A New Fuzzy Inferencing Method for Inductive Reasoning

Francisco Mugica[*] François E. Cellier[†]

Abstract

In this paper, a new fuzzy inferencing method is presented, and its performance is compared to that of other fuzzy inferencing methods (sometimes referred to as "defuzzification" methods) that are frequently cited in the literature. The choice of the fuzzy inferencing engine can decide over success or failure of applications of fuzzy technology in areas such as qualitative modeling or qualitative control. It will be demonstrated that the new fuzzy inferencing technique, called the five-nearest-neighbors (5NN) rule, performs exceedingly well in a data-driven environment, i.e., in a situation where a fuzzy system is automatically being synthesized from available measurement data of its surroundings. All fuzzy inferencing algorithms that are compared in this paper have been implemented in SAPS-II, an experimental software for the automated synthesis of qualitative models, fuzzy controllers, and fuzzy inductive reasoners. The advantages of the new fuzzy inferencing method will be shown by means of the identification of a qualitative abstraction of a differential equation model.

Keywords: Fuzzy Inferencing; Defuzzification; Fuzzy Systems; Fuzzy Control; Inductive Reasoning.

1 Introduction

An *inductive reasoner* is a finite state machine that infers the qualitative behavior of a derived variable (an "output") from the qualitative behavior of other variables (the "inputs") that stand in some sort of a (possibly time-dependent) cause-effect relationship with it.

Qualitative modeling refers to the process of identifying a set of rules (i.e., the finite state machine) that best characterize the behavior of the system under investigation in qualitative terms. In the case of our software SAPS-II [3], [4], this is done using the General Systems Problem Solving (GSPS) approach [8], but that is of no direct concern to this paper. Any method that generates a set of rules qualitatively characterizing a (possibly time-dependent) process can be rightfully called a qualitative modeling tool.

Qualitative simulation refers to the process of using the previously identified rules to predict future values of the output as a function of its own past and new as well as old values of the inputs. In the context of knowledge-based systems, such as expert systems, this process is called *inferencing*. In the context of the GSPS methodology, it is called *forecasting*.

Inductive reasoning refers to a special type of algorithm that involves a qualitative modeling and a qualitative simulation engine. "Inductive" here refers to the fact that the modeling engine

[*]Departament d'ESAII, Universitat Politècnica de Catalunya, Diagonal 647, 2da. planta, Barcelona 08028, Spain, mujica@esaii.upc.es - Supported by a grand from the CONACyT - México

[†]Department of Electrical and Computer Engineering, The University of Arizona, Tucson, Arizona 85721, U.S.A., cellier@ece.arizona.edu

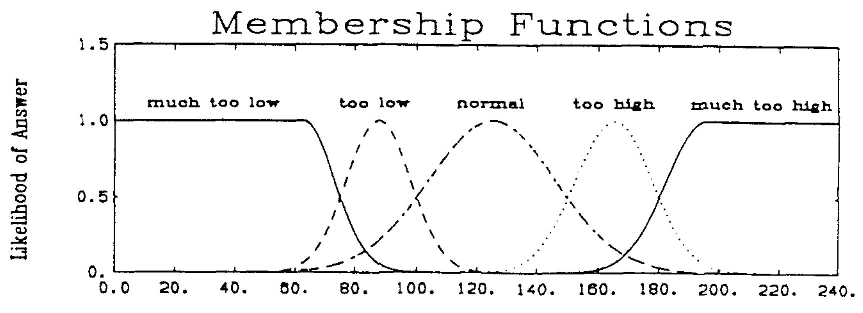


Figure 1: Fuzzification in SAPS–II

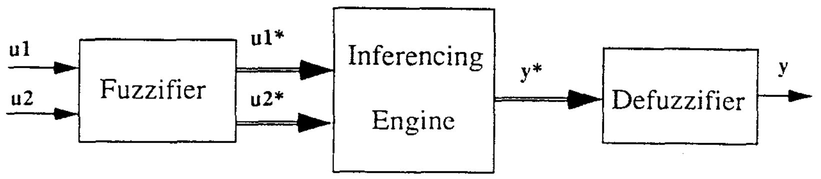


Figure 2: Fuzzy Inductive Reasoning

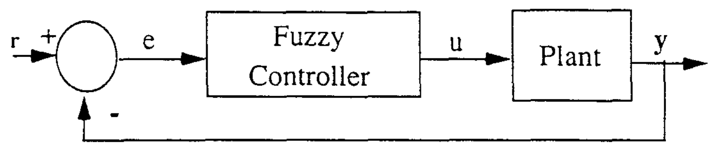
is data–driven, i.e., rules are derived on the basis of measurement data rather than through some sort of meta–knowledge. "Reasoning" refers to the process of inferring new facts from old facts and a given set of rules. Qualitative simulation qualifies as a temporal reasoning technique.

Fuzzy inductive reasoning adds one more facet to the inductive reasoning process. Here, quantitative input variables are first characterized into qualitative class values together with some kind of fuzzy membership value. In the context of control engineering, this process is referred to as *fuzzy discretization*; in the context of statistics, it is called *fuzzy classification*; in the context of the GSPS methodology, the process has been coined *fuzzy recoding*; and in the context of the fuzzy system methodology, it has been named *fuzzification*. In SAPS–II, the process of fuzzification maps quantitative (real–valued) signals into *qualitative triples* including the class value, the fuzzy membership value, and a side value. The fuzzification process is depicted in Figure 1.

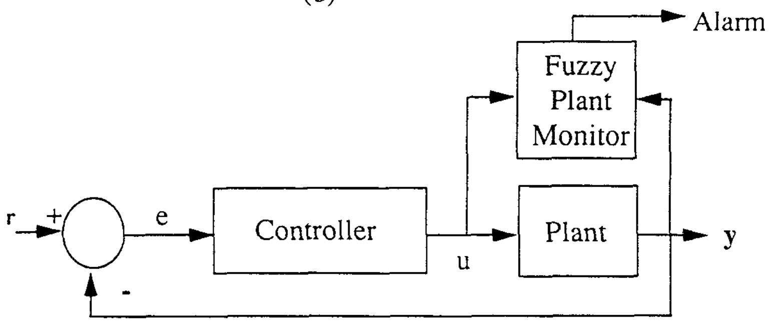
SAPS–II uses bell–shaped (Gaussian) membership functions that carry a value of 0.5 at the landmark between two neighboring classes, and a value of 1.0 at the arithmetic mean between two neighboring landmarks. The side function is an enumerated variable with the values "left" (meaning: to the left of the maximum) or "right" (meaning: to the right of the maximum). No information loss is incurred in the process of *recoding* quantitative signals into qualitative signals (i.e., qualitative triples). The original quantitative signals can be *regenerated* from the qualitative signals without any error.

Figure 2 illustrates the process of fuzzy inductive reasoning.

The fuzzifier converts each quantitative input signal u_i into a qualitative input signal (triple) u_i^*. The inference engine infers a qualitative output signal (another triple) y^* from the qualitative inputs, and the defuzzifier converts the qualitative output signal back to a quantitative output signal y. Since all global inputs and outputs of the fuzzy inductive reasoner are quantitative signals, fuzzy inductive reasoners can be embedded in (used in conjunction with) quantitative signal processing engines. The fuzzy inductive reasoner may e.g. represent a fuzzy controller of a plant (Figure 3a), or it can be used as an intelligent supervisory plant monitor (Figure 3b). On a superficial level, fuzzy inductive reasoners look just like sampled–data controllers with



(b)



(b)

Figure 3: Applications of Fuzzy Inductive Reasoners

the fuzzifier replacing the A/D–converters, the inference engine replacing the discrete–time controller, and the defuzzifier replacing the D/A–converter.

The inference engine can be decomposed into two parts: a *crisp inference engine* that infers a class value and a side value of the output from class and side values of the inputs, and a *fuzzy inference engine* that maps the fuzzy membership values of the inputs into a fuzzy membership value of the output, using some *fuzzy inferencing algorithm.*

A discussion of various fuzzy inferencing algorithms is at the focal point of this paper. In the literature, the fuzzy inferencing algorithms are frequently referred to as "defuzzification algorithms," but this is actually a misnomer, since they are not even a part of the defuzzifier. The confusion stems from the fact that many authors (and codes) do not clearly separate the defuzzifier from the inferencing engine, which we believe to be a mistake. The task of the defuzzifier is simply that of regenerating quantitative signals from their qualitative counterparts.

2 Fuzzy Inferencing Mechanisms

Since no information is lost in the process of recoding (fuzzification), the reader may be inclined to ask what the advantage is of decomposing quantitative signals into qualitative ones. The answer to this question is simple. Inductive modeling necessarily involves some sort of optimization algorithm. *Quantitative inductive modeling tools* identify a model by searching a continuous search space for an optimal solution (curve fitting). To this end, most algorithms parameterize the model and minimize some kind of performance index over the (real–valued) parameter space, e.g. through some sort of gradient technique. *Qualitative inductive modeling tools*, on the other hand, limit their search to a much reduced discrete search space. They can therefore afford to even consider exhaustive search (if necessary), and are guaranteed to find the global minimum (within the constraints of the discrete search space).

The purpose of the fuzzy membership values then is to smoothly interpolate between neighboring discrete solution points. Thus, by recoding (fuzzifying) both the inputs and the output of the system to be modeled, the solution of a very nasty and complicated real–valued optimization problem is reduced to the solution of a much simpler integer–valued optimization problem plus that of an interpolation between neighboring discrete solution points. The task of the fuzzy inferencing algorithm is to ensure that the interpolation between neighboring discrete solution points is smooth and to preserve on the way as much information about the system to be modeled as possible.

Fuzzy logic provides us with a beautiful mechanism to ensure the former of the two requirements, i.e., to guarantee the smoothness of the behavior of the defuzzified output variables. All fuzzy inferencing algorithms will achieve this. In fact, fuzzy inferencing techniques can be tuned to behave exactly like a linear interpolation algorithm [9]. The latter requirement, however, cannot be satisfied without providing the fuzzy inferencing algorithm with information about the system to be modeled, e.g. in the form of measurement data. Most fuzzy inferencing algorithms don't do this at all. In fact, fuzzy logic was not originally designed for use in an inductive algorithm, but, on the contrary, for use in deductive algorithms. Lotfi Zadeh, the father of fuzzy logic, maintains to this day that researchers (such as us) who design tools for the automated synthesis of fuzzy controllers actually *abuse* fuzzy logic (personal communication). According to Zadeh, the beauty of the fuzzy technology is that it makes the design of (fuzzy) controllers for highly non–linear plants *easy*. Even high–school students can master the task of designing a fuzzy controller for an inverted pendulum, for example. In this light, it is important that specific system knowledge need *not* to be used in the design of the controller. Zadeh certainly has a point there. His fuzzy controllers will, in all likelihood, be much more *robust* than ours, precisely because they don't rely on specific system knowledge.

Our fuzzy controllers are much more sophisticated algorithms. Since they make use of specific system knowledge, they can do much better than simple linear interpolation. Consequently, we get away with a much coarser discretization, which in turn makes the task of finding the optimal finite state machine a much simpler and faster one and reduces the size of the finite state machine (i.e., the number of rules) dramatically. We usually get away with three to five classes for each variable, whereas more traditional fuzzy controllers always use somewhere between seven and 13 classes. On the other hand, our technique won't work at all without being fed with specific and explicit knowledge of the system to be modeled, and if this knowledge changes over time, our controller may not be able to adjust and would then fail miserably.

In an analogy: our fuzzy controllers act similarly to optimal state feedback controllers. They are sophisticated algorithms that are minutely tuned to the system they are designed for. However, if the system suddenly changes its characteristics, they fall flat on their bellies. They are not robust at all, just like the optimal state feedback controllers are not robust. On the other hand, traditional fuzzy controllers resemble the classical PI–controllers (the "vacuum cleaners" among the control algorithms). They never work exceedingly well, but they usually get the work done somehow.

In the next few sections, two of the more commonly used fuzzy inferencing algorithms: the Mean–of–Maxima (MoM) technique and the Center–of–Area (CoA) technique are described together with our own Five–Nearest–Neighbors (5NN) algorithm. The paper concludes with an example that demonstrates the far superior performance of the 5NN algorithm in comparison with its two contenders, which can be explained by the fact that the 5NN algorithm relies much more explicitly and extensively than the other two algorithms on measurement data.

3 Mean–of–Maxima Fuzzy Inferencing Algorithm

Once the selected variables of the system have been recoded, the qualitative modeling process in SAPS–II is able to find the best inductive reasoner (a so–called "mask" in the SAPS terminology). A *mask* denotes a dynamic relationship between qualitative variables. The mask can be used to flatten a dynamic relationship out into a rule, or static relationship. To this end, the

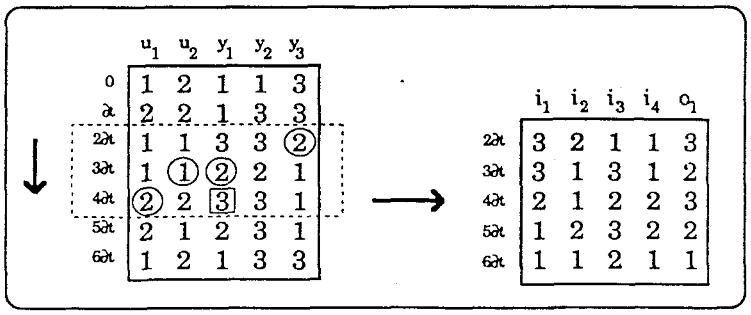


Figure 4: Flattening dynamic relationships through masking

mask can be shifted over the episodical behavior (i.e., the recoded measurement data matrix), pick out the selected inputs and output, and write them next to each other in one row. Figure 4 illustrates this process.

After the mask has been applied to the raw data, the formerly dynamic episodical behavior has become static, i.e., the relationships are now contained within single rows. These relationships, properly sorted, constitute the *behavior matrix* of the system. The behavior matrix can be viewed as a set of rules in the contexts of fuzzy controllers and expert systems. The behavior matrix is a finite state machine. For each combination of qualitative input values, it shows which qualitative outputs have been observed. In truth, the behavior of the system consists of three separate matrices: the behavior matrix itself in which the class values shown in figure 4 are stored, its corresponding side matrix, and the associated fuzzy membership matrix.

As long as the qualitative simulation process progresses, new facts are being produced by the inductive reasoner (the mask), actualizing the state of input variables of the qualitative model (the antecedents). It is the task of the qualitative simulation to find the respective outputs (the conlusions).

With the new set of input variables, all rows of the behavior matrix that match up with the inputs will be selected (these rules will be fired). The set of selected (fired) rows (rules) must be evaluated in order to reach a conclusion. They form the *selected conclusion set.*

Up to this point, only the behavior matrix itself was used in the algorithm. The determination of the selected conclusion set is entirely crisp, and it is the same for all three inferencing algorithms.

The evaluation of the selected conclusion set, however, depends on the membership information, and it varies in the three inferencing algorithms. The confidence expressed in a given behavior record (a row of the behavior matrix) is determined as the joint membership of all the variables associated with that row (rule). The joint membership of i membership functions is defined as the smallest individual membership of all its elements:

$$Memb_{\text{joint}} = \bigcap_{\forall i} Memb_i = \inf_{\forall i}(Memb_i) \overset{\text{def}}{=} \min_{\forall i}(Memb_i) \qquad (1)$$

Once the joint memberships have been computed for each row separately, these will then be regrouped, merging all records that are identical in their input class values and in their output class value, assigning to the newly formed group record a confidence value that is equal to the largest of the joint membership values of the individual records. This operation is known in fuzzy theory as a *min-max operation* [13]. The result of this operation forms the *potential conclusion set.*

$$Memb_{\text{conclusion}} = \max_{x}\{Memb_{\text{joint}}(x)\} \tag{2}$$

The Mean–of–Maxima (MoM) inferencing algorithm, at this point, computes directly the defuzzified output (i.e., the crisp conclusion) as the weighted average of the peak values of all potential conclusions obtained with the min–max operation, thereby merging the steps of fuzzy inferencing and defuzzification. The information of the shapes of the membership functions and the specific membership values of the potential conclusions are lost because only the peak values are being considered.

$$crisp_{\text{conclusion}} = \frac{\sum_{\forall i} Memb_i(peak_i) \times peak_i}{\sum_{\forall i} Memb_i(peak_i)} \tag{3}$$

However, for modularity and other reasons, we prefer to separate the defuzzification step from the fuzzy inferencing step. Thus, the forecasting algorithm of SAPS will not return the defuzzified output value itself, but instead, it will return a single qualitative triple (i.e., a class value, a membership value, and a side value), that, when defuzzified later, leads to a regeneration of the desired crisp conclusion. Since the MoM algorithm goes after the defuzzified output directly, we had no choice but to recode the defuzzified output again within the forecasting algorithm in order to be able to return a qualitative triple.

4 Center–of–Area Fuzzy Inferencing Algorithm

Maybe the most commonly used fuzzy inferencing algorithm is the Center–of–Area (CoA) algorithm. In contrast to the MoM technique, not only the peak values of the potential conclusions are considered, but also the shapes of the membership functions are taken into account. The justification for the CoA algorithm is twofold. Sometimes membership functions are chosen that are non–symmetric to their peak values. In that case, a selection of the peak to represent the membership function as a whole may be a poor choice. Secondly, not all classes must always be equally powerful. In the MoM technique, each class is considered equal. Weighting takes place only in relation to the confidence expressed in each of the potential conclusion classes, i.e., the potential conclusions extracted by the min–max rule are combined by means of the union of their respective membership functions truncated to the maximum value for each interval. Figure 5 illustrates this process. The CoA algorithm is a powerful and general method that applies not only to individual output variables (as needed in our application), but also to multidimensional outputs. In that case, the technique is usually referred to as *Center–of–Gravity (CoG)* fuzzy inferencing technique.

The general continuous form of this algorithm can be computed as:

$$crisp_{\text{conclusion}} = \frac{\sum_{\forall i} \int_{X_i} x \times Memb_i(x)dx}{\sum_{\forall i} \int_{X_i} Memb_i(x)dx} \tag{4}$$

Most references approximate the integrals by Riemann–sums, since an accurate evaluation of the integrals is computationally expensive. Two interesting applications of the CoA inferencing algorithm related to neural networks are cited in [10]. If the number of quantization levels of the output i is q_i, the approximated solution can be written as follows:

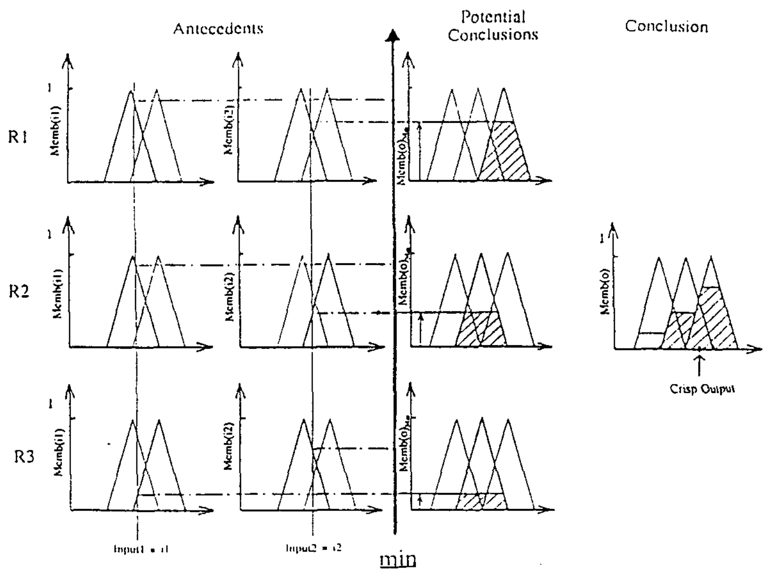


Figure 5: Diagrammatic representation of the min–max center–of–gravity method

$$crisp_{\text{conclusion}} = \frac{\sum_{k=1}^{q_i} x_k \times Memb_i(x_k)}{\sum_{k=1}^{q_i} Memb_i(x_k)} \tag{5}$$

In SAPS, we decided to formally keep the integrals, but approximate their solutions by means of numerical integration using the trapezoidal algorithm. As in the case of the MoM algorithm, the crisp conclusion is then immediately recoded again in order to be able to return a qualitative triple.

5 Five–Nearest–Neighbors Fuzzy Inferencing Algorithm

In the five–nearest–neighbors (5NN) fuzzy inferencing algorithm, the membership and side functions of the new input are compared with those of all previous recordings of the same qualitative input contained in the behavior matrix. The one input with the most similar membership and side functions is identified. For this purpose, a cheap approximation of the regenerated quantitative signal

$$d_i = 1 + side_i * (1 - Memb_i) \tag{6}$$

is computed for every input variable of the new input set, and the regenerated d_i values are stored in a vector. This reconstruction is then repeated for all previous recordings of the same input set. Finally, the $\mathcal{L}_2$ norms of the difference between the d vector of the new input and the d vectors of all previous recordings of the same input are computed, and the previous recording with the smallest $\mathcal{L}_2$ norm is identified. Its *output* and *side* values are then used as forecasts for the *output* and *side* values of the current state.

Forecasting of the new membership function is done a little differently. Here, the five previous recordings with the smallest $\mathcal{L}_2$ norms are used (if at least five such recordings are found in the behavior matrix), and a distance–weighted average of their fuzzy membership functions is computed and used as the forecast for the fuzzy membership function of the current state.

Absolute weights are computed as follows:

$$w_{\text{abs}_k} = \frac{d_{\max} - d_k}{d_{\max}} \tag{7}$$

where the index k loops over the five nearest neighbors, and $d_i \leq d_j \quad , \quad i < j; \quad d_{\max} = d_5$. The absolute weights are numbers between 0.0 and 1.0. Using the sum of the five absolute weights:

$$s_w = \sum_{\forall k} w_{\mathrm{abs}_k} \tag{8}$$

it is possible to compute relative weights:

$$w_{\mathrm{rel}_k} = \frac{w_{\mathrm{abs}_k}}{s_w} \tag{9}$$

Also the relative weights are numbers between 0.0 and 1.0. However, their sum is always equal to 1.0. It is therefore possible to interpret the relative weights as percentages. Using this idea, the membership function of the new output can be computed as a weighted sum of the membership functions of the outputs of the previously observed five nearest neighbors:

$$Memb_{out_{new}} = \sum_{\forall k} w_{rel_k} \cdot Memb_{out_k} \tag{10}$$

Contrary to the MoM and the CoA techniques, the 5NN algorithm clearly separates the fuzzy inferencing step from the defuzzification step. Rather than coming up with a single formula to estimate the crisp conclusion, the algorithm contains three formulae to separately estimate the class value, the side value, and the fuzzy membership value of the conclusion. It does so on the basis of past experience (previous measurement data) rather than on the basis of assumed membership functions of the outputs. This turns out to be a fruitful idea.

6 An Experiment

The performance of the new fuzzy inferencing technique will be compared with that of the other two methods by means of the identification of a qualitative abstraction of the following differential equation model:

$$\dot{\mathbf{x}} = \mathbf{A} \cdot \mathbf{x} + \mathbf{b} \cdot u = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -2 & -3 & -4 \end{pmatrix} \cdot \mathbf{x} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \cdot u \tag{11}$$

$$\mathbf{y} = \mathbf{C} \cdot \mathbf{x} + \mathbf{d} \cdot u = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \mathbf{x} + \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \cdot u \tag{12}$$

The system was implemented in the Matlab enviroment. 900 seconds (300 sampling steps) were quantitatively simulated, and the results of this simulation were stored. The first 270 steps were used by SAPS–II/Matlab for the characterization of the system's behavior, i.e., for the identification of a qualitative model (a rule base). The input of the system was excited by using a random binary input signal. Since the input is already binary, no recoding was necessary. The output variables were recoded into three classes. The landmarks between these classes were chosen such that each class contains the same number of observations.

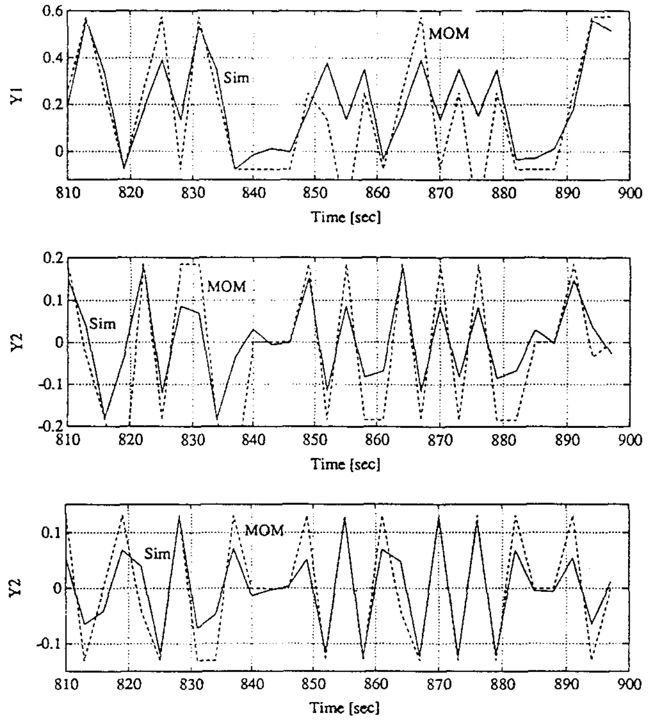


Figure 6: Behavior of the MoM fuzzy inferencing method

The last 30 steps of the continuous-time simulation were used as a gauge to judge the performance of the different methods of fuzzy inferencing.

6.1 Results

The dismal behavior of the mean-of-maxima fuzzy inferencing algorithm can be seen in figure 6.

The poor behavior of the MoM inferencing method is predominantly caused by the small number of discretization classes chosen. Three classes for each of the output variables does not provide the method with sufficient discrimination power to enable it to come up with a decent estimate of the defuzzified outputs. An augmentation of the number of classes per variable would solve this problem, but at the expense of requiring a considerably larger number of measurement data records for the identification of the qualitative model. If we want to have five members in each class as recommended by statistics, the currently used discretization requires $5 \times 2 \times 3 \times 3 \times 3 = 270$ measurement records for the identification of the qualitative model. Had we decided to recode each of the outputs into seven classes (as recommended for MoM inferencing), we should have used $5 \times 2 \times 7 \times 7 \times 7 = 3430$ data records.

The behavior of the center-of-area fuzzy inferencing algorithm is shown in figure 7.

It can be noticed that the performance of the CoA algorithm is somewhat better than that of the MoM method. This observation can be explained be the fact that the CoA algorithm takes the shape of the membership functions into account, whereas the MoM technique only looks at their maxima. In addition, CoA attempts some sort of relative weighting of the different membership functions against each other, something MoM doesn't do. Unfortunately, CoA inferencing is computationally very expensive.

Figure 8 shows the comparison between the forecasting errors of the Center-of-area, Maximum-of-Media, and 5-Nearest-Neighbors techniques.

The forecasting power of this technique is so astoundingly excellent that we first were suspicious that we might have made a mistake (such as using the very values that we tried to

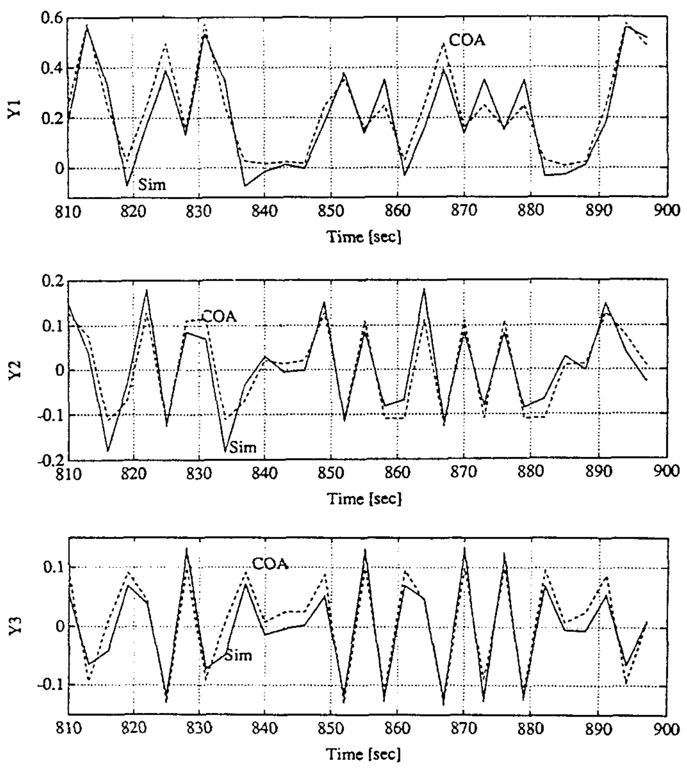


Figure 7: Behavior of the CoA fuzzy inferencing method

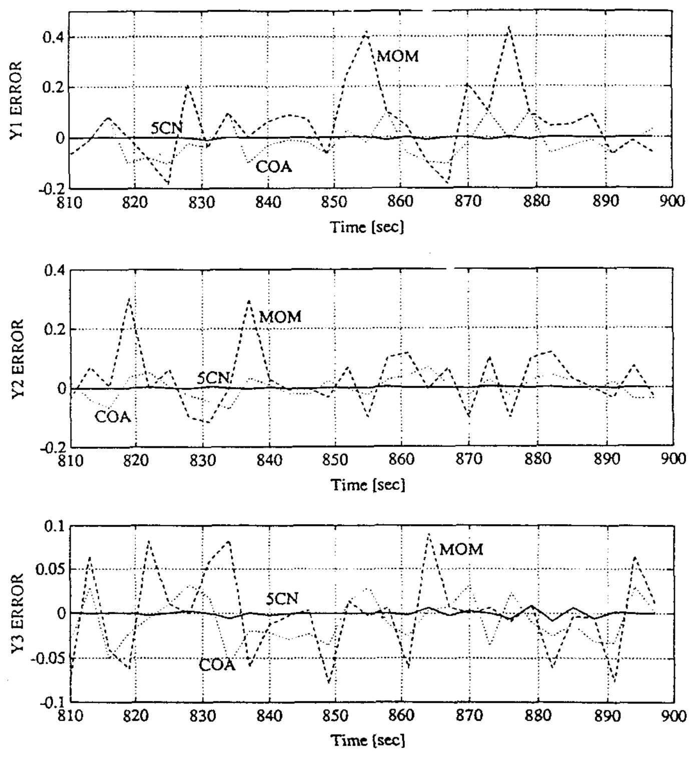


Figure 8: Forecasting errors of CoA, MoM, and 5NN.

forecast in the algorithm that generates the forecast, or something of that nature!). However, we convinced ourselves that this was definitely not the case.

The success of the algorithm must be explained by the fact that it is heavily data–driven. In the MoM and CoA techniques, data were only used in the qualitative modeling stage, i.e., for the identification of an optimal mask. Once the mask had been found, the data were thrown away, and only the qualitative (crisp) model itself and the fuzzy membership functions of the inputs and outputs were preserved. The 5NN algorithm, on the other hand, exploits the available knowledge to its maximum extent both in the qualitative modeling and the qualitative simulation stages. This makes perfect sense in an inductive reasoning environment. Why throw precious knowledge away when it is needed anyway for a different purpose and is readily available?

On the other hand, many fuzzy system applications are based on deductive rather than inductive modeling. The qualitative relationships between the inputs and the outputs are determined on the basis of meta–knowledge, i.e., a general understanding of how the device is supposed to function, rather than on measurement data. In such applications, MoM and CoA can still be used, whereas 5NN no longer works since it simply won't predict anything without appropriate measurement data.

7 Conclusions

In this paper, a new fuzzy inferencing method, called the five–nearest–neighbors (5NN) rule, was presented. The example demonstrates that the new fuzzy inferencing technique performs better that the two most commonly used defuzzification methods in the context of inductive reasoning applications. The main advantages of this new method are: (1) it performs exceedingly well in a data–driven environment; (2) it works well even if the number of discrete states is small (three or four), whereas the traditionally used fuzzy inferencing techniques require considerably more discrete classes; (3) it is economical both with respect to number–crunching and memory requirements; (4) the process of fuzzy inferencing is clearly separated from that of defuzzification. Its main limitation is that it cannot be applied without appropriate measurement data.

The choice of the fuzzy inferencing (defuzzification) method can play a significant role in many areas of fuzzy system applications. The influence of the fuzzy inferencing process can dictate both the behavior and the robustness of a fuzzy controller. In fact, a major motivation to study the defuzzification process stems from applications of fuzzy set theory to the field of process control. During recent years, the number of fuzzy control applications has grown rapidly. Yet, most references don't pay sufficient attention to the defuzzification process. An ill–conceived defuzzification method employed by a fuzzy controller may contribute significantly to the deterioration of the inferences made by the fuzzy reasoner. The importance of the defuzzification process in dynamical aspects of fuzzy controllers has been discussed in [7]. The influence of selecting irregular fuzzy intervals in the process of defuzzification, and the influence of the number of fuzzy rules in the convergence of fuzzy controllers have been studied in [12] and [2], respectively. However, neither these nor any other studies have taken into account data–driven approaches to fuzzy controller synthesis. In this paper, it was shown that the fuzzy inferencing (defuzzification) process can be significantly improved in such cases by adequately exploiting the available data.

Although the example discussed in this paper is linear, this choice was only made for reasons of simplicity. The linearity of the application is not important for the success of the technique. Several nonlinear applications of qualitative simulations for the purposes of fault detection, fault diagnosis, and fuzzy control have been successfully completed using the SAPS–II inductive reasoning environment, all applying the five–nearest–neighbors fuzzy inferencing