A New Fuzzy Inference Method for Systems with Large Number of Rules

Hoda Moodi
E.E. Department, Ferdowsi Uni. of Mashhad
hodamoodi@yahoo.com

Danyal Bustan[1]
E.E. Department, Ferdowsi Uni. of Mashhad
dbustan@yahoo.com

Habib Rajabi Mashhadi
E.E. Department, Ferdowsi Uni. of Mashhad
Rajabi_mashhadi@yahoo.com

Abstract:
In this paper, a new fuzzy inference method which is suitable for systems with large number of rules, is proposed. As we know, there are two well-known fuzzy inference systems, Mamdani and TSK. Each one has its own drawbacks and advantages but both of them have been encountered with problem while tuning their parameters especially when there is large number of rules in the system. Mamdani type systems faced to a huge amount of calculation and TSK type faced to large number of parameters. In our proposed method a combination of these two systems, is used. So it has small number of parameters for tuning as Mamdani has and it is as fast as TSK. We called this system Extended TSK because it is based upon it.

Keywords: Fuzzy modeling, parameter tuning

1. Introduction

As fuzzy theory is developed, the research on fuzzy modeling, which describes a real system very successfully with its nonlinear property, is conducted actively [1]. Generally speaking, fuzzy models have advantages of excellent capability to describe a given system and intuitive persuasion toward human operators over linear models. Many studies regarding fuzzy modeling have been reported heretofore. Some of them are based on the pattern-recognition techniques [2], [3], [4] and others are based on system programming theory [5], [6]–[8]. In [9], a fuzzy model identification

[1] Corresponding Author

algorithm using referential fuzzy sets is proposed. In addition, other methods such as neural-network-based methods are suggested [10]–[13].

One of the most outstanding models among them is the model suggested by Takagi and Sugeno in 1985 [5]. The model is based on the system programming method, has excellent capability to describe a given unknown system, and is very suitable for model-based control [14]–[16]. However, this model has huge number of parameters for tune. So tuning this identification algorithm is too complex to implement in a digital computer and consumes much time due to its complexity, which keeps the model from being used in practical applications.

But model which suggested by Mamdani [17] has less parameters than TSK but it consumes a lot of time for reasoning. Now we encountered with a problem: which one to choose for tuning a system: less parameters or less time?

In this paper, a new fuzzy modeling algorithm combining the advantages of both [5] and [17] is proposed: it can express a given unknown system with a small number of fuzzy rules as well as Takagi and Segno's model and is easy to implement. The suggested model has the same structure as that of Takagi and Segno's model and its identification mimics the simple identification procedure of Sugeno.

This paper is organized as follows. In Section 2, fuzzy models of [5] and [17] are explained and their problems are presented. In Section 3, an in-depth explanation on the new fuzzy modeling algorithm is given. In Section 4, one example is given to illustrate the new algorithm and compare it with others (AFM and ANFIS). In Section 5, the conclusion is presented.

2. Fuzzy logic[20]

Fuzzy logic was invented by Zadeh [18] as an extension of Boolean logic. While classical logic assigns to a variable either the value "1" for "true" or the value "0" for "false", fuzzy logic allows one to assign to a variable any value in the interval [0,1].This extension is motivated by the observation that human often think and communicate in a vague and uncertain way – partly because of insufficient information, partly due to human nature. In order to deal with such statements in a rule-based form new approximation reasoning mechanism based on fuzzy logic had to be developed [19].

Fuzzy logic allows one to express and process relationships in form of rules. In fuzzy rules the linguistic variables are expressed in the form of fuzzy sets. The linguistic input variables are labeled by linguistic terms. These linguistic terms are defined by their associated membership functions (MFs). These MFs define the degree of membership of a specific variable to the fuzzy sets. MFs are usually functions of a single variable. So fuzzy systems typically deal with each input separately, and inputs are combined in the rules by logic operators such as AND and OR.

In the fuzzy system only the degrees of membership are further processed. This can be seen as a nonlinear transformation of the inputs. Often information is lost during this procedure for example in a trapezoidal MF with parameters *(a,b,c,d)* it does not matter whether the input is *b* or *c* or some value in between, because the degrees of membership are not affected.

After the degrees of membership for each linguistic statement have been evaluated, the next step is to combine these values by logic operators such as AND and OR.

With the logic operators it is possible to combine the degrees of membership of all linguistic statements within the rule premise. Obviously, the outcome of a fuzzy system is strongly dependent on the specific choice of operators.

The combination of the degrees of membership of all linguistic statements is called the degree of rule fulfillment or rule firing strength, since it expresses how well a rule premise matches a specific input value.

Care must be taken that a whole feasible input space is covered by rules in order to avoid the situation where all fuzzy rules are inactive.

After the degree of rule fulfillment has been calculated for all rules the consequents have to be evaluated and accumulated to generate one output of the fuzzy system. Finally for must applications this fuzzy system output, which generally is a fuzzy set, has to be *defuzzified* in order to obtain one crisp output value.

2.1. Mamdani Fuzzy systems

Mamdani fuzzy systems, also known as *linguistic fuzzy systems* posses rules in the form

$$R_i : IF\ u_1 = A_{i1}\ AND\ u_2 = A_{i2}\ AND \ldots\ u_p = A_{ip}\ THEN\ y = B_i\ , \qquad (1)$$

Where $u_1,\ldots,\ u_p$ are the p inputs of the fuzzy system gathered in the input vector $\boldsymbol{u}$, and y is the output, the index $i = 1,\ldots,M$ runs over all M fuzzy

rules, A_{ij} denotes the fuzzy set used for input u_j in rule i, And B_i is the fuzzy set for the output in rule i. On the first sight, such linguistic fuzzy systems are most appealing because both the inputs and output are described by linguistic variables. However, the analysis of the fuzzy inference will show that quit complex computation are necessary for the evaluation of such a linguistic fuzzy system. The following steps must be carried out:

Fuzzification → Aggregation → Activation → Accumulation → Defuzzification

The Fuzzification uses the MFs to map crisp inputs to the degrees of membership. The aggregation combines the individual linguistic statements to the degree of fulfillment. Both steps are identical for all types of fuzzy systems discussed here, and have been explained in the previous section. The last three steps depend on the fuzzy system type considered.

In the fuzzification phase the degrees of membership for all linguistic statements are calculated. They will be denoted by $\mu_{ij}(u_j)$, $i = 1,\ldots,M$, $j = 1,\ldots,p$.

In the aggregation phase these degrees of membership are combined according to the fuzzy operators. When the fuzzy system is in conjunction form and the product operator is applied as t-norm the degree of fulfillment of rule i becomes

$$\mu_i(\boldsymbol{u}) = \mu_{i1}(u_1).\ \mu_{i2}(u_2).\ \ldots.\ \mu_{ip}(u_p)\ . \qquad (2)$$

In the activation phase these degrees of fulfillment are utilized to calculate the output activation of the rules. This can, for example, be done by cutting the output MFs at the degree of rule fulfillment, i.e. ,

$$\mu_i^{act}(\ ,y) \quad \min[\mu_i(\ ),\mu_i(y)],$$

```
ERROR: rangecheck
OFFENDING COMMAND: .buildcmap

STACK:

-dictionary-
/WinCharSetFFFF-V2TT621301FBt
/CMap
-dictionary-
/WinCharSetFFFF-V2TT621301FBt
```