Multiple fuzzy Sugeno λ-measures in networks. An application.

Inmaculada Gutiérrez* [a] Daniel Gómez[a,b] Javier Castro[a,b]

Rosa Espínola[a,b]

[a] Facultad de Estudios Estadísticos, Universidad Complutense de Madrid, Avenida Puerta de Hierro s/n, 28040, Madrid

[b]Instituto de Evaluación Sanitaria, Complutense University, Madrid

January 31, 2021

In this document we set some information to facilitate the follow-up of the paper entitled *Multiple fuzzy Sugeno λ-measures in networks. An application*. Particularly, we detail the generation processes considered to prepare the benchmark models which are later used to test the proposed methodology. Each benchmark graph represents a multi-dimensional extended vector fuzzy graph. On the one hand, w explain how we generate the adjacency matrix. Then we detail the calculation process of the trapezoidal fuzzy numbers which set the basis of the generation of the synergies matrix.

1 Adjacency matrix

The adjacency matrix A is randomly generated according to equation (1) for a set V with 256 nodes, by considering the different combinations of the values of the parameters α and β regarding the input/output values (z_{in} and z_{out}), showed in Table 1. These parameters regulate the density of the connections matrix, A. The process to generate this adjacency matrix is showed in the Algorithm 1.

$$P(i,j) = \begin{cases} \alpha & \text{if} \quad i,j \in C_k \\ \beta & \text{if} \quad \text{otherwise} \end{cases} \tag{1}$$

	Network 1	Network 2	Network 3	Network 4	Network 5	Network 6	Network 7	Network 8	Network 9
α	0.45	0.4	0.35	0.325	0.3	0.275	0.25	0.225	0.2
β	0.016	0.033	0.05	0.058	0.066	0.075	0.083	0.091	0.1

Table 1: Parameters used to generate the adjacency matrix A of each model.

Algorithm 1 *Generate Adjacency*

Input: $(|C_1|, \ldots |C_r|), \alpha, \beta, n$;
Output: A;
$A(i,j) \leftarrow 0, \forall i,j = 1, \ldots, n$;
for $(i = 1)$ **to** (n) **do**
  for $(i = 1)$ **to** (n) **do**
    for $(\ell = 1)$ **to** (r) **do**
      $\varepsilon \leftarrow \text{rand}(0,1)$;
      if $(|C_{\ell-1}| < i \leq |C_\ell|)$ and $(|C_{\ell-1}| < j \leq |C_\ell|)$ **then**
        if $\varepsilon < \alpha$ **then**
          $A(i,j) \leftarrow 1$;
        end if
      else
        if $\varepsilon < \beta$ **then**
          $A(i,j) \leftarrow 1$;
        end if
      end if
    end for
  end for
end for
return(A);

2 Low fuzzy number generation

This type of fuzzy numbers, showed in the Figure 1, are generated to represent, in each vector, the components related to the elements with a *low* value in the characteristic of the corresponding vector.

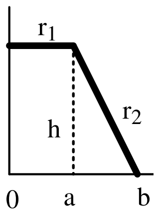

Figure 1: *Low* trapezoidal fuzzy number.

Fixed the values a and b, we can calculate the equations of the lines r_1 and r_2 so that the value of the area of the trapeze under them is 1. Particularly, the line r_1 is defined as $y = h$, where h is a value chosen so that the value of the corresponding integral is 1:

$$1 = ah + \left(\frac{b-a}{2}\right) h \Longrightarrow h = \frac{2}{a+b}$$

On the other hand, r_2 is the line trough the points $\left(\frac{2}{a+b}, a\right)$ and $(0, b)$, so we obtain this system:

$$r_2 = \begin{cases} \frac{2}{a+b} = \alpha + \beta a \\ 0 = \alpha + \beta b \end{cases}$$

By isolating α and β, it holds $\alpha = \frac{-2b}{(a+b)(a-b)}$ and $\beta = \frac{2}{(a+b)(a-b)}$, so the distribution function is:

$$F(x) = \begin{cases} \frac{2x}{a+b}, & x \in [0, a] \\ \frac{2a}{a+b} + \int_a^x \left(\frac{-2b}{(a+b)(a-b)} + \frac{2z}{(a+b)(a-b)}\right) dz, & x \in (a, b] \end{cases}$$

where $\frac{2a}{a+b} + \int_a^x \left(\frac{-2b}{(a+b)(a-b)} + \frac{2z}{(a+b)(a-b)}\right) dz = \frac{2a}{a+b} + \int_a^x \left(\frac{2(z-b)}{(a+b)(a-b)}\right) dz = \frac{2a}{a+b} + \left[\frac{2(z-b)^2}{(a+b)(a-b)2}\right]_a^x = \frac{2a}{a+b} + \frac{(x-b)^2-(a-b)^2}{(a+b)(a-b)}$.

Once the *low* fuzzy number is characterized, on the following denoted by ℓ, we apply the inverse method. First, we have to calculate the inverse function of F, $F^{-1}(x)$; then we simulate a value between 0 and 1 (p). Finally, $F^{-1}(p)$ is the value assigned to and edge between nodes which are not in the same community.

- If $p \leq \frac{2a}{a+b} \Longrightarrow p = \frac{2x}{a+b} \Longrightarrow x = \frac{(a+b)p}{2}$
- If $p > \frac{2a}{a+b} \Longrightarrow p = \frac{2a}{a+b} + \frac{(x-b)^2-(a-b)^2}{(a+b)(a-b)}$
  $\Longrightarrow x = b - \sqrt{\left(p - \frac{2a}{a+b}\right)(a+b)(a-b) + (a-b)^2}$
  We take the sign '–' because $x - b < 0$.

Hence, if $p = \text{randUni}(0, 1)$ is a random value obtained from an uniform distribution $U(0, 1)$, the *low* values consideNetwork in the simulation are obtained according to the following equation.

$$\begin{cases} x = \frac{(a+b)p}{2} & \text{if } p \leq \frac{2a}{a+b} \\ x = b - \sqrt{\left(p - \frac{2a}{a+b}\right)(a+b)(a-b) + (a-b)^2} & \text{otherwise} \end{cases}$$

This process is summarized in the Algorithm 2.

Algorithm 2 *Low Fuzzy Number*

Input: a, b;
Output: ℓ;
$p \leftarrow \text{rand}(0, 1)$;
if $p \leq \frac{2a}{a+b}$ **then**
  $\ell \leftarrow \frac{(a+b)p}{2}$;
else
  $\ell \leftarrow b - \sqrt{\left(p - \frac{2a}{a+b}\right)(a+b)(a-b) + (a+b)^2}$;
end if
return(ℓ);

3 High fuzzy number generation

This type of fuzzy numbers, showed in the Figure 2, are generated to represent, in each vector, the components related to the elements with a *high* value in the characteristic of the corresponding vector.

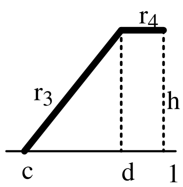

Figure 2: *High* trapezoidal fuzzy number.

Fixed the values c and d, we can calculate the equations of the lines r_3 and r_4 so that the area of the trapeze under them is 1. Particularly, the line r_3 is defined as $y = h$, where h is a value chosen so that the value of the corersponing integral is 1:

$$1 = \frac{(d-c)h}{2}(1-d)h \Longrightarrow h = \frac{2}{(1-d)+(1-c)}$$

On the other hand, r_4 is the line trough the points $\left(\frac{2}{(1-d)+(1-c)}, d\right)$ and $(0, c)$, so we obtain this system:

$$r_4 = \begin{cases} \frac{2}{(1-d)+(1-c)} = \alpha + \beta d \\ 0 = \alpha + \beta c \end{cases}$$

By isolating α and β, it holds $\alpha = \frac{-2c}{[(1-d)+(1-c)](d-c)}$ and $\beta = \frac{2}{[(1-d)+(1-c)](d-c)}$, so the distribution function is:

$$F(x) = \begin{cases} \int_c^x \left(\frac{-2c+2z}{[(1-d)+(1-c)](d-c)}\right) dz, & x \in [c,d] \\ \int_c^d \left(\frac{-2c+2z}{[(1-d)+(1-c)](d-c)}\right) dz + \int_{z=d}^{z=x} \frac{2}{(1-d)+(1-c)}, & x \in (d,1] \end{cases}$$

where

- $\int_c^x \left(\frac{-2c+2z}{[(1-d)+(1-c)](d-c)}\right) dz = \frac{\left[(z-c)^2\right]_c^x}{[(1-d)+(1-c)](d-c)} = \frac{(x-c)^2}{[(1-d)+(1-c)](d-c)}$
- $\int_c^d \left(\frac{-2c+2z}{[(1-d)+(1-c)](d-c)}\right) dz + \int_d^x \frac{2}{(1-d)+(1-c)} = \frac{\left[(z-c)^2\right]_c^d}{[(1-d)+(1-c)](d-c)} + \frac{2[z]_{z=d}^{z=x}}{(1-d)+(1-c)} = \frac{(d-c)^2}{[(1-d)+(1-c)](d-c)} + \frac{2(x-d)}{(1-d)+(1-c)} = \frac{(x-d)+(x-c)}{(1-d)+(1-c)}$

As it is described concerning *low* fuzzy numbers, we apply the inverse method to simulate the values of the *high* fuzzy numbers ($\hbar$ on the following). Hence, the value $F^{-1}(p)$ is:

- If $p \le \frac{d-c}{(1-d)+(1-c)} \Longrightarrow p = \frac{(x-c)^2}{[(1-d)+(1-c)](d-c)} \Longrightarrow x = c + \sqrt{p(d-c)[(1-d)+(1-c)]}$
- If $p > \frac{d-c}{(1-d)+(1-c)} \Longrightarrow p = \frac{(x-d)+(x-c)}{(1-d)+(1-c)} \Longrightarrow p = \frac{(x-d)+(x-c)}{(1-d)+(1-c)} \Longrightarrow x = \frac{p[(1-d)+(1-c)+d+c]}{2}$

  We take the sign '+' because $x - d > 0$.

Hence, if $p = \text{randUni}(0,1)$ is a random value obtained from an uniform distribution $U(0,1)$, the *high* values considered in the simulation are obtained according to the following equation.
$$\begin{cases} x = c + \sqrt{p(d-c)[(1-c)+(1-d)]} & \text{if } p \le \frac{d-c}{(1-c)+(1-d)} \\ x = \frac{p[(1-c)+(1-d)]+c+d}{2} & \text{otherwise} \end{cases}$$
This process is summarized in the Algorithm 3.

Algorithm 3 *High Fuzzy Number*

Input: c, d;
Output: $\hbar$;
$p \leftarrow \text{rand}(0,1)$;
if $\left(p \le \frac{d-c}{(1-c)+(1-d)}\right)$ **then**
  $\hbar \leftarrow c + \sqrt{p(d-c)((1-c)+(1-d))}$;
else
  $\hbar \leftarrow \frac{p((1-c)+(1-d))+c+d}{2}$;
end if
return($\hbar$);

4 Generate multiple vectors

For each vector, the component related to the nodes which are in the same community, are generated as *high* fuzzy numbers, whereas the components related to nodes of different communities are generated as *low* fuzzy numbers. In each benchmark model, we have r vectors as starting point, where r is the amount of communities embedded in the synergies matrix, $\mathbb{X}$. Each vector is associated with a community C_i, so that nodes belonging to C_i will have a *high* value in x^i, whereas the nodes which are not in C_i will have a *low* value in x^i. ($x_j^i = \hbar$, if $j \in C_i$; $x_j^i = \ell$, if $j \notin C_i$). Different combinations of the parameters a, b, c y d are considered to generate the *low*/*high* fuzzy numbers. These combinations affect to the scattering of the ℓ and $\hbar$ fuzzy numbers. The process is summarized in the Algorithm 4.

Algorithm 4 *Generate Multiple Vectors*

Input: $(|C_1|, \ldots |C_r|), a, b, c, d$;
Output: multipleVectors;
$|C_0| \leftarrow 0$;
multipleVectors $\leftarrow 0$; (matrix $r \times n$, the line ℓ represents the vector x^ℓ)
for $(\ell = 1)$ **to** (r) **do**
  for (i = 1) **to** (n) **do**
    if $|C_{\ell-1}| < i \le |C_\ell|$ **then**
      multipleVectors$(\ell, i) \leftarrow$ HighFuzzyNumber(c, d);
    else
      multipleVectors$(\ell, i) \leftarrow$ LowFuzzyNumber(a, b);
    end if
  end for
end for
return(multipleVectors);

5 Synergies matrix

From the family of vectors previously generated with the Algorithm *Generate Multiple Vectors*, we obtain the family of fuzzy Sugeno λ-measures $\left(\mu^{a}_{X^1}, \ldots, \mu^{a}_{X^r}\right)$. We consider the matrices $\left(X^1, \ldots, X^r\right)$, the adjacency of the corresponding MAWG. The second component of every benchmark model considered is related to an aggregation of these matrices, $\mathbb{X} = \Phi\left(X^1, \ldots, X^r\right) = \max\left(X^1, \ldots, X^r\right)$. We summarize this process in the Algorithm 5.

Algorithm 5 *Matrix From Multiple Vectors*

Input: $(|C_1|, \ldots |C_r|), a, b, c, d$;
Output: $\mathbb{X}$;
$\mathrm{multipleVectors} \leftarrow \mathrm{GenerateMultipleVectors}((|C_1|, \ldots |C_r|), a, b, c, d)$;
for $(\ell = 1)$ **to** (r) **do**
  for $(i = 1)$ **to** (n) **do**
    $\mathrm{Sh}(\ell, i) \leftarrow \frac{\mathrm{multipleVectors}(\ell,i)}{\sum_{k=1}^{n} \mathrm{multipleVectors}(\ell,k)}$;
    for $(j = 1)$ **to** (n) **do**
      $\mathrm{Sh}_j(\ell, i) \leftarrow \frac{\mathrm{multipleVectors}(\ell,i)}{\sum^{n}_{\substack{k \neq i \\ k \in V}} \mathrm{multipleVectors}(\ell,k)}$;
    end for
  end for
end for
for $(\ell = 1)$ **to** (r) **do**
  for $(i = 1)$ **to** (n) **do**
    for $(j = 1)$ **to** (n) **do**
      $X^{\ell}(i,j) \leftarrow \min\{|\mathrm{Sh}(\ell,i) - \mathrm{Sh}_j(\ell,i)|, |\mathrm{Sh}(\ell,j) - \mathrm{Sh}_i(\ell,j)|\}$;
    end for
  end for
end for
$\mathbb{X} \leftarrow \max\{X^1, \ldots, X^r\}$;
return$(\mathbb{X})$;