Mackey-Glass time series prediction
Python for scientific computation
ČVUT

Adam Peichl

January 2017

1 Introduction

The purpose of this work is to create a fuzzy system for time series prediction for Mackey-Glass delay differential equation:

$$\frac{dx(t)}{dt} = \beta \frac{x(t-\tau)}{1+x^n(t-\tau)} - \gamma x(t) \tag{1}$$

The goal is in time t_1 to predict $x(t_1+1)$, we choose[1] $\tau = 30$. We need to set $n \sim 10$, because that is the value where chaos appears. The remaining two parameters are set as follows: $\beta = 0.2$ and $\gamma = 0.1$. It is because I want to compare my results with results in a book called A course in fuzzy systems and control by Li-Xin Wang.

2 Implementation of Mackey-Glass

We can approximate function (1) like this:

$$x(t+t_1) = x(t) + t_1 \frac{dx(t)}{dt} \tag{2}$$

We put equations (1) and (2) together and $t_1 = 1$, since we go step by step by samples. The following equation is our approximation:

$$x(t+1) = x(t) + \beta \frac{x(t-\tau)}{1+x^n(t-\tau)} - \gamma x(t) \tag{3}$$

Put our constants in equation (3):

$$x(t+1) = x(t) + \frac{0.2x(t-30)}{1+x^{10}(t-30)} - 0.1x(t) \tag{4}$$

Which is the final form of our equation.
As an input set we use 30 random values from $(0,1)$.

[1] τ should be more than 17.

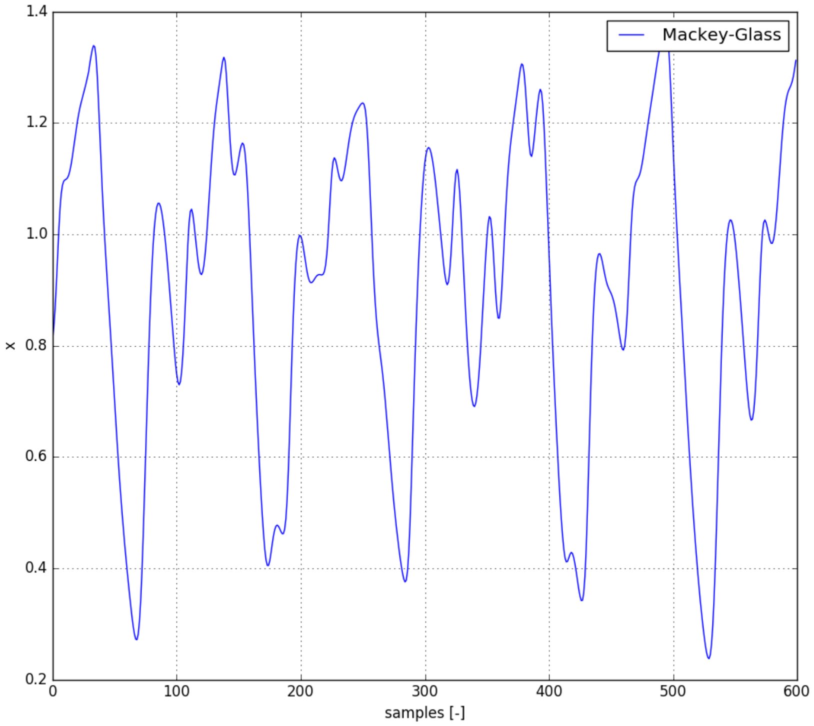

Figure 1: Mackey-Glass, delay diferencial equation

3 Design of a fuzzy system

The goal is to create a fuzzy system only from input-output data. We use only the first 300 values of x. The n represents number of statements in fuzzy IF-THEN rule. Thus we make a input-output pairs for creating fuzzy rules for look up table system only from input set.

$$\begin{array}{c} [x(0), x(1), ..., x(n-1); x(n)] \\ [x(1), x(2), ..., x(n); x(n+1)] \\ . \\ . \\ . \\ [x(299-n), x(299-n+1), ..., x(298); x(299)] \end{array}$$

We choose that our system is Look up table with product inference engine, singleton fuzzyfier and center average defuzzyfier:

$$f(x) = \frac{\sum_{l=1}^{M} \overline{y}^l (\prod_{i=1}^{n} \mu_{A_i^l}(x))}{\sum_{l=1}^{M} (\prod_{i=1}^{n} \mu_{A_i^l}(x))} \tag{5}$$

Where M is number of Fuzzy IF-THEN rules and n is number of fuzzy statements in one rule.
With fuzzy IF-THEN rules looking like this:

$$IF \bigwedge_{i=0}^{n} (x(k-n)\ is\ A_i^l)\ THEN\ x(k+1)\ is\ B^l \tag{6}$$

And we use triangular membership function:

$$\mu_{A_i^l}(x) = \begin{cases} 1 - \frac{|x - c_{A_i^l}|}{D}, & x \in (c_{A_i^l} - D, c_{A_i^l} + D) \\ 0, & else \end{cases} \tag{7}$$

4 Look up table prediction results

We use first 300 samples to create our Look up table, after that we use our created system to compute whole interval of 600 samples.

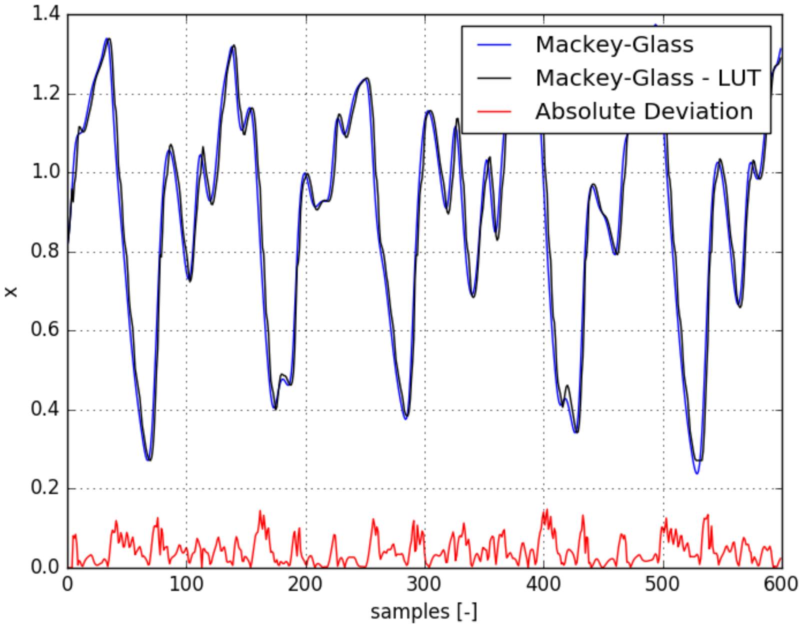

Figure 2: n=4, 7 fuzzy sets

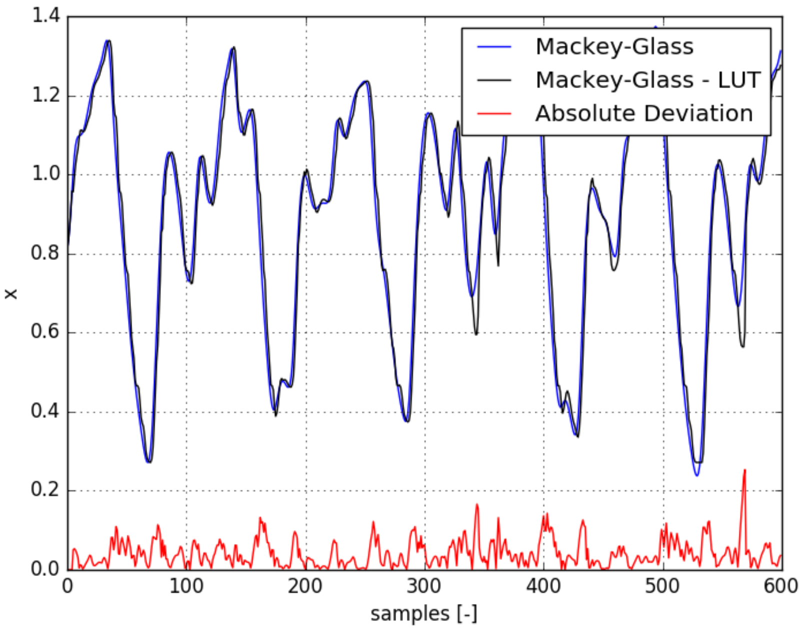

Figure 3: n=4, 12 fuzzy sets

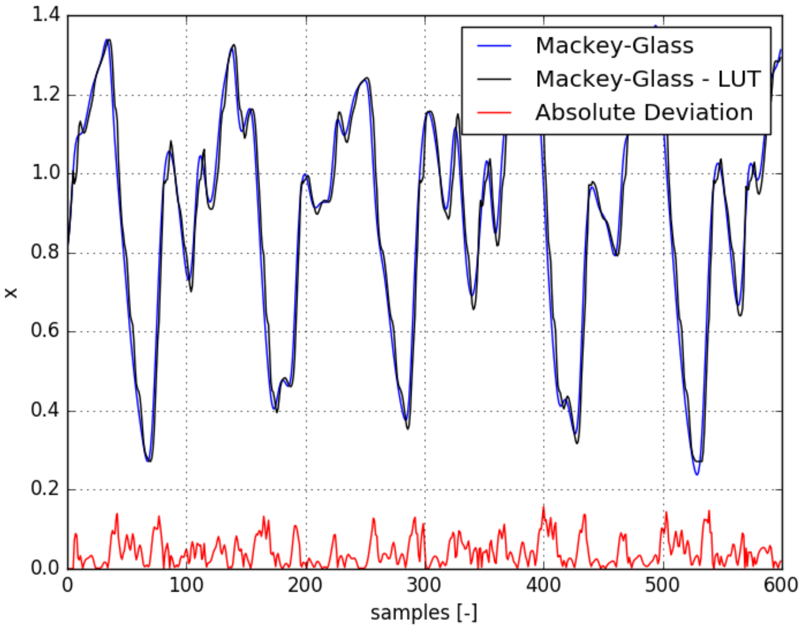

Figure 4: n=5, 7 fuzzy sets

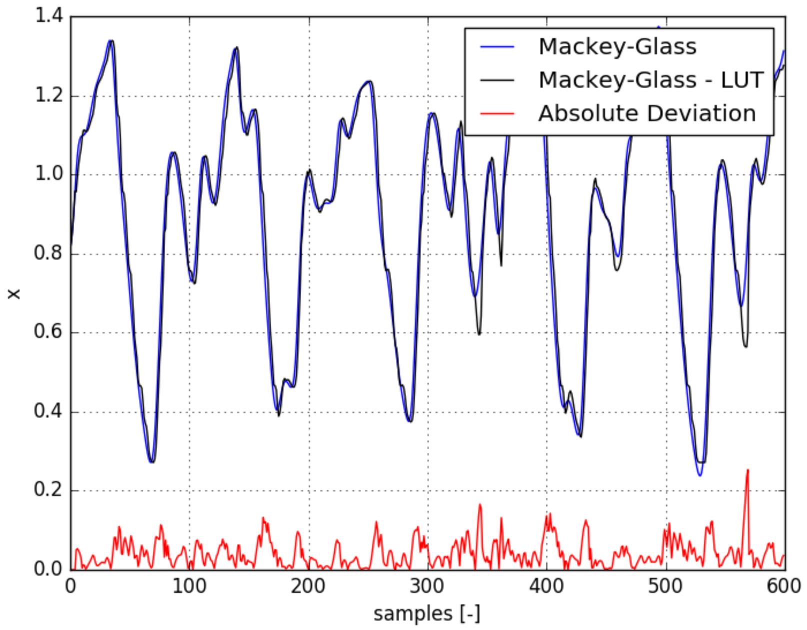

Figure 5: n=5, 12 fuzzy sets

5 Python tutorial

The goal of this python script is to implement Look up table for Mackey-Glass function to python, however, it can be used to create look up table fuzzy system from any time series-looking like data.

5.1 Script

Script name: Mackey-Glass.py
Python version used: 3.5.2
Modules nevessary to import: numpy, math, matplotlib.pyplot
Input data are created by given parameters in 5.3 Script does not have GUI and input parameters are entered in the script.

5.2 Look up table parameters

My look up table design has 3 adjustable parameters:

- **Training data - DATA** - 1xN array ,time series-like data, I use first 300 samples of Mackey-Glass function

- **Number of fuzzy statements - NumStatements** - integer, number of fuzzy statements in fuzzy IF-THEN rule
- **Number of fuzzy sets - NumFuzzySets** - integer, number of fuzzy sets in input and output area

5.3 Mackey-Glass parameters

From equation (1), we can see the adjustable parameters[2]:

- β **- Beta** - float
- γ **- Gamma** - float
- τ **- Tau** - integer
- **n - n** - float
- **y - y** - 1xN array ,time series-like data, it has to have length of at least Tau+1, becase Mackey-Glass is delay differencial equation[3]
- **samples - samples** - integer

5.4 Look up table fuzzy system as a python object

Object LUT will create look up table:

```
ourlookuptable = LUT(DATA, NumStatements , NumFuzzySets)
```

Now, a fuzzy system is already created and our object has two functions:

```
ourlookuptable.value(X)
```

Where: X is 1x(n-1) long array [x(t-n+2),x(t-n+3),...,x(t)] and this function returns predicted value of x(t+1) as a float.

```
ourlookuptable.compare(Y)
```

Where: Y is our whole time series and this function returns all predicted values from Y as a time series-like array, aditionaly it compares our prediction with Y measures absolute deviation and put all those function in one graph, wich is saved near our python code.

5.5 Using this script

This script is design to work in two ways:

[2] Just have in mind, chaos appears only for some values.

[3] You can use some random values from (0,1) as I did.

5.5.1 Running this script

You can fit in adjustable parameters from 5.2 and 5.3, the output of this code will be:

- Graph of Mackey-Glass originial function saved as Mackey_Glass_Original.png
- Graph with Mackey-Glass compared to our predicted values with absolute deviation saved for example as LUT_Mackey_Glass_5_Statements_7_Fuzzy_Sets.png, where number 5 represents NumStatemets and number 7 represents NumFuzzySets[4].

5.5.2 Using LUT as an object

You can use LUT python object to design your look up table from your data to try to predict completely different time series. (That is the main reason function .value(X) is implemented.

[4] figure 1 is original Mackey-Glass, figures 2-5 are compared ones.