# Generating Fuzzy Rules by Learning from Examples

EE658A Course Project Report

Atreya Goswami
*Department of Computer Science and Engineering*
*Indian Institute of Technology Kanpur*
Kanpur, India
atreya@iitk.ac.in

***Abstract*—This report reviews a general method of generating fuzzy rules by learning from examples and its application to two benchmark problems. The method consists of five steps: Step 1 divides the input and output spaces of the given numerical data into fuzzy regions; Step 2 generates fuzzy rules from the given data; Step 3 assigns a degree of each of the generated rules for the purpose of resolving conflicts among the generated rules; Step 4 creates a combined fuzzy rule base based on both the generated rules and linguistic rules of human experts; and, Step 5 determines a mapping from input space to output space based on the combined fuzzy rule base using a defuzzifying procedure. Implementation of the method in truck backer-upper control problem and Mackey-Glass chaotic time series prediction problem are also presented.**

## INTRODUCTION

A new approach to generate fuzzy rules from numerical data, was introduced by Wang and Mendel in [1]. This report reviews the proposed approach and presents its application to two benchmark control problems - the truck backer-upper control problem and the Mackey-Glass chaotic time series prediction problem.

Given a complex control system consisting of the environment and a human controller, the task is to design a control system to replace the human controller. The environment cannot be modelled mathematically due to its complexity, hence the problem needs to be solved using a *model-free* design strategy.

### *Background*

For most control and signal processing problems, information available can be classified into two kinds - numerical data i.e., input-output pairs sampled from successful control, and experience of the human controller expressed in the form of linguistic "IF-THEN" rules.

However contemporary intelligent control and signal processing approaches were predominantly heuristic in nature - they used some *ad hoc* method to combine these two kinds of information. The main issues with this approach are 1) the approach is problem specific - it might not be suitable for a general class of problems, and 2) difficulty in theoretical analysis of the approach due to its problem dependant nature.

Previous research in this area showed that in most control design problems, neither of the two kinds of information is usually sufficient. While fuzzy control was found to be useful in modelling linguistic rules as in [2], neural control was shown to be effective in utlizing numerical data as in [3]. Hence, most contemporary fuzzy controllers used only expert annotated linguistic rules, whereas numerical data was learnt using neural controllers.

### *Contribution*

Wang and Mendel, in [1], propose a general method for combining both numerical and linguistic information into a single framework - a fuzzy rule base. The key ideas of the proposed approach are to generate fuzzy rules from numerical data pairs, collect these fuzzy rules and the linguistic fuzzy rules into a common fuzzy rule base, and, finally, design a control or signal processing system based on this combined fuzzy rule base.

## APPROACH

Given a set of $N$ desired input-output data pairs $\{(x_1^{(i)}, x_2^{(i)}; y^{(i)})\}_{i=1}^{N}$, where $x_1$, $x_2$ are inputs and $y$ is the output, the task is to generate a set of fuzzy rules and use them to determine a mapping $f : (x_1, x_2) \rightarrow y$. The proposed approach consists of the following five steps:

### *Step 1 - Dividing input and output spaces into Fuzzy regions*

First, let us assume that the domain intervals of $x_1, x_2$ and $y$ are $[x_1^-, x_1^+], [x_2^-, x_2^+]$ and $[y^-, y^+]$ respectively. We then divide each domain interval into $2N + 1$ regions (SN, ... , S2, S1, CE, B1, B2, ... , BN) and assign each region a fuzzy membership function as shown in Fig. 1.

### *Step 2 - Generating Fuzzy Rules from given data pairs*

Next, we determine the degrees of the inputs $x_1, x_2$ and the output $y$ in each of the defined regions. We assign $x_1, x_2$ and $y$ to the regions with maximum degree. Hence, from the

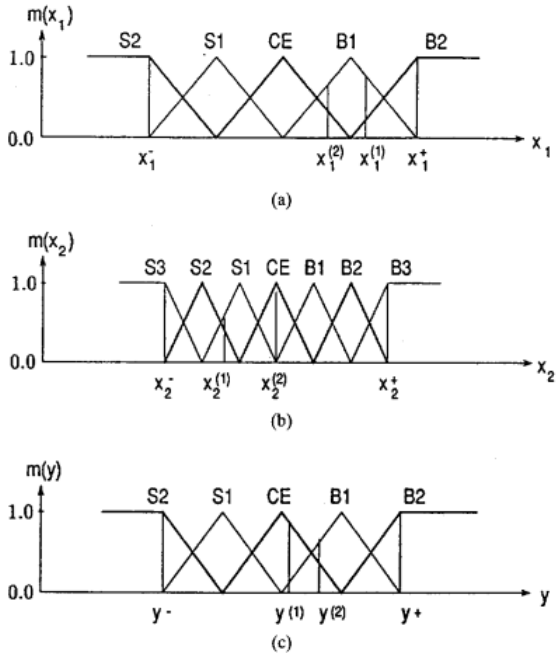


Fig. 1: Division of input and output spaces into fuzzy regions

assigned regions of $x_1, x_2$ and $y$, we obtain an "AND" rule. Every input-output data pair yields one such rule.

For example, let's say, degree of $x_1$ is maximum in region S1, degree of $x_2$ is maximum in region CE, and degree of corresponding $y$ is maximum in region S2. Then, from this input-output pair, we obtain the rule as follows:

Rule1: IF $x_1$ is S1 AND $x_2$ is CE, THEN $y$ is S2.

*Step 3 - Assigning a degree to each rule*

Since every data pair generates a rule, there will be lot of redundant and conflicting rules. To resolve this issue, let us consider a conflict group - i.e., set of rules with the same *antecedents* (IF part) having same or different *consequents* (THEN part). We assign a degree to all rules and accept only the rule with maximum degree from a conflict group. The rule obtained in Step 2 will be assigned a degree as follows:

$$D(\text{Rule1}) = m_{\text{S1}}(x_1)\, m_{\text{CE}}(x_2)\, m_{\text{S2}}(y)$$

Often we also multiply the "usefulness" $m^{(i)}$ of a given data pair (annotated by a human expert) in the expression for degree of a rule. Hence, we can redefine the degree of Rule1 as

$$D(\text{Rule1}) = m_{\text{S1}}(x_1)\, m_{\text{CE}}(x_2)\, m_{\text{S2}}(y)\, m^{(1)}$$

*Step 4 - Creating a combined Fuzzy Rule Base*

In this step, a combined fuzzy rule base is created by combining the rules obtained in Step 3 with linguistic fuzzy rules. Rules derived from either language rules or numerical data are assigned to a combined fuzzy rule base. Assuming that linguistic rules are also assigned a degree, in case of a conflict, only the rule with a higher degree will be accepted. Hence, this method codifies linguistic and numerical data into a single framework known as the combined fuzzy rule base, as illustrated in Fig. 2.

We know from Step 2 that fuzzy rules obtained from numerical data pairs will always be "AND" rules. However, linguistic fuzzy rules may be "OR" rules, in which case, all the boxes corresponding to the antecedents of the rule need to be filled with the output of the rule.

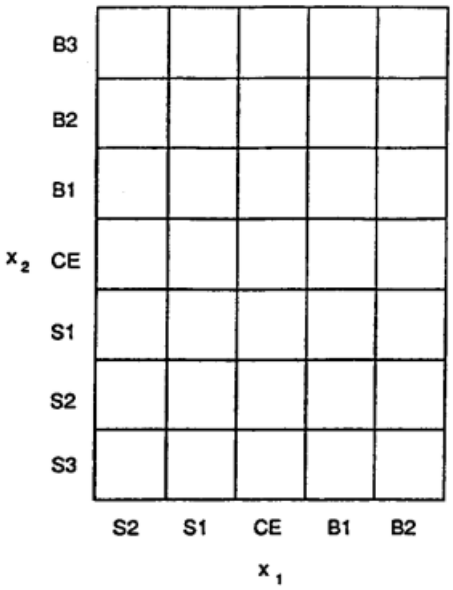


Fig. 2: Example of a fuzzy rule base box

*Step 5 - Determining a mapping based on the rule base*

Since we have created the combined fuzzy rule base, the "training" part of the control design is complete. Now for any new data pair $(x_1, x_2)$, we need to determine the output control $y$. We use the following defuzzification strategy to achieve this:

1) For each fuzzy rule, we multiply the degrees of $x_1$ and $x_2$ in the antecedents' corresponding regions to determine the degree of the output control corresponding to $(x_1, x_2)$.

$$m^i_{O^i} = m_{I^i_1}(x_1)\, m_{I^i_2}(x_2)$$

where, $I^i_1, I^i_2$ are the input regions corresponding to the inputs $x_1$ and $x_2$ and $O^i$ denotes the output region in the $i^{th}$ rule.

2) Now, we use the *centroid defuzzification formula* to determine the output control $y$.

$$y = \frac{\sum_{i=1}^{K} m^i_{O^i} \bar{y}^i}{\sum_{i=1}^{K} m^i_{O^i}}$$

where, $K$ is the number of fuzzy rules in the combined fuzzy rule base, and $\bar{y}^i$ denotes the *center value*[1] of output region $O_i$.

[1]Center of a fuzzy region is defined as the point that has the smallest absolute value among all the points at which the membership function for this region has membership value equal to one

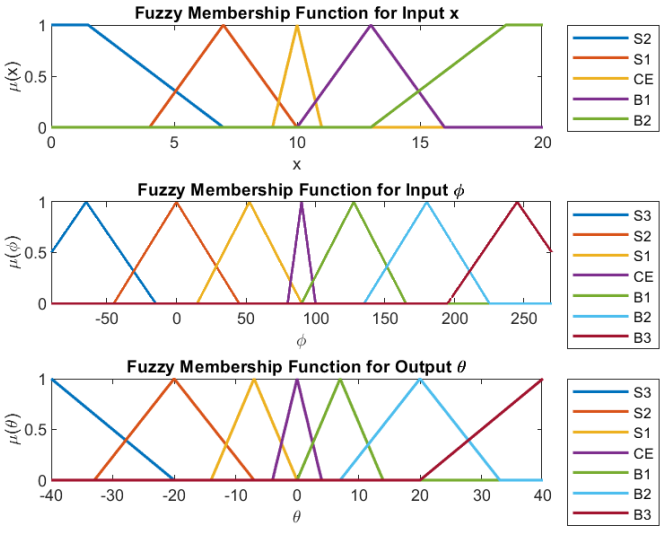


Fig. 3: Fuzzy membership functions for truck backer-upper problem

## Fuzzy System as Universal Approximator

The proposed five-step approach generates a mapping from input space to output space. For the general $n-$input one output case, the mapping can be represented by

$$y = f(\boldsymbol{x}) = \frac{\sum_{i=1}^{K} \bar{y}^i m^i}{\sum_{i=1}^{K} m^i} = \frac{\sum_{i=1}^{K} \bar{y}^i \prod_{1\le j\le n} \left[m_j^i(x_j)\right]}{\sum_{i=1}^{K} \prod_{1\le j\le n} \left[m_j^i(x_j)\right]}$$

where, $m_j^i$ is the membership function of the $i^{th}$ rule for the $j^{th}$ component of the input vector, and $\bar{y}^i$ is the center value of the output region of the $i^{th}$ rule. The generated fuzzy system can approximate any real continuous function $g : Q \to \mathbb{R}$ and thus acts as a *universal approximator* where the compact set $Q$ is defined as

$$Q = [a_1, b_1] \times [a_2, b_2] \times [a_3, b_3] \times \cdots [a_n, b_n] \subset \mathbb{R}^n$$

## Application to Truck Backer-Upper Control

Backing up a truck to its loading dock is a non-linear control problem, hence traditional control systems design methods are ineffective. We apply the proposed "numerical-fuzzy" approach to this control problem to combine the information obtained from sampled successful truck operation data, and the experience of an expert driver, expressed in terms of linguistic rules.

We define the control system using three state variables $(x, y, \phi)$ and the output control is $\theta$, where $(x, y)$ is the coordinate of the midpoint of the rear of the truck, $\phi$ is the angle of the truck with the horizontal. $y$ is not considered to be an input variable as the area is considered to be unbounded in the $y$ direction. Also we impose the constraints $x \in [0, 20]$, $\phi^\circ \in [-90, 270]$ and $\theta^\circ \in [-40, 40]$. The task is to design a control system to back up the truck from some initial configuration $(x, \phi)$ to the loading dock configuration $(x_f, \phi_f{}^\circ) = (10, 90)$ covering unit distance in each step. Kinematics of the truck are assumed to be governed by the following approximate equations:

$$x(t+1) = x(t) + \cos[\phi(t) + \theta(t)] + \sin[\phi(t)] \sin[\theta(t)]$$
$$y(t+1) = y(t) + \sin[\phi(t) + \theta(t)] - \cos[\phi(t)] \sin[\theta(t)]$$
$$\phi(t+1) = \phi(t) - \sin^{-1}\left[\frac{2\sin(\theta(t))}{b}\right]$$

Data tables for 14 sequences of desired trajectories $(x, \phi, \theta)$ with different initial configurations, and membership functions (Fig. 3) for each region of the variables have been taken from [4]. We have assumed the length of the truck $b = 4$.

In the first example, we used the input-output data pairs $(x, \phi; \theta)$ and membership functions to generate the fuzzy rule base (Fig. 4). In this case, we assumed that there were no linguistic rules. Then starting from three different initial configurations $(x, \phi^\circ) = (3, -30), (10, 220)$ and $(13, 30)$ (named Truck 1, Truck 2 and Truck 3 respectively) using the kinematic equations in step 5, we simulated the trajectory of the truck (Fig. 5) for 50 steps, within which the trajectory converged to the loading dock configuration.

| $\phi \backslash x$ | S2 | S1 | CE | B1 | B2 |
|---|---|---|---|---|---|
| S3 | S3 | S3 | X | X | X |
| S2 | S2 | S3 | S3 | S3 | X |
| S1 | B1 | S1 | S2 | S3 | S3 |
| CE | B2 | B2 | CE | S2 | S2 |
| B1 | B3 | B3 | B2 | B1 | S1 |
| B2 | X | B3 | B3 | B3 | B2 |
| B3 | X | X | X | B3 | B3 |

Fig. 4: Example 1 - Fuzzy Rule Base

In the second example, we use only first three data pairs from the 14 data tables to generate the fuzzy rule base (Fig. 6). Then we repeat the simulations for 100 steps as in the first example for Trucks 1,2 and 3 to get the trajectories (Fig. 7). However, we observe that the trajectory of Truck 2 does not even start, and even after 100 steps the trajectories of Truck 1 and 3 do not converge to the loading dock configuration.

Now, we add the following 4 rules to the existing rule base in Fig. 6 to derive the combined fuzzy rule base (Fig. 8).
Linguistic fuzzy rules:

IF $x$ is CE and $\phi$ is S1, THEN $\theta$ is S2.

IF $x$ is CE and $\phi$ is CE, THEN $\theta$ is CE.

IF $x$ is CE and $\phi$ is B1, THEN $\theta$ is B2.

IF $x$ is CE and $\phi$ is B2, THEN $\theta$ is B3.

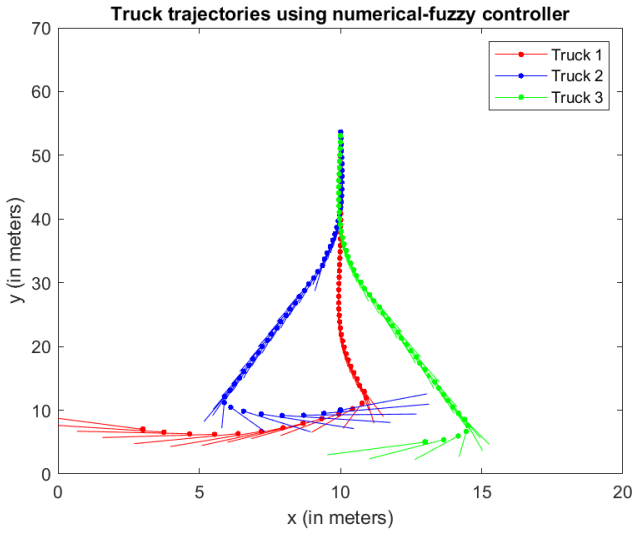


Fig. 5: Example 1 - Trajectory of trucks using complete data

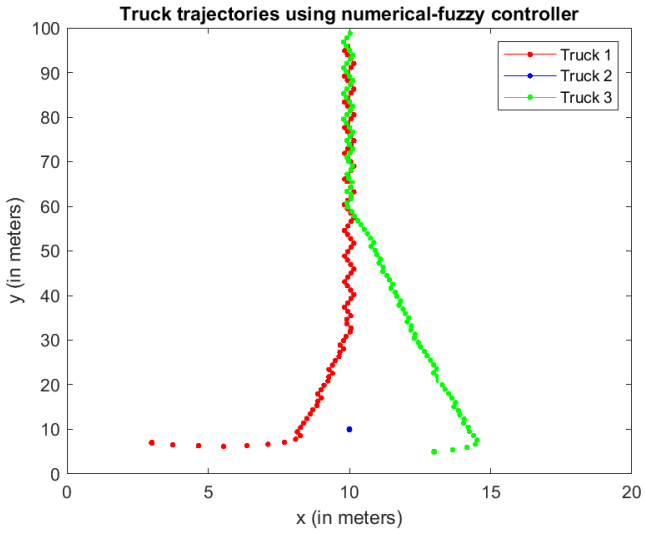


Fig. 7: Example 2 - Trajectory of trucks using truncated data

| $\phi \backslash x$ | S2 | S1 | CE | B1 | B2 |
|---|---|---|---|---|---|
| S3 | S3 | S3 | X | X | X |
| S2 | S2 | S3 | X | S3 | X |
| S1 | B2 | S3 | X | S3 | X |
| CE | B2 | B2 | X | S2 | S2 |
| B1 | X | B3 | X | B1 | S2 |
| B2 | X | B3 | X | B3 | B2 |
| B3 | X | X | X | B3 | B3 |

Fig. 6: Example 2 - Fuzzy Rule Base (with truncated data)

| $\phi \backslash x$ | S2 | S1 | CE | B1 | B2 |
|---|---|---|---|---|---|
| S3 | S3 | S3 | X | X | X |
| S2 | S2 | S3 | X | S3 | X |
| S1 | B2 | S3 | S2 | S3 | X |
| CE | B2 | B2 | CE | S2 | S2 |
| B1 | X | B3 | B2 | B1 | S2 |
| B2 | X | B3 | B3 | B3 | B2 |
| B3 | X | X | X | B3 | B3 |

Fig. 8: Example 2 - Fuzzy Rule Base (with truncated data and added linguistic rules)

Then we repeat the simulations for Truck 1,2 and 3 to get the trajectories (Fig. 9), which converge to the loading dock state in 80 steps.

## Application to Time Series Prediction

The Mackey-Glass chaotic time series is derived from the following delay differential equation:

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t)$$

We generated an approximation of the time series[2] with $\tau = 30$ (Fig. 10). For the time series $z(k)(k = 1, 2, 3...)$, given the values of $z(k-m+1), z(k-m+2), ..., z(k)$, the task is to determine $z(k+l)$. Specifically assuming that $z(1), z(2), ..., z(M)$ are given, we can derive the following $M - m$ data pairs to generate the fuzzy rule base.

$$[z(M-m), \cdots, z(M-1); z(M)]$$
$$[z(M-m-1), \cdots, z(M-2); z(M-1)]$$
$$\vdots$$
$$[z(1), \cdots, z(m); z(m+1)]$$

In the paper, $m = 9$ and $l = 1$ have been assumed. We apply the numeric fuzzy approach to the prediction of the time series $x(t)$. In Step 1, we divide the domain interval of $x(t)$, taken as $[0.2, 1.4]$, into $2N+1$ fuzzy regions. We have repeated the experiment taking $N = 3, 7, 14$ for both examples. In the first example, we use 700 data points $x(1)$ to $x(700)$ to construct the fuzzy rule base. We then use Step 5 to predict the next 300

[2]Marco Cococcioni (2023). Mackey-Glass time series generator (https://www.mathworks.com/matlabcentral/fileexchange/24390-mackey-glass-time-series-generator), MATLAB Central File Exchange. Retrieved March 20, 2023.

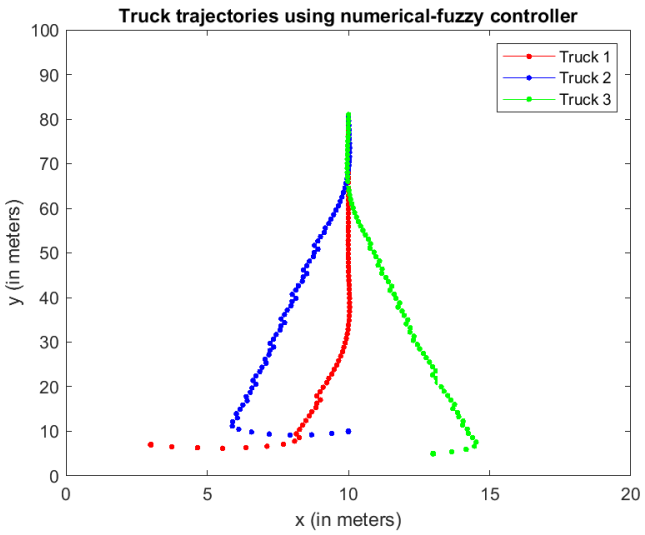


Fig. 9: Example 2 - Trajectory of trucks using truncated data and added linguistic rules

points $x(701)$ to $x(1000)$ (Fig. 11). In the second example, we use only 200 data points $x(501)$ to $x(700)$ to construct the fuzzy rule base, and use it to predict $x(701)$ to $x(1000)$ (Fig. 12).

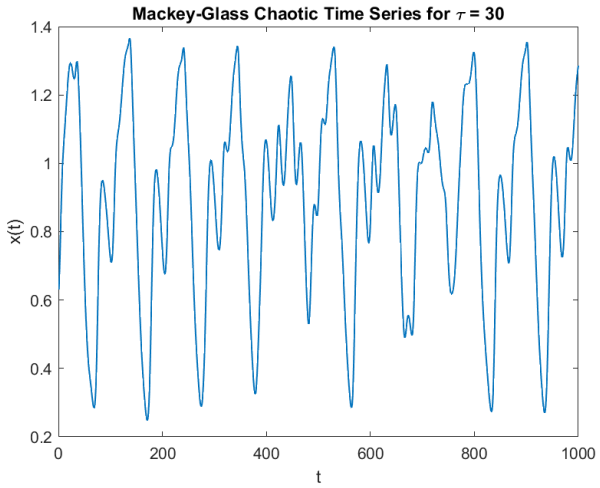


Fig. 10: A section of the Mackey-Glass chaotic time series

## Conclusion

In this report, we revisited a generic technique for producing fuzzy rules from numerical data. This technique can be applied generally to bring together linguistic and numerical data into a single framework, or fuzzy rule base. This fuzzy rule base consists of two types of fuzzy rules: those that were received from experts and others that were created using the technique described in this paper utilising measured numerical data.

The main advantages of the proposed approach are:

1) It provides us with a general method to combine measured numerical information and human linguistic information into a common framework, which can be used to develop some theoretically analyzable control algorithms that use both numerical and linguistic information.

2) It is a simple and straightforward one-pass build-up procedure; hence, no time-consuming iterative training is required, so that it requires much less construction time than a comparable neural network.

3) There is a lot of freedom in choosing the membership functions; this provides us with lots of flexibilities to design systems according to different requirements.

4) It can perform successful control for some cases where neither a pure neural network nor a pure fuzzy control can.

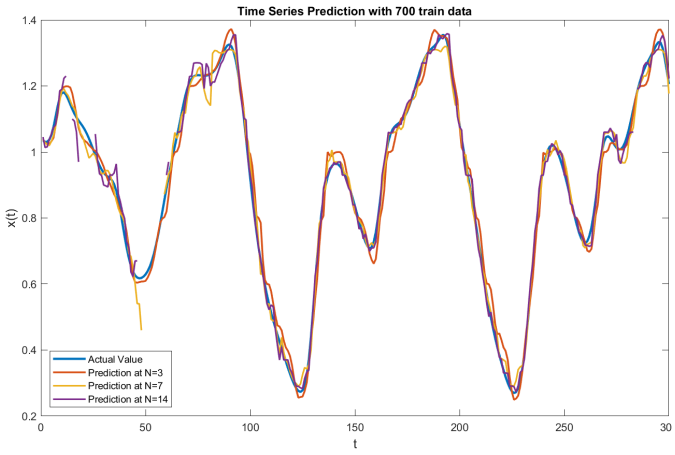


Fig. 11: Prediction of the chaotic time series from $x(701)$ to $x(1000)$ using the numerical-fuzzy predictor when 700 training data (from $x(1)$ to $x(700)$) are used.

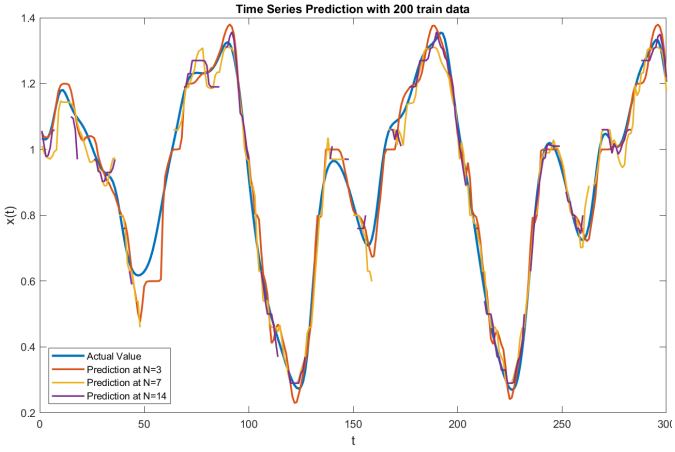


Fig. 12: Prediction of the chaotic time series from $x(701)$ to $x(1000)$ using the numerical-fuzzy predictor when 200 training data (from $x(501)$ to $x(700)$) are used.

## References

[1] L-X Wang and Jerry M Mendel. “Generating fuzzy rules by learning from examples”. In: *IEEE Transactions on systems, man, and cybernetics* 22.6 (1992), pp. 1414–1427.

[2] S-G Kong and Bart Kosko. “Comparison of fuzzy and neural truck backer-upper control systems”. In: *1990 IJCNN International Joint Conference on Neural Networks*. IEEE. 1990, pp. 349–358.

[3] Derrick Nguyen and Bernard Widrow. “The truck backer-upper: An example of self-learning in neural networks”. In: *Advanced neural computers*. Elsevier, 1990, pp. 11–19.

[4] Li-Xin Wang. “Generating fuzzy rules from numerical data, with applications”. In: *University of Southern California, tech. report* (1991).