### Sub-Criterion: Safety



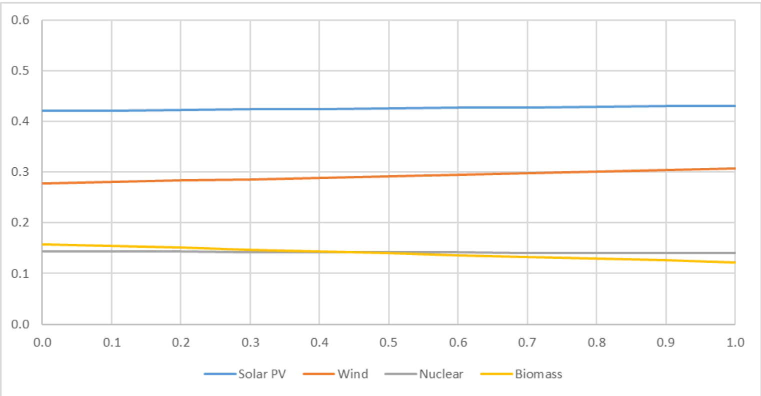
## Sub-Criterion:  Efficiency



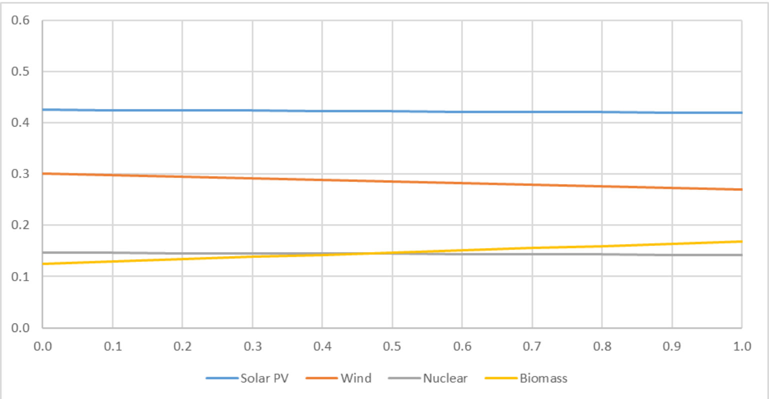

## Sub-Criterion: Useful life



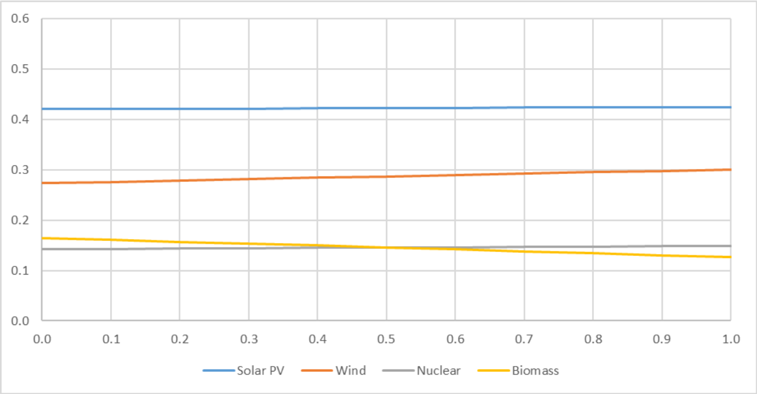

## Sub-Criterion:  Investment cost



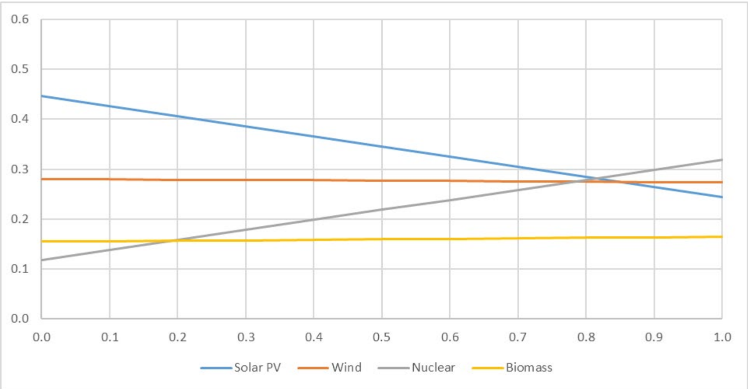

## Sub-Criterion:  O&M costs



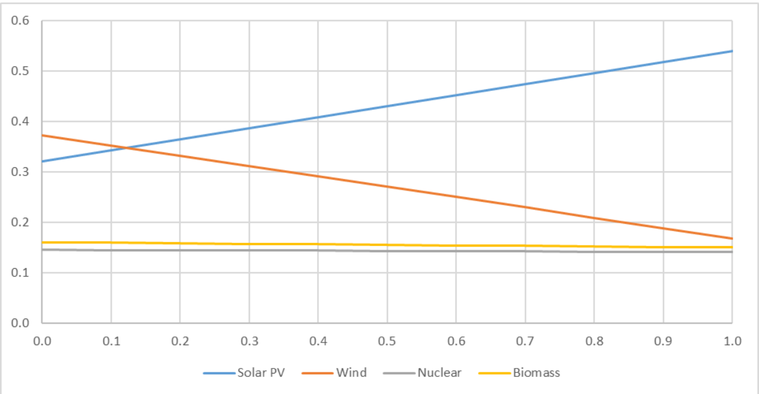
## Sub-Criterion: Feed-in-Tariff



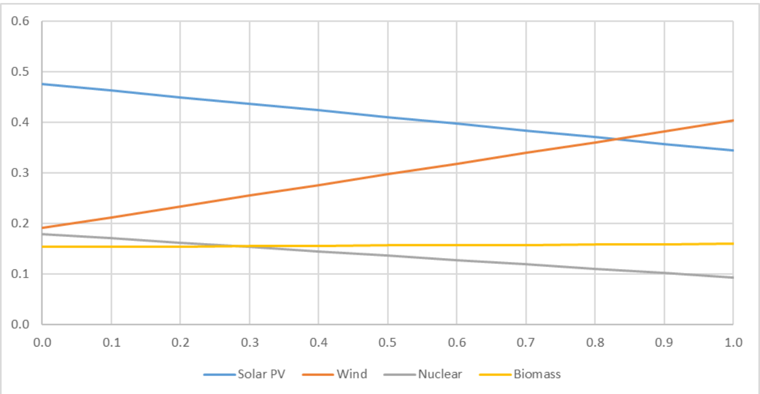

## Sub-Criterion: CO2 emission



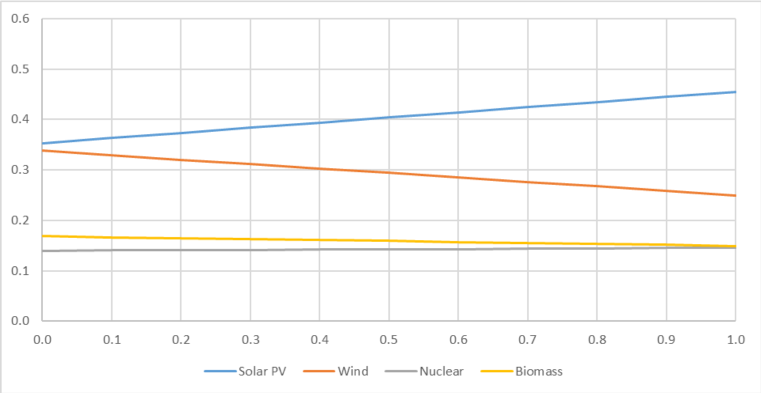

## Sub-Criterion: Land use



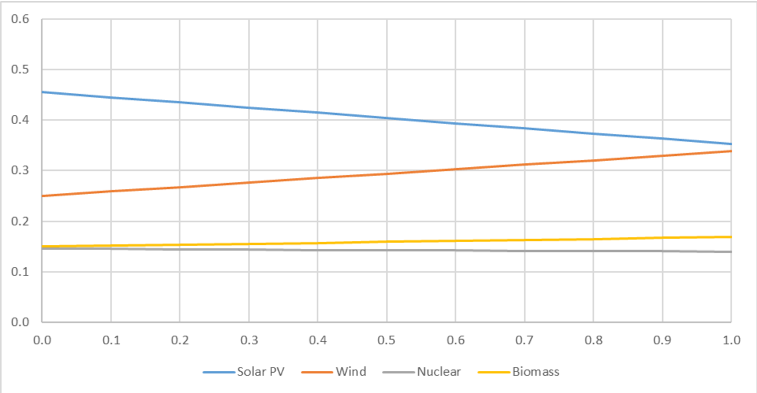

## Sub-Criterion: Water consumption



## Sub-Criterion: Job creation




## Sub-Criterion: Projected capacity



# 7 The Rating Method in AHP

## 7.1 Evaluating Alternatives with Ratings

- In the standard AHP method, the alternatives at the lowest level of the hierarchy are pairwise compared with respect to each of the leaf criteria. When the number of alternatives is large, it may not be practical as the resulting weights will all be very small. The Rating Method may be used instead.

- In the rating approach, a hierarchy is developed in the usual way down to the level of criteria or sub-criteria.

- The criteria and sub-criteria are prioritized in the usual way and their weights are expressed in **Normalized** or **Distributive Form**.

- Each of the leaf criteria is then given set of **Intensity** or **Performance Ratings** with respect to that criterion.

  Examples: *Excellent, Good, Average, Poor, Very poor.*
  *Very high, High, Average, Low*, *Very low*.

- The type and number of intensity/performance ratings for each criterion may be different.

- The performance ratings for each criterion are then prioritized by pairwise comparisons to determine their relative importance with respect to the criterion they are measuring. The weights of these performance ratings are expressed in **Idealized Form**.

- The alternatives or candidates are independently evaluated one at a time in terms of rating intensities for each of the criteria.

- The global weight for each alternative is then evaluated in the usual additive weighted sum manner.

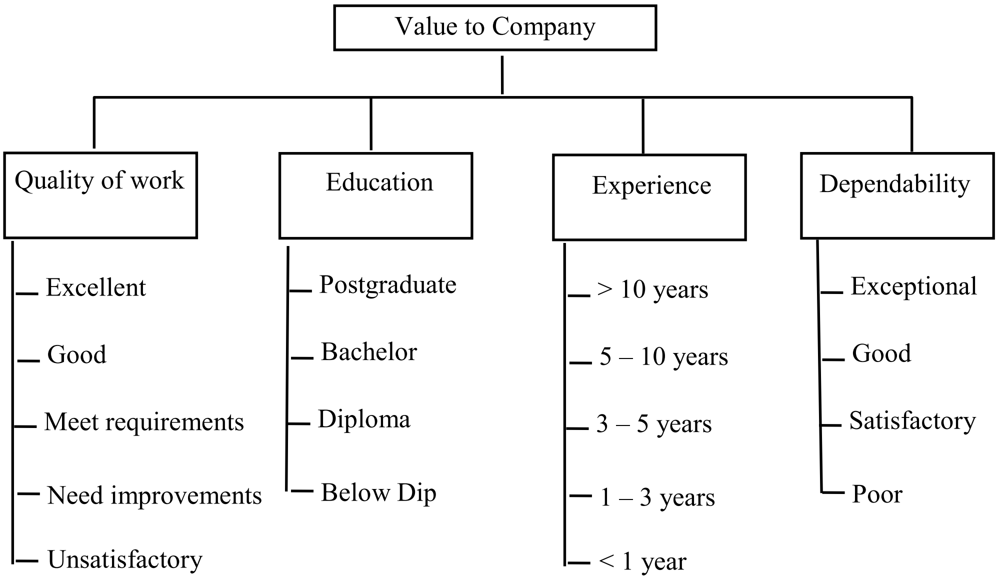
## 7.2 Case Study: Evaluation of Employees

- A firm would like to evaluate the value of its employees to the company.

- The company consider these four criteria contributing to the goal:

  1. Quality of Work
  2. Education
  3. Experience
  4. Dependability

- Each of the four criteria is measured using performance ratings as shown below:




- The four criteria are pairwise compared with respect to the Goal.

| | Quality of work | Education | Experience | Dependability | Local Weight |
|---|---|---|---|---|---|
| Quality of work | 1 | 5 | 7 | 3 | 0.565009 |
| Education | 1/5 | 1 | 3 | 1/3 | 0.117504 |
| Experience | 1/7 | 3 | 1 | 1/5 | 0.055285 |
| Dependability | 1/3 | 3 | 5 | 1 | 0.262201 |

$\lambda = 4.117$ CI = 0.03899 CR = 0.04333 < 0.1 $\boldsymbol{w}$ = [0.565009, 0.117504, 0.055285, 0.262201]

- The weights for the four criteria are: 0.565009, 0.117504, 0.055285, and 0.262201, respectively.

- The managers then pairwise compare the performance ratings or intensities according to the priority with respect to their parent criterion or sub-criterion.

- For examples, the managers might ask:

  **(*a*)** With respect to "Education", how important to the company is an employee with a post-graduate degree is with respect to an employee with only a bachelor degree.

  The assessments would be different if education is applied to waiters in a restaurant than to researchers in a technology driven company.

  **(*b*)** With respect to "Dependability", how important to the company is an employee who is rated "exceptional" compared with an employee who is rated "average"?

  The assessments would be different if dependability is applied to waiters in a restaurant than to captains of a passenger airplane.

- For the criterion “Quality of Work”, the following matrix was obtained:

| **Criterion: “Quality of Work”** | Excellent | Good | Meet Requirements | Need Improvements | Unsatisfactory | Local weight |
|---|---|---|---|---|---|---|
| Excellent | 1 | 1 | 5 | 7 | 9 | 0.428747 |
| Good | 1 | 1 | 3 | 5 | 7 | 0.337852 |
| Meet Requirements | 1/5 | 1/3 | 1 | 3 | 5 | 0.136310 |
| Need Improvements | 1/7 | 1/5 | 1/3 | 1 | 2 | 0.060049 |
| Unsatisfactory | 1/9 | 1/7 | 1/5 | 1/2 | 1 | 0.037042 |

$\lambda = 5.1356$, CI = 0.03390, CR = 0.03027 < 0.1 $\quad$ $\boldsymbol{w}$ = [0.428747, 0.337852, 0.136310, 0.060049. 0.037042]

- The weights of the ratings **idealized form**:

| Rating under Quality of Work | Weight in Ideal Form |
|---|---|
| Excellent | 1 |
| Good | 0.787997 |
| Meet Requirements | 0.317927 |
| Need Improvements | 0.140058 |
| Unsatisfactory | 0.086395 |

- For the criterion “Education”, the following matrix was obtained:

| **Criterion: “Education”** | Postgraduate | Bachelor | Diploma | Below Dip | Local weight |
|---|---|---|---|---|---|
| Postgraduate | 1 | 3 | 5 | 9 | 0.573455 |
| Bachelor | 1/3 | 1 | 3 | 7 | 0.271227 |
| Diploma | 1/5 | 1/3 | 1 | 3 | 0.110233 |
| Below Dip | 1/9 | 1/7 | 1/3 | 1 | 0.045086 |

$\lambda = 4.0876$, CI = 0.02921, CR = 0.0325 < 0.1

- The weights of the ratings **idealized form**:

| Rating under Education | Weight in Ideal Form |
|---|---|
| Postgraduate | 1 |
| Bachelor | 0.472971 |
| Diploma | 0.192225 |
| Below Dip | 0.078621 |

- For the criterion “Experience”, the following matrix was obtained:

| **Criterion: “Experience”** | > 10 years | 5 - 10 years | 3 - 5 years | 1 - 3 years | < 1 year | Local weight |
|---|---|---|---|---|---|---|
| > 10 years | 1 | 1 | 3 | 5 | 7 | 0.393131 |
| 5 - 10 years | 1 | 1 | 2 | 3 | 5 | 0.304538 |
| 3 - 5 years | 1/3 | 1/2 | 1 | 2 | 3 | 0.153971 |
| 1 - 3 years | 1/5 | 1/3 | 1/2 | 1 | 3 | 0.099081 |
| < 1 year | 1/7 | 1/5 | 1/3 | 1/3 | 1 | 0.049280 |

$\lambda = 5.0872$, CI = 0.02181, CR = 0.01947 < 0.1

- The weights of the ratings **idealized form**:

| Rating under Experience | Weight in Ideal Form |
|---|---|
| > 10 years | 1 |
| 5 - 10 years | 0.774648 |
| 3 - 5 years | 0.391653 |
| 1 - 3 years | 0.252030 |
| < 1 year | 0.125353 |

- For the criterion "dependability", the following matrix was obtained:

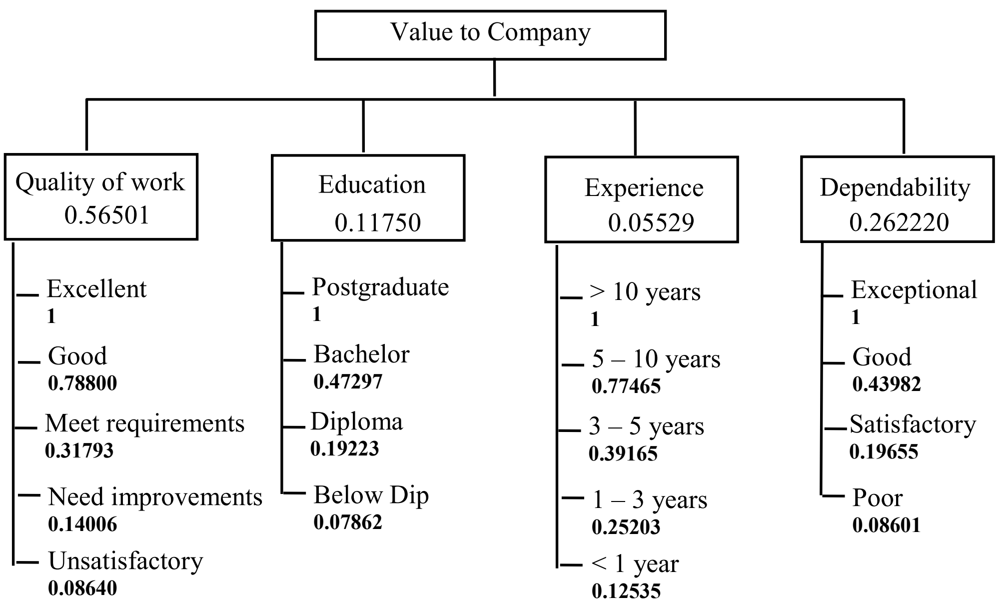
| **Criterion: "Dependability"** | Exceptional | Good | Satisfactory | Poor | Normalized weight |
|---|---|---|---|---|---|
| Exceptional | 1 | 3 | 5 | 9 | 0.58059 |
| Good | 1/3 | 1 | 3 | 5 | 0.25536 |
| Satisfactory | 1/5 | 1/3 | 1 | 3 | 0.11411 |
| Poor | 1/9 | 1/5 | 1/3 | 1 | 0.04994 |

$\lambda = 4.0763$, CI = 0.02543, CR = 0.02836 < 0.1

- The weights of the ratings **idealized form**:

| Rating under Dependability | Weight in Ideal Form |
|---|---|
| Exceptional | 1 |
| Good | 0.43983 |
| Satisfactory | 0.19655 |
| Poor | 0.08601 |

- The hierarchy with criteria weights and ratings' weights:



- Finally, the managers rate each individual employee by assigning the intensity rating that applies to him or her under each criterion.

| | **Candidate** | 0.56501 | 0.11750 | 0.05529 | 0.26220 |
|---|---|---|---|---|---|
| | | Rating for Quality | Rating for Education | Rating for Experience | Rating for Dependability |
| 1 | John Lim | Good | Postgraduate | 3 - 5 years | Satisfactory |
| 2 | Tan Ah Huay | Meet Requirements | Diploma | > 10 years | Good |
| 3 | Chow Ah Beng | Need Improvements | Below Dip | 1 -3 years | Poor |
| 4 | Mary Lau | Good | Bachelor | < 1 year | Exceptional |
| 5 | Harry Lee | Excellent | Diploma | 5 – 10 years | Good |
| 6 | | | | | |
| 7 | | | | | |

- The Overall Rating of each employee with respect to the Goal is the criteria-weighted sum of their individual ratings.

| | Candidate | 0.56501 | 0.11750 | 0.05529 | 0.26220 | 1.0000 |
|---|---|---|---|---|---|---|
| | | Rating for Quality | Rating for Education | Rating for Experience | Rating for Dependability | **Overall Rating** |
| 1 | John Lim | 0.78800 | 1.00000 | 0.39165 | 0.19655 | 0.63592 |
| 2 | Tan Ah Huay | 0.31793 | 0.19223 | 1.00000 | 0.43982 | 0.37283 |
| 3 | Chow Ah Beng | 0.14006 | 0.07862 | 0.25203 | 0.08601 | 0.12486 |
| 4 | Mary Lau | 0.78800 | 0.47297 | 0.12535 | 1.00000 | 0.76993 |
| 5 | Harry Lee | 1.00000 | 0.19223 | 0.77465 | 0.43982 | 0.74575 |
| 6 | | | | | | |
| 7 | | | | | | |