16th World Congress of the International Fuzzy Systems Association (IFSA)
9th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT)

Algorithm for simultaneous defuzzification under constraints: shifted mean-max

Jörg Verstraete[1]

[1]jorg.verstraete@ibspan.waw.pl
[1]Systems Research Institute, Polish Academy of Sciences, ul. Newelska 6, 00-375 Warszawa, Poland

Abstract

In this article, we present an algorithm for defuzzifying multiple fuzzy sets simultaneously, where the defuzzified values are bound by a constraint. The algorithm aims at maximizing the lowest membership grade of the defuzzified values in each fuzzy set, while satisfying the constraint. In the examples, MeanOfMax will be used as a starting point, while the constraint is that the sum of the defuzzified values has to match a given crisp value. The article describes the algorithm, experiments and evaluation of the outcome.

Keywords: defuzzification, spatial disaggregation, constrained defuzzification

1. Introduction

Fuzzy sets represent imprecise or uncertain information, allowing for the representation of a set of possible values with an associated possibility for each of these values. Many computations can be performed and conclusions can be drawn, but very often it is necessary to extract a single crisp value that best approximates the fuzzy set. This can be either for use in future calculations using a traditional system or just for providing an estimate that is easier to comprehend (but is less informative). The representation of a fuzzy set to a single value is done using appropriate defuzzification methods, where a crisp value deemed representative for a fuzzy set is determined. In current methods, this is done by giving priority to a specific desired aspect of the defuzzification and then applying the most appropriate defuzzifier: should the defuzzified value have the highest occurring membership grade, or is it more important that it reflects the shape of the fuzzy set? In the first case, MeanOfMax (i.e. the mean of the elements with highest occurring membershipgrade) can for instance be chosen, in the second case CenterOfGravity (i.e. the value associated with center of gravity of the fuzzy set) may be more appropriate, but additional considerations and defuzzifiers are possible. In [1], additional constraints on the domain of the fuzzy set are considered, which allows for further control over the defuzzified value. These constraints still do not incorporate constraints imposed by the not yet determined defuzzified value of other fuzzy sets. To our knowledge, the problem of combined defuzzification, in which the defuzzified values together have to satisfy a constraint, has not been considered before.

In this article, additional constraints for the defuzzification are introduced. Consider the situation where there are multiple fuzzy sets, but their defuzzified values should be constrained by each other. In our specific application, explained in section 2, a number of fuzzy sets need to be defuzzified while the sum of their defuzzified values should equal a given, crisp value. To find an optimal solution for this problem, it is not possible to apply an algorithm to each fuzzy set individually, but it necessitates an algorithm that considers all sets simultaneously. Similar to criteria that exist for defuzzification methods, different criteria will need to be considered and evaluated. In this article, the modification of the criteria and an algorithm using MeanOfMax as a basis for a combined defuzzification, is presented. In the article, only the constraint of the sum is considered, as this is the problem that occurred in our application (section 2). Other constraints may require a different solution method. The presented methods starts with finding the MeanOfMax of the fuzzy sets considered and then calculates a correction for each of them to yield defuzzified values that meet the constraint. The correction is calculated taking into account the shape of the fuzzy set, in order to maximize the lowest membership grade of the resulting defuzzified value in its respective fuzzy set. By taking into account the shape of the fuzzy sets, it is possible to derive a globally better solution than what would be obtained by rescaling the crisp values. The problem of combined defuzzification generally occurs when a method yielding fuzzy solutions is used to find multiple crisp solutions, which are not independent of one-another. The presented method was developed for the specific problem of remapping gridded spatial data, as explained in the next section 2.

The subsequent section contains a short description on defuzzification, definitions and criteria; section 4 explains the proposed algorithm and criteria, after which conclusions are drawn in section 6.

2. Origin of the problem

The need for simultaneous defuzzification originated from the application of fuzzy inference systems in solving the map overlay problem, a specific spatial problem that occurs when combining or comparing raster data. A raster or grid is a commonly used structure to represent a numeric value over a region (e.g. concentration of a pollutant). The given input raster has to be transformed to match a different, given target raster, as this yields a one-to-one mapping of the grid cells, which in turn allows for both rasters to be compared or combined. The problem when using raster data is that the underlying spatial distribution of the numeric data is not known and needs to be either assumed or estimated. Figure 1 illustrates how different spatial distributions within a single grid cell can still yield the same value for the grid. Four different spatial distributions are depicted, ranging from a single point source to an area source. At the level of the grid cell, each of these four cases is represented by the same grid cell: one cell with an associated value of 100. Comparing raster data is frequently needed

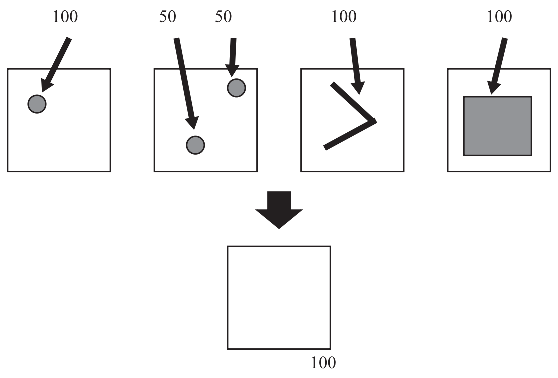

Figure 1: The problem with raster data sources: different underlying distributions can yield the same raster. The examples show: a single point source, two point sources, a line source and an area source.

when analysing and/or combining spatial data. As the data tends to come from different sources and is obtained using different techniques and models, the rasters on which the data are provided do not match: there is no one-to-one mapping between cells of the rasters. This makes a direct comparison impossible, and requires that one raster has to be remapped onto the other. Different approaches to this problem are considered, and presented in [2]. Current methods tend to use areal weighting ([3]), as this is the easiest approach. However, it assumes that the data is uniformly distributed over each individual grid cell. As the above illustration shows, this may not be the case, and there may be many different underlying distributions. Areal smoothing is a second approach where the spatial distribution is assumed to be smooth; the remapping is achieved by treating the numeric data as a third dimension, fitting a smooth surface over data and resampling the three dimensional structure. Spatial regression methods start from statistical assumptions on the distribution of the data, but the results of remapping highly depends on correct assumptions and fail when the spatial distributions are erratic and difficult to approximate. Independent of remapping, research has been done to combine data from different datasets, the authors in [4] combine linguistically annotated data from different sources into a better quality dataset.

In a new proposed approach presented in [5], additional knowledge (e.g. in the case of a pollutant stemming from car emissions, this can be road traffic information) is used to estimate the underlying spatial distribution. This additional information is used in a fuzzy inference system, which intelligently remaps the give raster onto a target raster, using the additional information to estimate the underlying distribution. The first step is to divide each cell of the raster into segments, which are defined as the intersections of input and target: this has the benefit that each cell in the input raster is partitioned in a number of segments, but as the target grid has the same property, the segments can also be recombined to form the target grid. This is illustrated on figure 2, where the input grid is the leftmost grid and the target grid on which the data should be remapped is the rightmost grid. For each cell in the input grid, the data is redistributed over the segments that are obtained from intersecting the input grid with the target grid. This is illustrated using the shaded cell in the input grid. Once a value for each segment is obtained, the segments can be recombined into the target grid, as shown using the highlighted cell in the target grid.

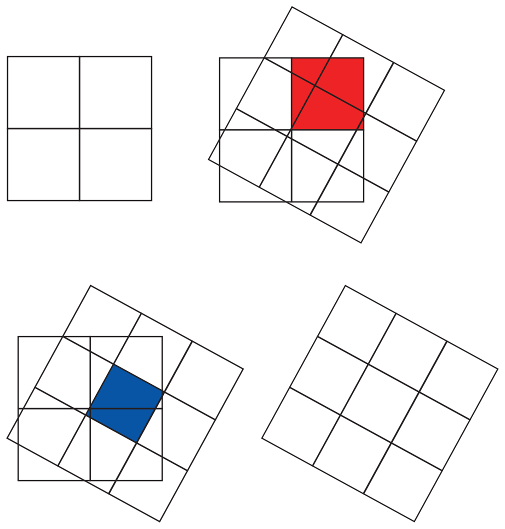

Figure 2: Stages in the remapping process: the input grid (top left), redistrubition of the data in each grid cell into segments (top right), recombination of segments into a target cell (bottom left), the target and output grid (bottom right).

Each input cell models a numeric value (e.g. concentration of a pollutant). Finding the underlying distribution means finding a way to distribute this value over the segments of the cell. However, the total value modelled inside the input cell should not change, resulting in the constraint that the sum of the values of the contained segments is the value of the input cell. As the new value for each segment is obtained through the application of a fuzzy inference system, a fuzzy set will be associated with each segment. A simple approach is to apply classic defuzzification methods on each segment individually, and rescale the values; this was the method used in [6]. While this does not take into account the fuzzy sets, the result were acceptable, as in many cases, the sum of the defuzzified values is not too different from the value after rescaling. However, the rescaling shifts the defuzzified values somewhat, and using a custom simultaneous defuzzification may allow for a better result in general.

In the considered problem for grid remapping, it would be natural for the extreme values to be more possible. Using the MeanOfMax as an initial estimate, combined with the presented algorithm yields defuzzifications that satisfy this.

3. Defuzzification

3.1. Definitions

Defuzzification is the process of reducing a fuzzy set to a single crisp value ([7]). Commonly, the only knowledge is the provided fuzzy set; all values of the domain of the fuzzy set are considered possible values. While it may seem natural to search the domain for a value that has the highest membership grade, arguments against this can be made in the case of e.g. very skewed fuzzy sets. In this case, a large portion of the knowledge (many values with sub-maximal membership) might be dismissed. To illustrate, compare e.g. a left skewed and a right skewed triangular possibility distribution ([8]) around a given number. Considering the value which has highest membership will - in both cases - yield the number, ignoring the distributions. Furthermore, there may be more than one value which has the highest membership grade, in which case one of these values need to be chosen. As there are different possibilities which are applicable in different cases, many defuzzification operations exist. Commonly used operations are CenterOfGravity and MeanOfMax ([9]). CenterOfGravity (COG) is defined as:

$$COG(\tilde{A}) = \frac{\sum_{x_{min}}^{x_{max}} x\mu_{\tilde{A}}(x)}{\sum_{x_{min}}^{x_{max}} \mu_{\tilde{A}}(x)} \tag{1}$$

CenterOfGravity considers the value that matches the center of gravity of the fuzzy set as the most representative value. This value not necessarily has the highest occurring membership grade, but does take into account the shape of the entire fuzzy set. MeanOfMax is defined as

$$MeOM(\tilde{A}) = \frac{\sum_{x \in \text{core}(\tilde{A})} x}{|\text{core}(\tilde{A})|} \tag{2}$$

Mean of max considers the mean value of all elements that have the highest membership grade. This also does not guarantee it has the highest occurring membership grade, and completely ignores the shape of the fuzzy set. Other possiblities that are somewhat similar to Mean of max are leftmost-max or rightmost-max (smallest/greatest domain value of those that have the highest membership grade). In [7], a number of defuzzifiers are listed and tested against objective criteria. Some of these criteria are summarized below, as they will be expanded to objectively judge the presented simultaneous defuzzification.

3.2. Criteria

In [7], a number of criteria by which a defuzzification operator can be judged, are presented. The authors consider different properties for different types of domains, from arbitrary universes to fuzzy quantities ([10]). In our application, only fuzzy quantities (fuzzy sets over the real domain) are considered, and the presented algorithm is also only considered or such fuzzy sets. The criteria from [7] that are appropriate here are summarized below:

1. Core selection or semantically correct defuzzification: the defuzzified value is a value with highest occurring membership grade.
2. Scale invariance: the scale of the domain does not influence the relative position of the defuzzified value (relative scale concerns translation of the fuzzy set, ratio scale concerns scaling of the unit, interval scale combines both relative and ratio scale).
3. Monotony: the defuzzified value of a fuzzy set that has greater values should be greater
4. x-Translation: if the fuzzy set is translated, the relative position of the defuzzified value should remain
5. x-Scaling: if domain values are multiplied with a constant factor, the position of the defuzzified value should remain
6. Continuity or robustness: a small change in membership grades should not yield a big change in defuzzified value.

Center of gravity for example, does not satisfy the core selection criteria (not even for non-convex fuzzy sets), as the method is not guaranteed to return an element with the highest occurring membership grade. It also does not satisfy any aspect of scale invariance. By contrast, MeanOfMax does not satisfy the core selection criteria in the event of non-convex fuzzy sets, but it is guaranteed to satisfy it if the fuzzy set is convex. It also satisfies all other criteria.

To develop the simultaneous defuzzification under constraints, an existing defuzzifier will be used as a starting point; in this article, MeanOfMax was chosen. The main arguments for choosing MeanOfMax are related to the application: the occurring fuzzy sets are known to be convex, and then it satisfies most criteria. Furthermore, Basic Defuzzification Distributions (BADD) reverts to MeanOfMax when the parameter γ, which can be seen as the confidence, is high. Similarly, Generalized Level Set Defuzzification (GLSD) also reverts to Mean of max for high confidence. In the application, experiments have shown that in most cases, a solution can be found when only considering the cores of the fuzzy sets, which increases our subjective confidence that a solution using values from the cores is a good solution. In the case of asymmetric fuzzy sets, such as those that represent the lowest values or the highest values of the domain, MeanOfMax tends to get closer to the extreme values than CenterOfGravity, which in our application, is a desired behaviour. An additional benefit of using MeanOfMax is the efficiency of computation.

3.3. Variants on defuzzification

In, [1] and [11], defuzzification of a fuzzy set under specific constraints was proposed. The constraints considered here were constraints imposed on the domain of the fuzzy set, limiting the possibilities for defuzzification. While this allows for further adjustment of the behaviour of defuzzifiers, it still considers defuzzification of a single fuzzy set at a time. In the current problem, this would still be too limited: the constraints imposed are still independent of the other fuzzy sets. When the contstraints involve multiple fuzzy sets, treating the fuzzy sets individually is no longer an option.

4. Simultaneous defuzzification

4.1. Concept

The aforementioned defuzzifications work on a single fuzzy set. The concept of the simultaneous defuzzification is that initially, each fuzzy set is defuzzified, in our case using MeanOfMax, after which the different defuzzified values are corrected in order to meet the criterion, in our case the sum of the defuzzified values is given.

For a given number of fuzzy sets, the aim is to find a defuzzification that chooses *best possible* crisp values so that the condition is satisfied. The given data constitutes of:

- n fuzzy sets over the real domain, $\tilde{A}_i : i = 1..n$. We currently limit ourselves to our specific problem where all fuzzy sets are known to be continuous (piecewise linear) and convex.
- total value T to which the defuzzified output should sum up

From a mathematical point of view, the problem is an optimization problem, where values $x_i : i = 1..n$ need to found such that the following conditions are met:

$$\sum_i x_i = T \tag{3}$$

$$\mu_{\tilde{A}_i}(x_i) = \text{as good as possible}, \forall i \tag{4}$$

where x_i are the defuzzifications of the different fuzzy sets $\tilde{A}_i$ and t is the given value. What constitutes *as good as possible* is a similar issue as to what constitutes a good defuzzifier in the classic sense: which criteria need to be considered and met. Some criteria for classic defuzzifiers are listed in section 3.2. In this article we considered maximization of the lowest membership grade as the main criterion. The reason for this is rather empirically established, but it also goes hand-in-hand with MeanOfMax. As MeanOfMax favours higher membership grades, the idea was that the combined defuzzifier derived from it should also favour higher membership grades. This means

$$\sum_i x_i = T \tag{5}$$

$$\min(\mu_{\tilde{A}_i}(x_i)) = \text{as high as possible}, \forall i \tag{6}$$

This not necessarily would result in a unique solution: increasing one x_i, while decreasing another $x_j, j \neq i$, might still keep the conditions satisfied and both solutions can have the same highest membership grade among the involved fuzzy sets. To overcome the problem that the solution is not necessarily unique, an additional criterion will be introduced. The algorithm starts from a known defuzzifier; the differences between the defuzzified value of each fuzzy set and the initial value of chosen defuzzifier for this fuzzy set should be as small as possible:

$$abs(x_i - d_i) = \text{as low as possible}, \forall i \tag{7}$$

where x_i is the final defuzzified value for fuzzy set $\tilde{A}_i$ and d_i is the value of the initial defuzzifier for fuzzy set $\tilde{A}_i$.

The developed algorithm is illustrated in pseudocode in 1. The first 7 lines help determine the starting point: for each fuzzy set $\tilde{(A)}_i$, the MeanOfMax (m_i), height (h_i) and the weak alpha cut $([c_i^l, c_i^r])$ at the height are calculated.

If the constraining value T is an element of $\sum_i [c_i^l, c_i^r]$, the corrected defuzzification for each fuzzy set can be immediately calculated by the formula:

$$x_j \leftarrow m_j + (T - \sum_i m_i)\frac{c_j^r - c_j^l}{\sum_i (c_i^r - c_i^l)} \tag{8}$$

In this case, the lowest occurring membership grade will be 1.

If however the constraining value T is not an element of $\sum_i [c_i^l, c_i^r]$, it means that it is not possible to satisfy the constraint while the lowest occurring membership grade will be 1. Lowering the

Algorithm 1 Simultaneous defuzzification using shifted Mean of max

```
for all i do
    $m_i \leftarrow \mathrm{MeOM}(\tilde{A}_i)$
    $h_i \leftarrow \mathrm{height}(\tilde{A}_i)$
                ▷ highest occuring membershipgrade
    $[c_i^l, c_i^r] \leftarrow \tilde{A}_{i_{h(A)_i}}$
                ▷ reverts to core for normalized $\tilde{A}_i$
end for
if T $\in \sum_i [c_i^l, c_i^r]$ then
    for all j do
        $x_j \leftarrow m_j + (T - \sum_i m_i)\frac{c_j^r - c_j^l}{\sum_i (c_i^r - c_i^l)}$
    end for
else
    $a_i^l \leftarrow c_i^l, a_i^r \leftarrow c_i^r$
    if $T < \sum_i c_i^l$ then
        repeat
            $c_i^l \leftarrow a_i^l$
            $\alpha \leftarrow \mathrm{findNextAlphaLeft}(A_i)$
            $]a_i^l, a_i^r[ = A_{i_{\overline{\alpha}}}$
        until T $\in \sum_i ]a_i^l, c_i^l[$
                ▷ Left shifts more left
                  Right becomes previous left
        for all j do
            $x_j \leftarrow c_j^l - (\sum_i c_i^l - T)\frac{a_j^l - c_j^l}{\sum_i (a_i^l - c_i^l)}$
        end for
    else if $T > \sum_i cr_i$ then
        repeat
            $c_i^r \leftarrow a_i^r$
            $\alpha \leftarrow \mathrm{findNextAlphaRight}(A_i)$
            $]a_i^l, a_i^r[ = A_{i_{\overline{\alpha}}}$
        until T $\in \sum_i ]c_i^r, a_i^r[$
                ▷ Right shifts more right
                  Left becomes previous right
        for all j do
            $x_j \leftarrow c_j^r + (T - \sum_i c_i^l)\frac{a_j^r - c_j^r}{\sum_i (a_i^r - c_i^r)}$
        end for
    end if
end if
```

alpha-level allows us to apply the algorithm on bigger intervals, which is what happens in the next steps. Notice that, the fuzzy sets are assumed to be convex, all α-levels are intervals. The functions *findNextAlphaLeft* and *findNextAlphaRight* are used to find the next α-value, going down from 1. The former will be used if the constraining value T is smaller than $\sum_i c_i^l$, the latter if T is greater than $\sum_i c_i^r$. The functions *findNextAlphaLeft* and *findNextAlphaRight* work on the same mechanism, looking respectively on left side of the initial interval or on the right side for the next breakpoint (the algorithm is optimized for piecewise fuzzy sets, for non-piecewise linear fuzzy sets, the alpha value can be lowered with a given constant, e.g. 0.1). The lowest possible value will be 0, in which case the support of the fuzzy sets are returned. No solution can be found if T is not contained in the sum or the supports. With the new alpha levels found, new intervals $]a_i^l, a_i^r[$ are associated. As soon as T is in the sum of these intervals, the best values for x_j can be calculated.

From the construction, the algorithm satisfies the criteria put forward: it maximizes the lowest membership grade and yields the solution with the shortest distances to MeanOfMax. To evaluate the algorithm, the criteria for defuzzifiers as mentioned in section 3.2 are reconsidered here.

1. Core selection: the algorithm will select elements from the core, if the constraint allows for this. In other words: if the given value T that constrains the sum of the defuzzified values is obtainable from values in the cores of the fuzzy set, all returned defuzzified values will belong the cores of their respective fuzzy sets.
2. Scale invariance: the algorithm exhibits scale invariance. Ratio scale is satisfied provided all fuzzy sets and the constraining value undergo the transformation. Relative scale is satisfied when the constraining value undergoes the sum of all the translations of the fuzzy sets involved. Interval scaling is satisfied under the combination of conditions for ratio and relative scale. This is explained through the fact that MeanOfMax exhibits scale invariance, and the operations in the algorithm are not scale dependent.
3. Monotony: if one fuzzy set is replaced by a fuzzy set that represents bigger values, while all other sets and the constraining value remain the same, monotony is not guaranteed. This is due to the fact that the values are adjusted in the last stage, and the amount of adjustment for the new fuzzy set can differ. Even if the constraining value also changes, it remains impossible to predict how this will affect the defuzzified value of one fuzzy set without further details on the shape of the fuzzy sets.
4. x-Translation: the algorithm satisfies x-Translation if the constraining value is translated over the sum of all the translations of the different fuzzy sets.
5. x-Scale: the algorithm satisfied x-Scale if the fuzzy sets and the constraining value are scaled with the same amount.

4.2. Examples

Consider the fuzzy sets A, B, C and D, with membership functions defined below and shown on figure 3.

$$
\begin{aligned}
\mu_{\tilde{A}} : \Re &\rightarrow [0,1] \\
\mu_{\tilde{A}}(x) &= \begin{cases} \frac{-1}{30}x + 1 & \forall x \in [0,30] \\ 0 & \text{elsewhere} \end{cases} \\
\mu_{\tilde{B}} : \Re &\rightarrow [0,1] \\
\mu_{\tilde{B}}(x) &= \begin{cases} \frac{1}{20}x & \forall x \in [0,20[\\ 1 & \forall x \in [20,40] \\ \frac{-1}{20}(x-40)+1 & \forall x \in]40,60] \\ 0 & \text{elsewhere} \end{cases} \\
\mu_{\tilde{C}} : \Re &\rightarrow [0,1] \\
\mu_{\tilde{C}}(x) &= \begin{cases} \frac{1}{40}(x-10) & \forall x \in [10,50[\\ 1 & \forall x \in [50,60] \\ \frac{-0.2}{20}(x-60)+1 & \forall x \in]60,80] \\ \frac{-0.8}{10}(x-80)+0.8 & \forall x \in]80,90] \\ 0 & \text{elsewhere} \end{cases} \\
\mu_{\tilde{C}} : \Re &\rightarrow [0,1] \\
\mu_{\tilde{C}}(x) &= \begin{cases} \frac{1}{30}(x-60) & \forall x \in [60,90[\\ 1 & \forall x \in [90,100] \end{cases}
\end{aligned}
$$

The algorithm works based on α levels at breakpoints. Due to the definition as convex fuzzy sets, all α levels of the sets in the example are intervals. The table 1 shows a number of these intervals for given α values. In a constraint defuzzification, it is obvious that situations can occur where the constraints are too limiting and would prevent any solution to be found. The last column in table 1 lists the supports of the involved fuzzy sets. The sum of the supports is $[70, 280]$, implying that if the desired sum of defuzzified values is outside of this interval, no solution will be found. Opposite, the sum of the kernels-intervals is $[160, 200]$. As the algorithm aims at maximizing the value of the lowest membership grade, it should result in satisfying constraining values from this interval while keeping the lowest membership equal to 1.

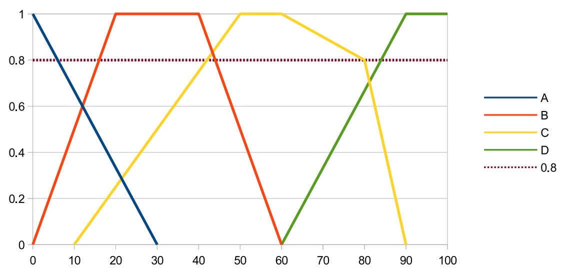

Figure 3: Fuzzy sets used in the example

To illustrate this, consider that the sum has to be 200. The centers of the core intervals sum up to: $0 + 30 + 55 + 95 = 180$. The difference with the target value is 20. The lengths of the intervals are 0, 20, 10, 10, with the total length of all core intervals equalling 40. Following the algorithm, the

	kernel	$\alpha = 0.8$	support
$\tilde{A}$	$[0, 0]$	$[0, 6]$	$[0, 30]$
$\tilde{B}$	$[20, 40]$	$[16, 44]$	$[0, 60]$
$\tilde{C}$	$[50, 60]$	$[42, 80]$	$[10, 90]$
$\tilde{D}$	$[90, 100]$	$[84, 100]$	$[60, 100]$
sum	$[160, 200]$	$[142, 230]$	$[70, 280]$
center	180	186	175

Table 1: Intervals

values are adjusted as follows:

$$
\begin{aligned}
x_A &= 0 + (200-180)\frac{0}{40} = 0 \\
x_B &= 30 + (200-180)\frac{20}{40} = 30 + 10 \\
x_C &= 55 + (200-180)\frac{10}{40} = 55 + 5 \\
x_D &= 95 + (200-180)\frac{10}{40} = 95 + 5
\end{aligned}
$$

The biggest intervals values are adjusted the most; the lowest occurring membership grade is 1. The calculations are similar for a constraining value of 160, and yield the smallest element of each core interval. In between, all elements values are adjusted linear in accordance with their length.

Consider now that the constrained value is decreased to 159; this constraint can no longer be solved using only values for the cores. The algorithm requires lowering of the α-level value. The function *findNextAlphaLevelLeft* scans from membership grade 1 (the starting value) downto 0, only considering the left sides of the fuzzy sets, and stops when it encounters a breakpoint, or reaches 0. In this case, 0 will be reached: the intervals in the column *support* in table 1 will be used; the formula in the algorithm (line 18) yields:

$$
\begin{aligned}
x_A &= 0 - (160-159)\frac{0-0}{70-160} = 0 \\
x_B &= 20 - (160-159)\frac{0-20}{70-160} = 19.78 \\
x_C &= 50 - (160-159)\frac{10-50}{70-160} = 49.56 \\
x_D &= 90 - (160-159)\frac{60-90}{70-160} = 89.67
\end{aligned}
$$

The defuzzified values are such that the smallest membership grade of each of the values in their respective fuzzy set is maximised. The amount of adjustment is inverse proportional to the steepness of the left side of the fuzzy set: B, which has the steepest left side adjusts the least; D, which has the least steep left side adjusts the most.The values continue to decrease linearly as the constraining value T is decreased; all the way till values match the minimum possible values of the respective supports.

The algorithm behaves similarly on the right side of the fuzzy sets. To illustrate the need for finding the breakpoints, consider the example where the value $T = 229$. This situation is similar to the previous situation, with the difference that getNextAlphaLevelRight will return 0.8, as there is a breakpoint in C. The algorithm yields:

$$\begin{aligned}
x_A &= 0 + (229 - 200)\frac{6-0}{230-200} = 5.8 \\
x_B &= 40 + (229 - 200)\frac{44-40}{230-200} = 43.87 \\
x_C &= 60 + (229 - 200)\frac{80-60}{230-200} = 79.33 \\
x_D &= 100 + (229 - 200)\frac{100-100}{230-200} = 100
\end{aligned}$$

In this example, the term added by C is the biggest, as it is the least steep fuzzy set. This changes when the constraining value increases further; for $T = 231$, the result becomes:

$$\begin{aligned}
x_A &= 6 + (231 - 230)\frac{30-6}{280-230} = 6.48 \\
x_B &= 44 + (231 - 230)\frac{60-44}{280-230} = 44.32 \\
x_C &= 80 + (231 - 230)\frac{90-80}{280-230} = 80.2 \\
x_D &= 100 + (231 - 230)\frac{100-100}{280-230} = 100
\end{aligned}$$

This time, the term added by C is the smallest, as now the right side of this fuzzy set is the steepest. The algorithm manages to balance the terms in order to maximise the lowest membership grade.

It is obvious that the algorithm is only applicable if the fuzzy sets are convex, continuous and piecewise linear.

5. Future work

In this article, the Mean of max was used as a starting point. The reason for this stems from the application: the Mean of max was considered to be a good defuzzifier. In general, starting from a different defuzzifier can be useful under different circumstances; how this can be achieved is possible future research. Starting from e.g. Center of gravity immediately implies that the algorithm would have to start differently, but it would incorporate more knowledge concerning the shape of the fuzzy set. Extending on that is considering the use of different defuzzifiers for the different fuzzy sets that play a part.

6. Conclusion

The problem of defuzzifying multiple fuzzy sets under a shared constraint was considered here. An algorithm for shifting Mean of max values in order to maximise the lowest membership grade is presented and tested. The algorithm is limited to continuous, convex, piecewise linear fuzzy sets. It was directly developed to be used in spatial data processing, where multiple fuzzy sets need to be defuzzified while their sum equals a given value. The algorithm will improve the results of a novel approach to the map overlay problem; the approach uses a fuzzy inference system and the presented algorithm aids at by better interpreting the outcome of this system.

References

[1] Ronald R. Yager and Dimitar P. Filev. Defuzzification with constraints. *Fuzzy Logic and its Applications to Engineering, Information Sciences, and Intelligent Systems - Theory and Decision Library*, 16:157–166, 1996.

[2] Carol A. Gotway and Linda J. Young. Combining incompatible spatial data. *Journal of the American Statistical Association*, 97(458):632–648, 2002.

[3] R. Flowerdew and M. Green. *Spatial analysis and GIS; eds. Foterhingham S. and Rogerson P.*, chapter Areal interpolation and types of data, pages 141–152. Taylor & Francis, 1994.

[4] M Duckham and M. Worboys. An algebraic approach to automated information fusion. *International Journal of Geographic Information Systems*, 19:537–558, 2005.

[5] Jörg Verstraete. Solving the map overlay problem with a fuzzy approach. *Climatic Change*, pages 1–14, 2014.

[6] Jörg Verstraete. A fuzzy rulebase approach to remap gridded spatial data: Initial observations. In Anne Laurent, Olivier Strauss, Bernadette Bouchon-Meunier, and Ronald R. Yager, editors, *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, volume 442 of *Communications in Computer and Information Science*, pages 366–375. Springer, 2014.

[7] Werner Van Leekwijck and Etienne E. Kerre. Defuzzification: criteria and classification. *Fuzzy Sets and Systems*, 108:159–178, 1999.

[8] Didier Dubois and Henry Prade. *Fundamentals of fuzzy sets.* Kluwer Academic Publishers, 2000.

[9] Hans-Jurgen Zimmerman. *Practical Applications of Fuzzy Technologies.* Kluwer Academic Publishers, 1999.

[10] Lotfi A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[11] Ronald R. Yager and Dimitar P. Filev. Constrained defuzzification. In *Proc. 5th IFSA Congress*, pages 1167–1170, 1993.