KON 426E

Intelligent Control Systems

Assignment 2

Lecturer: Prof. Dr. Gülay Öke Günel
TA: Res. Asst. Murat Furkan Mansur

Prepare a detailed, illustrative report for your homework. Make sure your report is well formatted and any figures, equations, tables are aligned and has appropriate numbering. Report formatting will be considered in scoring your homework. Submit your codes and report in a single zip file.

You are free to use any LLM for help, but please do not blindly copy-paste back-and-forth between an LLM and your report/IDE.

For any questions regarding the homework, you can e-mail me at mansur17@itu.edu.tr. Good luck!

Question 1: Blackbox System Identification & Control

In this question, you are given two unknown systems. These systems are black-box models given to you as compiled code (bytecode for Python, and P-Code for MATLAB). For each system you are given both MATLAB and Python versions and you are free to choose whichever you want. But make sure to state which one you used.

For each system, please analyze and experiment with their input-output characteristics and try to understand them. Explain your thought process and show useful graphs to explain how you work.

If you are using Python, you can use any deep learning framework you like such as Pytorch, Tensorflow, etc. To be able to use Python codes, you need Python 3.8. You can use conda (anaconda or miniconda) to prepare the required environment. See the appendix for conda environment creation. You can prepare regular python scripts (.py file) or Jupyter notebooks (.ipynb) file for your implementations.

If you are using MATLAB, you can use any Toolbox such as Neural Network Toolbox, Deep Learning Toolbox, etc. You can prepare live scripts or regular scripts for your implementations.

Part A: System 1

The system for this part is named System1. You are given a sample code on how to use the related code to experiment with the system.

System1 is given as a class to you, and the class definition is given in the box below.

System1 Class Overview

System1 represents an unknown dynamic system. It provides three public methods for interacting with the system using standard input signals.

Constructor:

When you construct the **System1** class, use **last two digit (last digit if it is 0x) of your student ID**, for example:

- system = System1(42)

Methods:

- `update(input)`
  Computes the system output for a given input.
  - **Input:** A scalar or vector of numerical values.
  - **Output:** System output(s) for that input.
- `step(t_vec)`
  Simulates the response to a step input.
  - **Input (optional):** A time vector. Uses default if not provided.
  - **Output:** Time vector and corresponding output.
- `ramp(t_vec)`
  Simulates the response to a ramp input.
  - **Input (optional):** A time vector. Uses default if not provided.
  - **Output:** Time vector and corresponding output.

An example usage of System1 is given in:

- **python/part_a_example.py**: If you prefer python.
- **matlab/part_a_example.m**: If you prefer MATLAB.

System Identification

a.1): Experiment with the system and find out the system characteristics. Don't forget to construct the class with your student ID as described in the overview box!

Answer the followings:

- What type of system is this?
- What are the input-output characteristics? Obtain the parameters of the system, and express the input output equations of the system.
- Explain step by step how did you approach to identifying the system and provide useful graphs and plots.

An example answer should at least be like:

"This system is ..., we can clearly see from the step/ramp/other input output response as shown in Figure x. The parameters are identified as ..., the input output equations are ..."

Controller Design

a.2): Design a controller for this system that accomplishes good reference tracking. You can use step input as input to the system. Explain why did you use such a control mechanism. Provide resulting plots.
(**Hint:** An ideal controller is the inverse of the dynamics of the plant)

Part B: System 2

The system for this part is named System2. You are given a sample code on how to use the related code to experiment with the system. This system is very familiar to any control engineer.

System2 is given as a class to you, and the class definition is given in the box below.

System2 Class Overview

System2 models a dynamic system with internal state. It provides methods to simulate the system's response to different input types.

Constructor:

When you construct the **System2** class, use **last two digit (last digit if it is 0x) of your student ID**, for example:

- system = System2(42)

Methods:

- `output(input)`
  Computes the system output in response to a given scalar input.
  - **Input:** A scalar input value.
  - **Output:** The system's output, including internal noise.
- `reset()`
  Resets the internal state of the system.
- `step(t_final)`
  Simulates the response to a unit step input.
  - **Input (optional):** Final time of simulation. Default is 5 seconds.
  - **Output:** Time vector and system output over time.
- `ramp(t_final)`
  Simulates the response to a ramp input.
  - **Input (optional):** Final time of simulation. Default is 5 seconds.
  - **Output:** Time vector and system output over time.

Note: This system evolves over time and has some internal dynamics, so **don't forget to reset the internal states whenever you need to start from initial conditions**.

An example usage of System2 is given in:

- **python/part_b_example.py**: If you prefer python.
- **matlab/part_b_example.m**: If you prefer MATLAB.

System Identification

b.1): Experiment with the system and find out the system characteristics. Don't forget to construct the class with your student ID as described in the overview box!

Answer the followings:

- What type of system is this?
- What are the input-output characteristics? Obtain the parameters of the system, and express the input-output equations of the system.
- Explain step by step how you approached to identifying the system and provide useful graphs and plots.

An example answer should at least be like:

"This system is ..., we can clearly see from the step/ramp/other input output response as shown in Figure x. The parameters are identified as ..., the input output equations are ..."

b.2): Design and train a neural network to fit the system. Use the following items as your guideline when you explain your approach.

- **Architecture:** Explain the architecture, input(s) and output(s) to the Neural Network.
- **Dataset:** Provide information about how you obtained the dataset and how many data points in the dataset you created.
- **Training:** Choose appropriate loss function and hyperparameters, explain your preference.
- **Validation:** Provide useful metrics for how well the neural network fits to the plant, and show useful graphs.

Controller Design

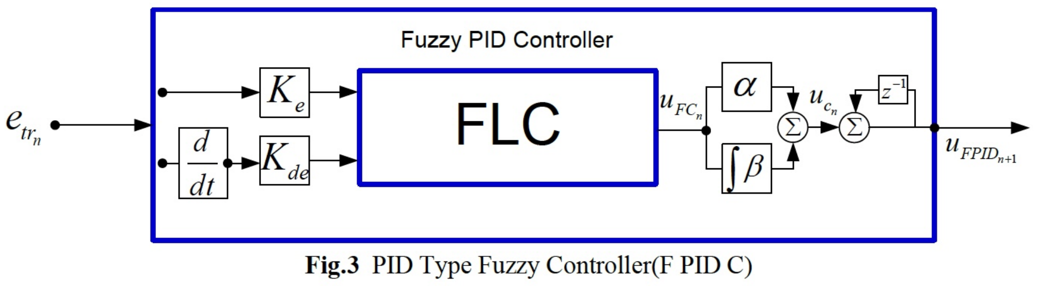
b.3): Design an intelligent/adaptive controller for the plant. You can use any architecture you learned in this course. Check the course slides, hands-on session slides for different architectures. Explain why you chose that architecture and explain your implementation. Show how well your intelligent controller controls the system.

b.4) [Optional]: For extra credits, design a conventional controller for the system. You can use any controller type you learned in Control System Design course. Compare the performance with your intelligent controller.

Question 2: PID Type Fuzzy Controller

In this question, a PID-type fuzzy controller is used to control a Continuous Stirring Tank Reactor (CSTR) system. The plant dynamics are simulated using Runge-Kutta 4 (RK4) method and given in CSTR.m file.

The structure of the fuzzy controller is shown in Figure 1. The controller uses triangular membership functions with cores $\{-1, -0.4, 0, 0.4, 1\}$ for each linguistic variable as shown in the Figure 2. The center of gravity method has been utilized as the defuzzification method.



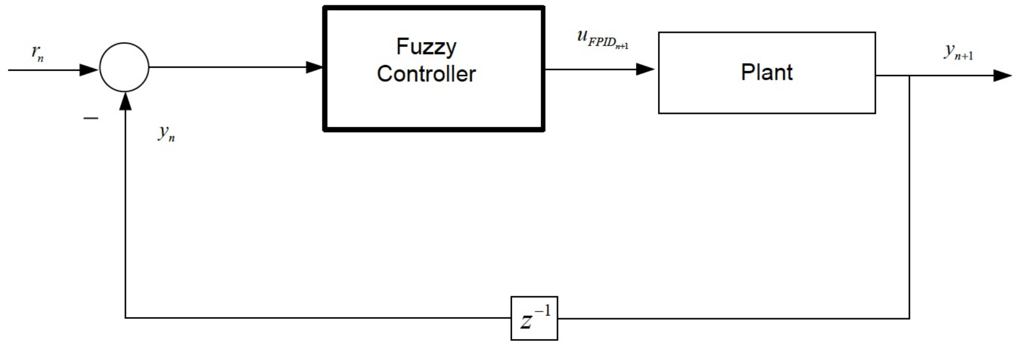


Figure 1: Structure of the PID-Type Fuzzy Controller

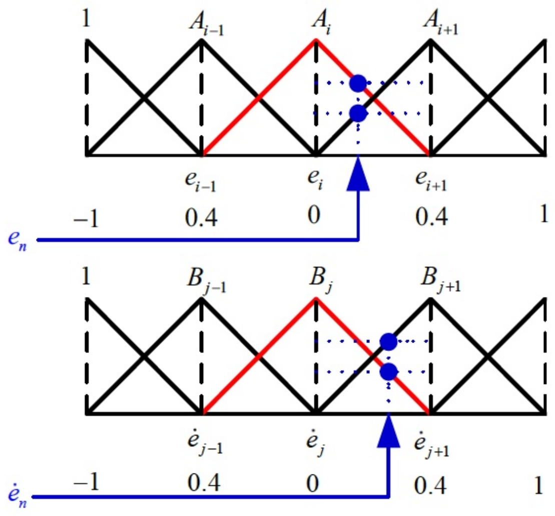


Figure 2: Membership Functions of A_i and B_j

The inputs of the fuzzy controller are defined as:

$$e_n = K_e \cdot e_{tr_n}, \qquad \dot{e}_n = K_{de} \cdot \dot{e}_{tr_n} \tag{1}$$

The PID-type fuzzy controller produces a control signal as follows:

$$u_{FC_n} = f(K_e, K_{de}, e_n, \dot{e}_n) \tag{2}$$
$$u_{c_n} = \alpha_n u_{FC_n} + \beta_n (u_{FC_n} + u_{FC_{n-1}}) \tag{3}$$
$$u_{FPID_{n+1}} = u_{FPID_n} + u_{c_n} \tag{4}$$

where K_e and K_{de} are the input scaling coefficients, and α_n, β_n are the output coefficients of the derivative and integral parts of the PID-type fuzzy controller, respectively. These are the hyperparameters of the Fuzzy PID controller. The values are given in the Table 3.

The control signal produced by the FLC which is given in the Equation 2 above is calculated as shown in Equation 5 below:

$$\begin{aligned} u_{FC_n} = \; & A_i(e_n)B_j(\dot{e}_n)u_{ij} + A_{i+1}(e_n)B_j(\dot{e}_n)u_{i+1j} \\ & + A_i(e_n)B_{j+1}(\dot{e}_n)u_{ij+1} + A_{i+1}(e_n)B_{j+1}(\dot{e}_n)u_{i+1j+1} \end{aligned} \tag{5}$$

Here, the membership functions A_i and B_j are defined using triangular functions:

$$A_i(e_n) = \frac{e_{i+1} - e_n}{e_{i+1} - e_i}, \qquad A_{i+1}(e_n) = \frac{e_n - e_i}{e_{i+1} - e_i} \tag{6}$$

$$B_j(\dot{e}_n) = \frac{\dot{e}_{j+1} - \dot{e}_n}{\dot{e}_{j+1} - \dot{e}_j}, \qquad B_{j+1}(\dot{e}_n) = \frac{\dot{e}_n - \dot{e}_j}{\dot{e}_{j+1} - \dot{e}_j} \tag{7}$$

The values you need to use for e_i and $\dot{e}_i$ are given in the Table 1, and the error values e_n and $\dot{e}_n$ are the scaled error values at time step n.

	$\dot{e}_{-2}$	$\dot{e}_{-1}$	$\dot{e}_0$	$\dot{e}_1$	$\dot{e}_2$
e_{-2}	-1	-0.7	-0.5	-0.3	0
e_{-1}	-0.7	-0.4	-0.2	0	0.3
e_0	-0.5	-0.2	0	0.2	0.5
e_1	-0.3	0	0.2	0.4	0.7
e_2	0	0.3	0.5	0.7	1

Table 1: Fuzzy Rule Table

a) Step-by-step Control Procedure

Explain the fuzzy control procedure step-by-step. For example:

- Step 1: Calculate the tracking error
- Step 2: Obtain the inputs to the controller
- ...
- Step k: Calculate current control signal
- Step k+1: Apply control signal to the plant

b) MATLAB Implementation

Write a MATLAB script that implements the given Fuzzy PID Controller, and control the CSTR plant. Construct and use the reference signal shown in Figure 3 as reference input to the controller. The parameters and initial values for simulating the CSTR plant is given in Table 2. The hyperparameters for the Fuzzy PID controller are given in Table 3.

- Plot:
  1. System output versus time
  2. Control signal versus time

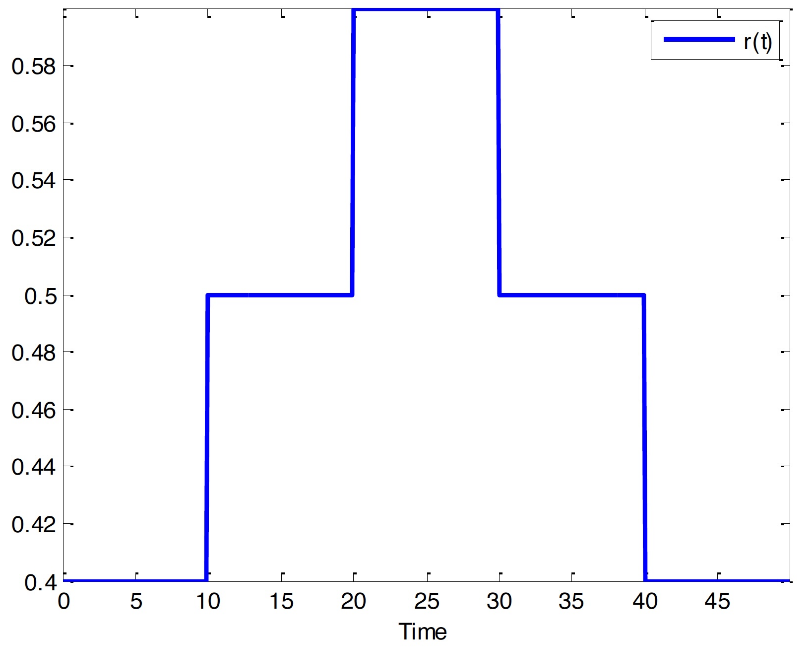


Figure 3: Reference Signal for Controller Evaluation

Symbol	**Value**
$Da1$	3
$Da2$	0.5
$Da3$	1
$d2$	1
h (sampling time)	0.1
x_1	Initial Value = 0.31
x_2	Initial Value = 0.71
x_3 (output)	Initial Value = 0.5

Table 2: CSTR Parameters and Initial Values

K_e	K_{de}	α	β
0.15	1.5	0.7	1.2

Table 3: Fuzzy PID Controller Parameters

c) Adaptive Mechanism Proposal

Propose and explain an adaptation mechanism to improve the controller performance over time. This mechanism may be based on:

- Neural networks
- Fuzzy logic adaptation
- Reinforcement learning
- Any other intelligent method

Appendix

Preparing Python Environment

To be able to use the given python codes, you need python 3.8. Using anaconda or miniconda, you can create a virtual environment. First, install anaconda or miniconda, then follow the box below.

Python Environment Setup Instructions

To ensure compatibility with the system modules, you must use **Python 3.8**. Follow these steps:

1. Create a Conda environment with Python 3.8:

```
conda create -n py38 python=3.8
```

2. Activate the environment:

```
conda activate py38
```

3. Install required Python packages:

```
pip install numpy matplotlib torch
```

Optional: You may also install other tools like `jupyter` if needed. Once set up, this environment will be compatible to use with the given python codes.