Matlab Assignment for Knowledge-Based Control Systems (SC4081)

Introduction

This MATLAB-based assignment is a compulsory part of the course Knowledge-Based Control Systems (SC4081). It will be graded and the mark counts for 20% in the final grade of the course (the exam grade is 60% of the final grade, and the literature assignment grade is 20%). The assignment is carried out in groups of two[1] students, and should take around 25 hours per person to solve, depending on your experience with MATLAB and Simulink. It must be worked out in the form of a short written report (in English, one report per group), to be delivered along with the corresponding MATLAB/Simulink software by **Wednesday 6 April 2016, 12:00 (noon) at the latest**. Please deliver the report on paper personally either to the lecturer or to the assistant. In their absence you can drop the report in the dedicated box placed in front of the DCSC secretariat. Do not forget to include your names and student numbers on the title page of the report. Note that it is strictly forbidden to take over results from other students or make your results available to others.

Please include in your report complete listings of the MATLAB code (functions and scripts) and Simulink models that you developed for solving the assignment problems. In addition to that, please send your software as a ZIP file by e-mail to the course assistant Sachin Navalkar. His e-mail address, that you will need to obtain the files for the assignments as well, is **`S.T.Navalkar@tudelft.nl`**.

On 23/Mar/2016 from 15:45 to 17:45, there will be question hours for the Matlab assignment in Computerzaal B in TBM. You can bring your own computer here with you, or use the computers in the computer room.

Use MATLAB version 6.5 or higher and mention the version you used. It is not required that you provide an electronic version of the report document; however, if you decide to provide one, please deliver it in PDF format, also via email to the course assistant. Note that you still need to provide a paper version, as described above.

The assignment consists of three problems. The first one is an implementation of a fuzzy model in MATLAB. In the second one, you are asked to design in Simulink a fuzzy supervisor to improve the performance of a linear controller. The last problem concerns data-driven black-box modelling of an unknown system by using a feedforward neural network. For the last two problems, each group will receive their own process simulation model and their own data set.

Problem 1. Takagi–Sugeno fuzzy model

Implement a MATLAB function for a SISO (single–input, single–output) static Takagi–Sugeno system with trapezoidal antecedent membership functions and affine linear consequent functions. The input is x, output y, the parameters are the antecedent trapezoid vertices and the consequent parameters in each rule. This function should handle the singleton fuzzy system as a special case.

For the singleton model, implement a *second* MATLAB function to estimate the consequent singletons by using the linear least-squares method (assume that the antecedent parameters are given).

Choose an arbitrary univariate nonlinear function $y = f(x)$, generate a set of 100 input–output samples, and demonstrate the working of your two functions by fitting a singleton model to the data. In the report, include a plot of the result obtained with the above data. Discuss the result.

Do not use the Fuzzy Logic toolbox of MATLAB to solve this problem. Tip: the least-squares estimation can be best implemented by using the backslash (left-division) command, see `help mldivide`.

[1] If you absolutely cannot find a partner, you may work alone. Note that groups of three or more students are not allowed.

Problem 2. Fuzzy supervisory control

Note: To receive the files for this problem, send an e-mail to Sachin Navalkar (S.T.Navalkar@tudelft.nl). Please mention the names and initials of the two members of your groups, the student numbers and an e-mail address at which you wish to receive the files.

A nonlinear dynamic process is controlled by a linear controller. As the closed-loop behavior is not satisfactory, you are asked to design a fuzzy supervisor to improve the performance. Typically, the settling time and the overshoot of the closed-loop system must be maintained within specified limits for a given setpoint range. You will find the actual required values in the Simulink models you will receive.

The process model is a 'black-box', so you cannot inspect the equations, but you are allowed to carry out any open-loop or closed-loop simulation experiments in order to learn more about its properties. To design the supervisor, it is also possible (but not compulsory) to use the MATLAB's trimming and linearization functions `trim` and `linmod`. In the report, include the rule base, the membership functions and other parameters of the supervisor. Explain the rationale behind the rule base (inputs, membership functions, etc.) and the method you used to tune the parameters. Compare the closed-loop performance before and after introducing the supervisor. Discuss the results.

A set of functions implementing fuzzy inference in MATLAB and an example of using them within a Simulink model will be sent along with the assignment. You may use these functions, but it is not compulsory. Also if you think that your ideas cannot be realized with these functions, feel free to implement you own methods.

Problem 3. Black-box data-driven modeling

Note: To receive the files for this problem, send an e-mail to Sachin Navalkar (S.T.Navalkar@tudelft.nl). Please, mention the names and initials of the two members of your groups, the student numbers and an e-mail address at which you wish to receive the files.

Given is a data set measured on an unknown **dynamic** system with one or more inputs and one output. The order of the dynamics is not larger than three. Your task is to develop a black-box model for this system, using a sigmoidal neural network. A second data set measured on the same system is provided for validating the developed model. The two data sets will be given as a MATLAB data file, containing one structure for each set. The identification data set is named `iddata` and the validation data set `valdata`. Each contains two column vectors `u`, `y` of identical lengths holding the measured input and output, and the sampling time `Ts` used in collecting the data (e.g., `iddata.u`, `iddata.y`, `iddata.Ts`).

Report the one-step-ahead and the simulation root-mean-squared errors for both sets and show a representative plot for the fit on the training and the validation data sets. Discuss the results, including the quality of the model fit on the two data sets. Refer to Lecture 3 (sheet 12) for the difference between one-step-ahead prediction and simulation.

It is recommended that you use the Neural Network toolbox of MATLAB to solve this problem. The `newff`, `train` and `sim` functions respectively create, train and simulate a feed-forward neural network. Use `help nnet`, `help newff`, etc. to get started with the Neural Network toolbox.

Matlab programming

Strive for a compact and elegant MATLAB code, use functions where suitable, avoid loops (`for`, `while`, etc.) and also `if-then` constructs at places where you can easily use vector and matrix operations. Search for "vectorization" in Matlab help for helpful tips on the proper MATLAB programming style.

If you are unfamiliar with programming in Matlab, here are some pointers that should help you to quickly learn the basics. To access the Matlab documentation, type `doc` at the command line. A good place to start is the "Getting Started" node of the Matlab documentation. Focus especially on "Matrices and Arrays" and "Programming". A minimal knowledge of "Graphics" is required in order to present your results in a graphical form. For a more in-depth introduction, see "Mathematics" > "Matrices and Linear Algebra", and under this node: "Matrices in Matlab" and "Solving Linear Systems of Equations".[2]

You will also find useful the MATLAB exercises on the Download section of the course website:

`http://www.dcsc.tudelft.nl/~sc4081/transp/matlex.pdf`

and the corresponding solutions:

`http://www.dcsc.tudelft.nl/~sc4081/transp/matlsol.pdf`

[2]These pointers assume the documentation structure in Matlab 7.3. While the the structure may vary in other versions, you should still be able to easily find these topics.