FUZZY CLUSTERING WITH EVOLUTIONARY ALGORITHMS

Frank KLAWONN

FB Elektrotechnik und Informatik, FH Ostfriesland, Constantiaplatz 4, D-26723 Emden, Germany

Abstract. Objective function based fuzzy clustering aims at finding a fuzzy partition by optimizing a function evaluating a (fuzzy) assignment of a given data set to clusters, that are characterized by a set of parameters, the so–called prototypes. The iterative optimization technique usually requires the objective function not only to be differentiable, but prefers also an analytical solution for the equations of necessary conditions for local optima. Evolutionary algorithms are known to be an alternative robust optimization technique which are applicable to quite general forms of objective functions. We investigate the possibility of making use of evolutionary algorithms in fuzzy clustering. Our experiments and theoretical investigations show that the application of evolutionary algorithms to shell clustering, where the clusters are in the form of geometric contours, is not very promising due to the shape of the objective function, whereas they can be helpful in finding solid clusters that are not smooth, for example rectangles or cubes. These types of clusters play an important role for fuzzy rule extraction from data.

Keywords. Fuzzy clustering, evolutionary algorithm

1 Introduction

Many fuzzy clustering algorithms are based on the idea to optimize an objective function. Usually, this objective function depends on the distances of the data to the clusters weighted by the membership degrees. By taking the first derivative of the objective function with respect to the cluster parameters, one obtains necessary conditions for the objective function to have an optimum. These conditions are then applied in an iteration procedure and define a clustering algorithm. This requires the objective function to be differentiable and the iteration procedure can be computationally efficient only if the derived conditions lead to explicit equations for the cluster parameters. The set of cluster parameters, that determine the size and the shape of a cluster, depends on the specific application field. We mainly distinguish between fuzzy clustering as an explorative data analysis method, especially for unsupervised classification tasks, techniques for rule extraction (for instance for fuzzy controllers), and shell clustering algorithms, that are designed for boundary detection in image recognition.

Unfortunately, in many cases the restrictions that are enforced on the cluster parameters to be able to derive the iteration procedure are too narrow and exclude a lot of interesting possibilities. Thus it is desirable to have an alternative optimization technique that allows for more freedom in the choice of the cluster parameters.

In this paper we investigate whether evolutionary algorithms may be a solution to this problem. After a short introduction on fuzzy clustering in Section 2, we discuss the principal approach to use evolutionary algorithms for fuzzy clustering in Section 3 and report on the experimental and theoretical results in Section 4.

2 Objective Function Based Fuzzy Clustering

Let us briefly review some fuzzy clustering methods based on objective functions. (For an overview see for example [8].) The cluster algorithm aims at minimizing the objective function

$$J(X,U,v) \;=\; \sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ik})^m d^2(v_i,x_k) \qquad (1)$$

under the constraints

$$\sum_{k=1}^{n} u_{ik} \;>\; 0 \qquad \text{for all } i \in \{1,\ldots c\} \qquad (2)$$

and

$$\sum_{i=1}^{c} u_{ik} \;=\; 1 \qquad \text{for all } k \in \{1,\ldots n\}. \qquad (3)$$

$X = \{x_1,\ldots,x_n\} \subseteq \mathbb{R}^p$ is the data set, c is the number of fuzzy clusters, $u_{ik} \in [0,1]$ is the membership degree of datum x_k to cluster i, v_i is the prototype or the vector of parameters for cluster i, and $d(v_i,x_k)$ is the distance between prototype v_i and datum x_k. The parameter $m > 1$ is called fuzziness index. For $m \to 1$ the clusters tend to be crisp, i.e. either $u_{ik} \to 1$ or $u_{ik} \to 0$, for $m \to \infty$ we have $u_{ik} \to 1/c$. Usually $m = 2$ is chosen.

The objective function (1) to be minimized uses the sum over the quadratic distances of the data to the prototypes weighted with their membership degrees. (2) guarantees that no cluster is completely empty, (3) ensures that for each datum its classification can be distributed over different clusters, but the sum of the membership degrees to all clusters has to be 1 for each datum.

Differentiating (1), taking the constraints into account by Lagrange multipliers, one obtains

$$u_{ik} = \frac{1}{\sum_{j=1}^{c}\left(\frac{d^2(v_i,x_k)}{d^2(v_j,x_k)}\right)^{\frac{1}{m-1}}} \qquad (4)$$

as a necessary condition for (1) to have a (local) minimum. The Equation (4) is therefore used for updating the membership degrees u_{ik} in an iteration procedure until the difference between the matrix (u_{ik}^{new}) and the matrix (u_{ik}^{old}) in the previous iteration step is less than a given tolerance bound ε.

The most simple fuzzy clustering algorithm is the fuzzy c–means (FCM) (see f.e. [2]) where the distance d is simply the Euclidean distance and the prototypes are vectors $v_i \in \mathbb{R}^p$. It searches for spherical clusters of approximately the same size and by differentiating (1) we obtain the necessary condition

$$v_i = \frac{\sum_{k=1}^{n}(u_{ik})^m x_k}{\sum_{k=1}^{n}(u_{ik})^m} \qquad (5)$$

for the prototypes that are used alternatingly with (4) in the iteration procedure.

Gustafson and Kessel [7] designed a fuzzy clustering method that is looking for hyper–ellipsoidal clusters. The prototypes consist of the cluster centers v_i as in the FCM and (positive definite) covariance matrices C_i. The Gustafson and Kessel algorithm replaces the Euclidean distance by the transformed Euclidean distance

$$d^2(v_i,x_k) = (\rho_i \det C_i)^{1/p} \cdot (x_k - v_i)^\top C^{-1}(x_k - v_i).$$

These or similar algorithms can also be applied to learn fuzzy rules from data for classification problems [6, 10] or function approximation [9, 11, 15]. The fuzzy rules are derived from projections of the clusters leading to a certain loss of information which could be minimized if the clusters had the shapes of rectangles or cubes. Unfortunately, such shapes lead to non–differentiable objective functions.

In contrast to the methods that are designed for solid clusters, the so–called shell clustering algorithms are tailored for clusters in the form of boundaries of circles, ellipses, etc. (For an overview on shell clustering see [12].) Davé [5] developed one of the first shell clustering algorithms for the detection of circles. Each prototype consists of the cluster center v_i and the radius r_i. The distance function for the fuzzy c–shells algorithm (FCS) is

$$d^2((v_i,r_i),x_k) = (\| x_k - v_i \| - r_i)^2$$

so that exactly those data have zero distance to the cluster that are located on the circle with radius r_i and center v_i. Unfortunately, this distance function leads to a set of coupled non–linear equations for the v_i and r_i that cannot be solved in an analytical way. Thus an additional numerical iteration procedure to solve non–linear equations is necessary in each iteration step of the clustering algorithm which causes a lot of computational costs. Although this specific problem is solved for circles by the fuzzy c–spherical shells algorithm (FCCS) [13] using the distance function

$$d^2((v_i,r_i),x_k) = (\| x_k - v_i \|^2 - r_i^2)^2,$$

the general problem for other shell shapes remains.

3 Evolutionary Algorithms for Fuzzy Clustering

Evolutionary algorithms are a class of optimization methods that are inspired by the process of biological evolution. The principal idea is to have a collection or *population* of possible problem solutions encoded as parameter vectors – the *chromosomes* – that define a solution. From this population a new population – the next *generation* – is generated by first producing offspring from the old chromosomes by changing some components of the chromosomes, the *genes*, randomly and sometimes also by a mixing of genes of different chromosomes (*crossover*), and then by selecting the best chromosomes for the next generation. For details we refer to books like [1, 14].

As a quite general optimization strategy, evolutionary algorithms might be applicable to objective function based fuzzy clustering. Thus it is necessary to find a suitable coding of the parameters to be determined in fuzzy clustering. Obviously, the parameters to be optimized are the prototypes v_i and the membership degrees u_{ik}. In [3] it was proposed to perform hard clustering (i.e. $u_{ik} \in \{0,1\}$), with genetic algorithms by taking the u_{ik} as the parameters for the evolutionary algorithm. For fuzzy clustering this does not seem to be suitable, since this means that besides the prototypes $c \cdot n$ real–valued parameters have to be optimized, where c is the number of clusters and n the number of data vectors. Since the formula (4) is generally valid independent of the special type of prototype, it is not necessary to optimize the parameters u_{ik} by an evolutionary algorithm. In addition, the problems for the standard iterative optimization procedure are

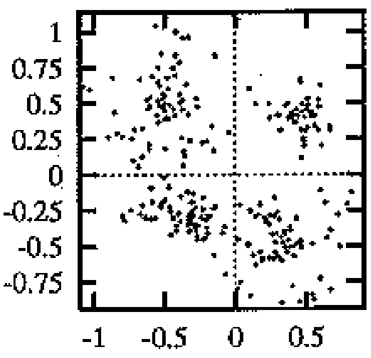
Figure 1: A test data set for the FCM

not caused by the membership degrees, but by the choice of the prototype parameters. Thus we restrict ourselves here to optimize only the prototypes by an evolutionary algorithm. The corresponding membership degrees are computed as in the usual algorithm on the basis of equation (4).

The aim of fuzzy clustering depends on the application domain. In the case of data analysis and classification tasks as well as for rule extraction it is important to find an appropriate (gradual) assignment of the data to suitable prototypes, i.e., the emphasis is on the assignment and not on the exact prototype parameters. For shell clustering algorithms to detect geometrical objects in images the exact parameters, i.e. the prototype, play an important role. This means that it is not sufficient just to compile the data points on a circle in one cluster, but that the center and the radius of the circle are of great interest.

4 Experimental and Theoretical Results

Before we apply evolutionary algorithms to fuzzy clustering with non–standard prototypes, we first test their performance by two standard fuzzy clustering techniques, namely the FCM as an example for the search after solid clusters and the shell clustering algorithm FCS.

Figure 1 shows a test data set for the FCM with four clusters. Evolutionary algorithms with different parameters were all able to solve the problem of finding suitable prototypes. An example run is illustrated in Figure 2 where the tracks of the prototypes computed over the generations by the evolutionary algorithm are indicated. A comparison of different selection strategies (left to right: roulette wheel, remainder stochastic sampling, tournament) in Figure 3 shows that tournament selection has the best performance. In the figure the dashed line indicates the average fitness and the other line the best

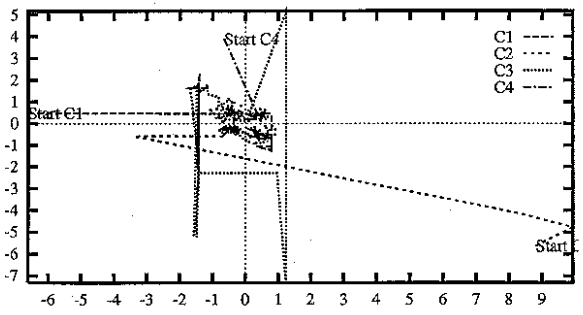

Figure 2: Tracking the prototypes

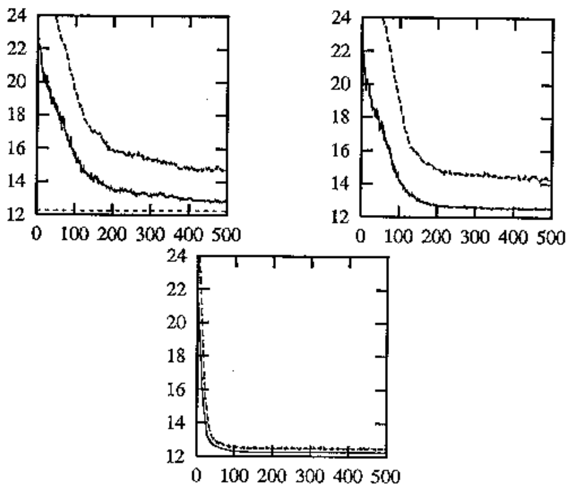
Figure 3: Comparison of selection strategies

fitness in each generation, averaged over 18 runs of the evolutionary algorithm.

Looking at these good results for the FCM we were quite optimistic also for shell clustering. However, the results were more or less disappointing. Figure 4 shows a test data set with five circles for the FCS and two results of the evolutionary algorithm. In one case no circle was detected correctly, in the other only one of the five. These results could not be improved, neither by experimenting with the mutation or crossover rate nor by introducing techniques like controlling the mutation rate on the basis of a span measure [14].

The typical results of the optimization of the objective function of the FCS by an evolutionary algorithm tend to yield larger circles – an observation which was also made in [4] where rectangular shells were considered. Thus we tested the evolutionary algorithm with a data set representing only one circle with center (0,0) and radius 2. In one case we limited the search space for the radius and the coordinates to the interval $[-2.2, 2.2]$, in the other case to the interval $[-22, 22]$. In the first case the evolutionary algorithm computed the correct radius and center in all test runs after about 30 generations. For the second case the circle was detected

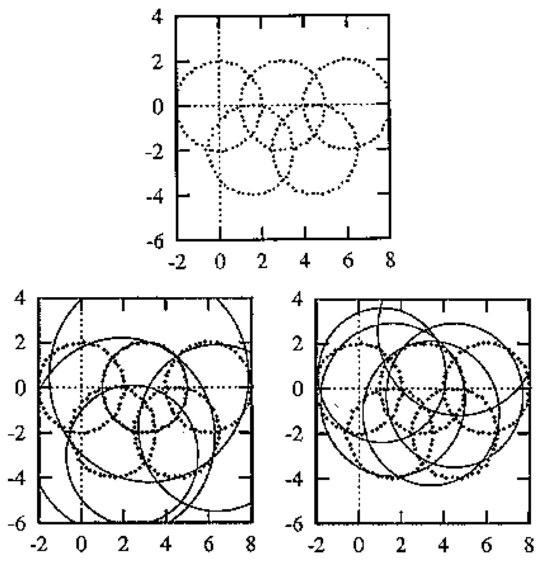
Figure 4: FCS: Test data and results

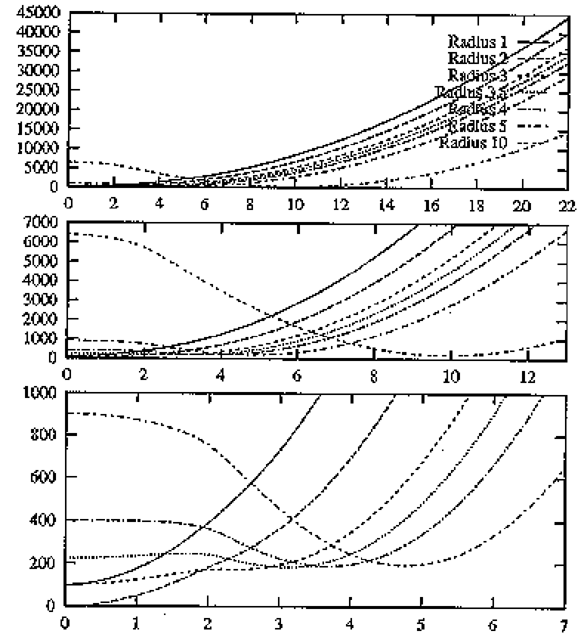

Figure 5: Evaluation of chromosomes with shifted circle centers

correctly only in about 60% of the cases after approximately 125 generations.

This motivated us to take a closer look at the fitness function. We consider again a circle with center (0,0) and radius 2 as the data set. In Figure 5 the evaluation of a chromosome is shown whose y–coordinate for the circle is the correct value 0, whereas the x–value is shifted to the right between 0 and 22. The different curves were drawn for radius values between 1 and 10. The middle and lower diagram are just magnified scalings of the upper diagram.

From this figure it is obvious that the correct radius 2 gets the best evaluation only as long as the circle center is not shifted to far away from the original circle center. The smaller radius 1 does never yield better values than a larger radius. And with increasing distance of the shifted center to the original center, a larger radius gets a better evaluation. This implies that in early generations where the random circle centers are still far away from the correct circle centers, chromosomes with smaller radius values are not being selected. But when these values are missing in the population, a random mutation to the correct radius without mutating also the center to the correct value leads to a very bad evaluation. Thus these genes are so strongly dependent on each other that only a simultaneous random jump of all genes to the correct values can lead to good results.

When we applied the evolutionary algorithm again to the data set of Figure 1 and limited the search range for radius genes to the interval $[0, 2.2]$, the results were satisfactory. In all test runs the circles were detected correctly after about 80 generations in average.

Our theoretical considerations and experiments show that the applicability of evolutionary algorithms to shell clustering is not very promising except when the parameter range can be restricted to quite limited bounds.

As mentioned in Section 2, the standard iteration procedure is difficult or impossible not only for certain types of shell clusters, but also for instance for solid rectangular clusters. Such clusters would be ideal for fuzzy rule extraction, especially when they are restricted to axis parallel rectangles or cubes.

In order to obtain clusters of this type we define as prototypes for p–dimensional data cluster centers $z_i \in \mathbb{R}^p$ and diagonal matrices B_i. As the distance function we choose

$$\begin{aligned} d^2(x_k, (z_i, B_i)) &= \| B_i(z_i - x_k) \|_\infty^2 \\ &= \left(\max_{1 \le j \le p} \{ b_{ij} \cdot |z_{ij} - x_{kj}| \} \right)^2 . \end{aligned}$$

In opposition to [4] we use the supremum norm $\| \cdot \|_\infty$ instead of the 1–norm. In order to avoid the undesired solution $b_{ij} = 0$ for all i, j, we have to enforce a restriction on the matrices B_i. Analogously to the Gustafson and Kessel algorithms we require the matrix B_i to have a fixed value ϱ_i for the determinant, which determines the size or the volume of the cluster i. If nothing is known about the data, we simply choose $\varrho_i = 1$ for all $i = 1, \ldots, c$.

All results were quite satisfactory. In 90–100 percent of the test runs the evolutionary algorithm was able to assign the data to the correct clusters

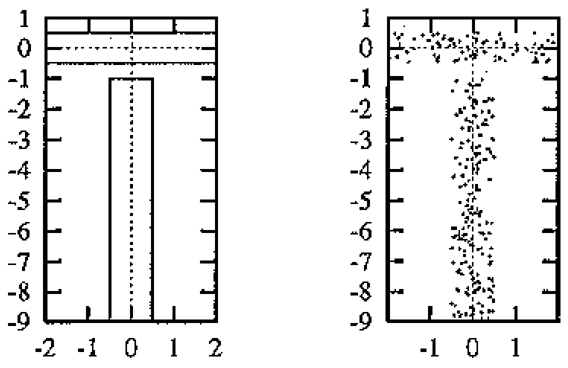

Figure 6: Two separated rectangular clusters (not filled and filled)

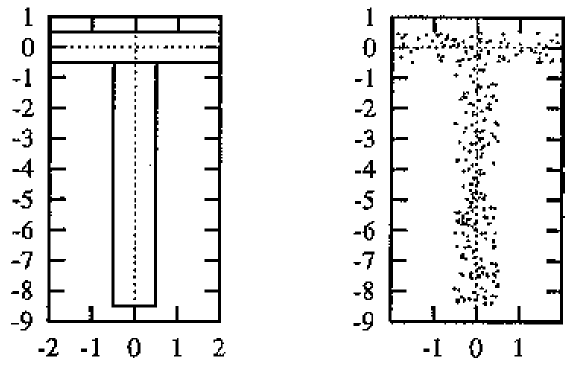

Figure 7: Two contiguous rectangular clusters (not filled and filled)

and the cluster centers were detected approximately correct. Figures 6 and 7 show two 2–dimensional examples. In both cases we used two data sets – one in which data points were only placed on the edges of the rectangles and one were the rectangles were filled with data points.

It is worth noticing that the best chromosome in Figure 7 for the data set with points only one the edges had the vectors $(0, -0.1)$ and $(0, -5.1)$ for the cluster centers and and the values 0.6 and 1.6 for the entree in the upper left corner of the diagonal matrices. The other non–zero value of the 2–dimensional diagonal matrix can then be calculated, since the determinants are fixed. This chromosome was assigned the error value 626.5 whereas the 'desired' solution with centers $(0, 0)$ and $(0, -4.5)$ and matrix values 0.5 and 2.0 gets a larger error value of 740.8. Thus this method is not suited for computing the correct parameters of rectangular shells. Nevertheless, the assignment of the data to the clusters was satisfactory even if data points were only present on the edges. Also the 3–dimensional case in Figure 8 and 9 caused no problems.

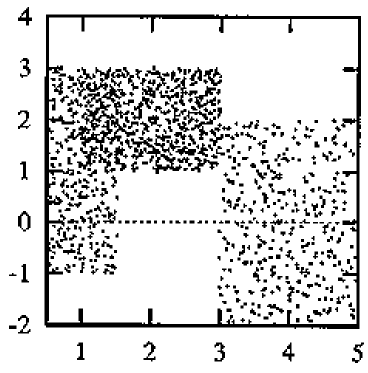

Figure 8: x/y–projections of three 3–dimensional clusters

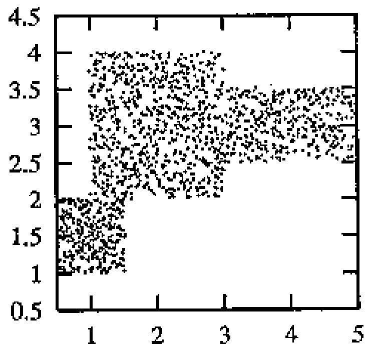

Figure 9: x/z–projections of three 3–dimensional clusters

5 Conclusions

Our investigations show that shell clustering with evolutionary algorithms seems to be quite problematic, since there exist a lot of local optima and the correct solution often looks like a very narrow optimum. The situation is better for solid clusters. It should however be noted that evolutionary algorithms require a much longer computation time to solve the problem compared to the standard iteration procedure so that the application of evolutionary algorithms makes only sense when the standard iteration procedure cannot be applied according to a non–differentiable distance function or when an analytical solution for the single iteration steps cannot be found.

Acknowledgments. I would like to thank Bettina Fischer for implementing the evolutionary algorithms and for carrying out a lot of experiments.

References

[1] T. Bäck, Evolutionary Algorithms in Theory and Practice. Oxford University Press, Oxford (1996).

[2] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York (1981).

[3] J.C. Bezdek, S. Boggavarapu, L.O. Hall, A. Bensaid, Genetic Algorithm Guided Clustering. Proc. First IEEE Conf. on Evolutionary Computation, Orlando (1994), 34–38.

[4] J.C. Bezdek, R.J. Hathaway, N.R. Pal, Norm–Induced Shell–Prototypes (NISP) Clustering. Neural, Parallel & Scientific Computation 3 (1995), 431–450.

[5] R.N. Davé, Fuzzy Shell Clustering and Application to Circle Detection in Digital Images. Intern. Journ. General Systems 16 (1990), 343–355.

[6] H. Genther, M. Glesner, Automatic Generation of a Fuzzy Classification System Using Fuzzy Clustering Methods. Proc. ACM Symposium on Applied Computing (SAC'94), Phoenix (1994), 180–183.

[7] D.E. Gustafson, W.C. Kessel, Fuzzy Clustering with a Fuzzy Covariance Matrix. Proc. IEEE CDC, San Diego (1979), 761–766.

[8] F. Höppner, F. Klawonn, R. Kruse, Fuzzy–Clusteranalyse: Verfahren für die Bilderkennung, Klassifikation und Datenanalyse (in German). Vieweg, Braunschweig (1997).

[9] F. Klawonn, R. Kruse, Clustering Methods in Fuzzy Control. In: W. Gaul, D. Pfeifer (eds.), From Data to Knowledge: Theoretical and Practical Aspects of Classification, Data Analysis and Knowledge Organization. Springer–Verlag, Berlin (1995), 195–202.

[10] F. Klawonn, R. Kruse, Derivation of Fuzzy Classification Rules from Multidimensional Data. In: G.E. Lasker, X. Liu (eds.), Advances in Intelligent Data Analysis. The International Institute for Advanced Studies in Systems Research and Cybernetics, Windsor, Ontario (1995), 90–94.

[11] F. Klawonn, R. Kruse, Automatic Generation of Fuzzy Controllers by Fuzzy Clustering. Proc. 1995 IEEE Intern. Conf. on Systems, Man and Cybernetics, Vancouver (1995), 2040–2045.

[12] R. Krishnapuram, H. Frigui, O. Nasraoui, Fuzzy and Possibilistic Shell Clustering Algorithms and Their Application to Boundary Detection and Surface Approximation – Part 1 & 2. IEEE Trans. on Fuzzy Systems 3 (1995), 29–60.

[13] R. Krishnapuram, O. Nasraoui, H. Frigui, The Fuzzy C Spherical Shells Algorithm: A New Approach. IEEE Trans. on Neural Networks 3 (1992), 663–671.

[14] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs. Springer, Berlin (1992).

[15] M. Sugeno, T. Yasukawa, A Fuzzy–Logic–Based Approach to Qualitative Modeling. IEEE Transactions on Fuzzy Systems 1 (1993), 7–31.