Fuzzy Systems Modeling: An Introduction

Young Hoon Joo
Kunsan National University, Korea

Guanrong Chen
City University of Hong Kong, China

INTRODUCTION

The basic objective of system modeling is to establish an input-output representative mapping that can satisfactorily describe the system behaviors, by using the available input-output data based upon physical or empirical knowledge about the structure of the unknown system.

BACKGROUND

Conventional system modeling techniques suggest constructing a model described by a set of differential or difference equations. This approach is effective only when the underlying system is mathematically well-defined and precisely expressible. They often fail to handle uncertain, vague or ill-defined physical systems, and yet most real-world problems do not obey such precise, idealized, and subjective mathematical rules. According to the incompatibility principle (Zadeh, 1973), as the complexity of a system increases, human's ability to make precise and significant statements about its behaviors decreases, until a threshold is reached beyond which precision and significance become impossible. Under this principle, Zadeh (1973) proposed a modeling method of human thinking with fuzzy numbers rather than crisp numbers, which had eventually led to the development of various fuzzy modeling techniques later on.

MAIN FOCUS OF THE CHAPTER

Structure Identification

In structure identification of a fuzzy model, the first step is to select some appropriate input variables from the collection of possible system inputs; the second step is to determine the number of membership functions for each input variable. This process is closely related to the partitioning of input space. Input space partitioning methods are useful for determining such structures (Wang & Mendel, 1996).

Grid Partitioning

Figure 1 (a) shows a typical grid partition in a two-dimensional input space. Fuzzy grids can be used to generate fuzzy rules based on system input-output training data. Also, a one-pass build-up procedure can avoid the time-consuming learning process, but its performance depends heavily on the definition of the grid. In general, the finer the grid is, the better the performance will be. Adaptive fuzzy grid partitioning can be used to refine and even optimize this process. In the adaptive approach, a uniformly partitioned grid may be used for initialization. As the process goes on, the parameters in the antecedent membership functions will be adjusted. Consequently, the fuzzy grid evolves. The gradient descent method may then be used to optimize the size and location of the fuzzy grid regions and the overlapping degree among them. The major drawback of this grid partition method is that the performance suffers from an exponential explosion of the number of inputs or membership functions as the input variables increase, known as the "curse of dimensionality," which is a common issue for most partitioning methods.

Tree Partitioning

Figure 1 (b) visualizes a tree partition. The tree partitioning results from a series of guillotine cuts. Each region is generated by a guillotine cut, which is made entirely across the subspace to be partitioned. At the $(k-1)$st iteration step, the input space is partitioned into k regions. Then a guillotine cut is applied to one of these regions to further partition the entire space into $k + 1$ regions. There are several strategies for determining which dimension to cut, where to cut at each step, and when to stop. This flexible tree partitioning algorithm resolves the problem of curse of dimensionality. However, more membership functions are needed for each input variable, and they usually do not have clear linguistic meanings; moreover, the resulting fuzzy model consequently is less descriptive.

Scatter Partitioning

Figure 1 (c) illustrates a scatter partition. This method extracts fuzzy rules directly from numerical data (Abe & Lan, 1995). Suppose that a one-dimensional output, y, and an m-dimensional input vector, $\underline{x}$, are available. First, the output space is divided into n intervals, $[y_0, y_1]$, $(y_1, y_2]$, …, $(y_{n-1}, y_n]$, where the ith interval is called "output interval i." Then, activation hyperboxes are determined, which define the input region corresponding to the output interval i, by calculating the minimum and maximum values of the input data for each output interval. If the activation hyperbox for the output interval i overlaps with the activation hyperbox for the output interval j, then the overlapped region is defined as an inhibition hyperbox. If the input data for output intervals i and/or j exist in the inhibition hyperbox, then within this inhibition hyperbox one or two additional activation hyperboxes will be defined. Moreover, if two activation hyperboxes are defined and they overlap, then an additional inhibition hyperbox is further defined. This procedure is repeated until overlapping is resolved.

Parameters Identification

After the system structure has been determined, parameters identification is in order. In this process, the optimal parameters of a fuzzy model that can best describe the input-output behavior of the underlying system are searched by optimization techniques.

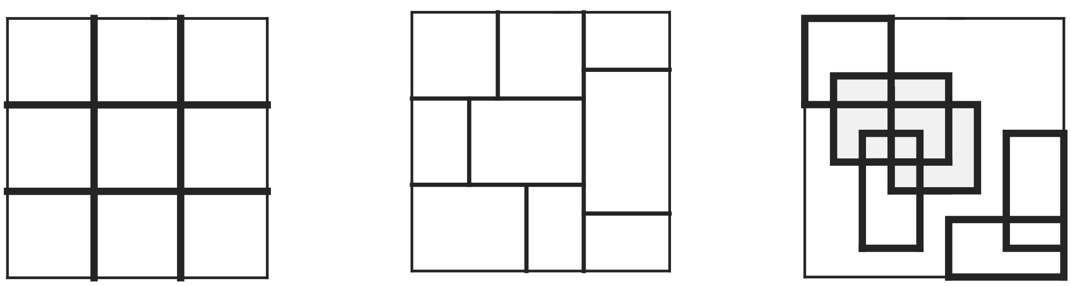
Sometimes, structure and parameters are identified under the same framework through fuzzy modeling. There are virtually many different approaches to modeling a system using the fuzzy set and fuzzy system theories (Chen & Pham, 1999, 2006), but the classical least-squares optimization and the general Genetic Algorithm (GA) optimization techniques are most popular. They are quite generic, effective, and competitive with other successful non-fuzzy types of optimization-based modeling methods such as neural networks and statistical Monte Carlo.

An Approach Using Least-Squares Optimization

A fuzzy system can be described by the following generic form:

$$f(\underline{x}) = \sum_{k=1}^{m} \alpha_k g_k(\underline{x}) = \underline{\alpha}^T \underline{g}(\underline{x}) \tag{1}$$

Figure 1. Three typical MISO partitioning methods



(a) fuzzy grid *(b) tree partition* *(c) scatter partition*