**EE449**

**Homework 3 – Fuzzy Control**

*Due: 23:55, 12/06/2022*

Berkay İPEK

2304814

Sec1

Table of Contents

[1. Vaccination v1 3](#_Toc105946558)

[1.1 Set Partitioning 3](#_Toc105946559)

[1.2 Fuzzy Control Rules 5](#_Toc105946560)

[1.3 Fuzzification and Defuzzification Interface 6](#_Toc105946561)

[1.4 Simulation 7](#_Toc105946562)

[2. Vaccination v2 8](#_Toc105946563)

[2.1 Set Partitioning 8](#_Toc105946564)

[2.2 Fuzzy Control Rules 11](#_Toc105946565)

[2.3 Implementation and Simulation 12](#_Toc105946566)

[2.4 Comparison 14](#_Toc105946567)

[3. Appendix 15](#_Toc105946568)

[3.1 Vaccination v1 Code 15](#_Toc105946569)

[3.2 Vaccination v2 Code 17](#_Toc105946570)

# Vaccination v1

## Set Partitioning

* *Partition the sets where the measurement and the output lie into 3 fuzzy sets. Namely, you are to represent [0, 1] and [−0.2, 0.2] with three fuzzy sets for each. You may experiment on different partitioning strategies by examining the performances. Decide your favorite partitioning and plot your fuzzy partitions for both sets in Python and include them in your report. Explain how you decide that partitioning.*

*Chart, line chart

Description automatically generated*

**Figure 1:** Vaccination Rate Partitioning

The center of medium should be 0.6, which is desired output. The control should give 0 when only vaccination rate reaches 60%. For medium and high rates, I changed initial parameters to see their effects. As high rate starting point to increase decreases (or low rate starting point to decrease increases), it is converging in a faster way. The reason is that the system labels a vaccination rate as medium in earlier stage. Therefore, it is output control will be smaller. If we want to get flawless result at the point 0.6, we can change these points (0.4 and 0.8) to wider points (0.3 and 0.9). However, I chose to converge them as soon as possible. Therefore, I gave their parameters as like that.

*Chart, line chart

Description automatically generated*

**Figure 2:** Control Rate Partitioning

The center of medium should be 0.0, which is desired output. The control should give 0 when only vaccination rate reaches 60%. For medium and high rates, I changed initial parameters to see their effects. The results were similar to previous one. As the medium one range increases, convergence time increases also since system response is much larger. Therefore, I gave these parameters like that.

For both input sets, I used trimf and trampf methods in the library of skfuzzy. (See the code in the appendix)

## Fuzzy Control Rules

*Considering your partitions, list your control rules.*

Text

Description automatically generated

**Figure 3:** Control Rules

## Fuzzification and Defuzzification Interface

Text

Description automatically generated

**Figure 4:** Fuzzification and Defuzzification Interface

Firstly, I defined control rules as explained in previous part. Then, assigned it to a member of a class, called as controlRules(it is a list of rule objects in skfuzzy library). Then, I imported control class from skfuzzy library. It has a method called as ControlSystemSimulation with an argument of rules. That’s how my fuzzification works.

For defuzzification, I used compute method for control system simulation variable in the skfuzzy library. Then, I got output by ‘control\_rates’ key of the dictionary. As a result, I gave this control output to system.

## Simulation

Chart, line chart

Description automatically generated

**Figure 5:** Simulation Result of Vaccination-1

# Vaccination v2

## Set Partitioning

* *Partition the sets where the measurements lie into 3 fuzzy sets. Namely, you are to represent [0, 1] and [−1, 1] with three fuzzy sets for each. Partition the output set into 5 fuzzy sets [−0.2, 0.2]. You may experiment on different partitioning strategies by examining the performances. Decide your favorite partitioning and plot your fuzzy partitions for both sets in Python and include them in your report. Explain how you decide that partitioning.*

*Chart, line chart

Description automatically generated*

**Figure 6:** Vaccination Rate Partitioning

The center of medium should be 0.6, which is desired output. The control should give 0 when only vaccination rate reaches 60%. For medium and high rates, I changed initial parameters to see their effects. As high rate starting point to increase decreases (or low rate starting point to decrease increases), it is converging in a faster way. The reason is that the system labels a vaccination rate as medium in earlier stage. Therefore, it is output control will be smaller. If we want to get flawless result at the point 0.6, we can change these points (0.4 and 0.8) to wider points (0.3 and 0.9). However, I chose to converge them as soon as possible. Therefore, I gave their parameters as like that.

*Chart, line chart

Description automatically generated*

**Figure 7:** Control Rate Partitioning

The center of medium should be 0.0, which is desired output. The control should give 0 when only vaccination rate reaches 60%. For medium and high rates, I changed initial parameters to see their effects. The results were similar to previous one. As the medium one range increases, convergence time increases also since system response is much larger. Therefore, I gave these parameters like that.

*Chart, line chart

Description automatically generated*

**Figure 8:** Failure Rate Partitioning

The center of medium should be 0.0, which is desired output. The control should give 0 when only vaccination rate reaches 60%. For medium and high rates, I changed initial parameters to see their effects. The results were similar to previous one. As the medium one range increases, convergence time (settle time) increases also since system response is much larger. However, as it increases, overshoot problem decreases. Moreover, at some point it starts to be underdamped situation rather than overshooting situation.

For all input sets, I used trimf and trampf methods in the library of skfuzzy. (See the code in the appendix).

## Fuzzy Control Rules

* *Considering your partitions, list your control rules. You may experiment on different set of rules by examining the performances. Decide your favorite set of rules and list them. Explain how you decide that rule set.*

A screenshot of a computer

Description automatically generated with medium confidence

**Figure 9:** Control Rules

We know that if vaccination rate and failure rate are fully mid, then we got 60% vaccination rate. Also, if we get vaccination rate of high and failure rate of low, we can achieve this rate. For example, if we have vaccination rate of 61% and failure rate of %1 then we will achieve real vaccination rate of %60. Also, we can achieve it with having vaccination rate of low and failure rate of high. Therefore, control output should be middle when these situations occur (Rule 2).

Rules 0 and 4 state that, when both rates have same partitioning, control should be very low or high. It is because as vaccination & failure rates are high (or low), control should be very low (or high) to decrease (increase) vaccination rate.

Rules 1 and 3 state that when one of them is middle, which is desired situation, other one is low or high, then we should give high or low to get balanced.

## Implementation and Simulation

Text

Description automatically generated

**Figure 10:** Fuzzification and Defuzzification Interface

It is same with the previous part expect for input size. Failure rate increases,

Firstly, I defined control rules as explained in previous part. Then, assigned it to a member of a class, called as controlRules(it is a list of rule objects in skfuzzy library). Then, I imported control class from skfuzzy library. It has a method called as ControlSystemSimulation with an argument of rules. That’s how my fuzzification works.

For defuzzification, I used compute method for control system simulation variable in the skfuzzy library. Then, I got output by ‘control\_rates’ key of the dictionary. As a result, I gave this control output to system.

Chart

Description automatically generated

**Figure 11:** Simulation Result of Vaccination-2

## Comparison

Main difference is overshooting problems. For vacc-1 version, convergence time is very long, and cost is quite high since the system has not enough input parameters. However, for vacc-2 version, convergence time is shorter, and cost is lower than vacc-1 version. As system gets proper inputs, results are getting better as this experiment show. Therefore, when we are going to simulate, firstly problem should be described in detail, so vacc-2 version is better vacc-1 version.

Vacc-1 version have an advantage of being simple and calculation time. When system needs more iteration and has negligible failure rate, rather than having vacc-2 can be chosen since it is easy to build and faster to compute.

## 3. Appendix

## Vaccination v1 Code

Text

Description automatically generated with medium confidence

Graphical user interface, text, application

Description automatically generated

## Text Description automatically generatedVaccination v2 Code

Text

Description automatically generated with low confidence