吴贤铭智能工程学院
SHIEN-MING WU SCHOOL OF INTELLIGENT ENGINEERING

SharpShooters

Controller Design for a Gas Separator Plant

082100071, Modern Control Theory

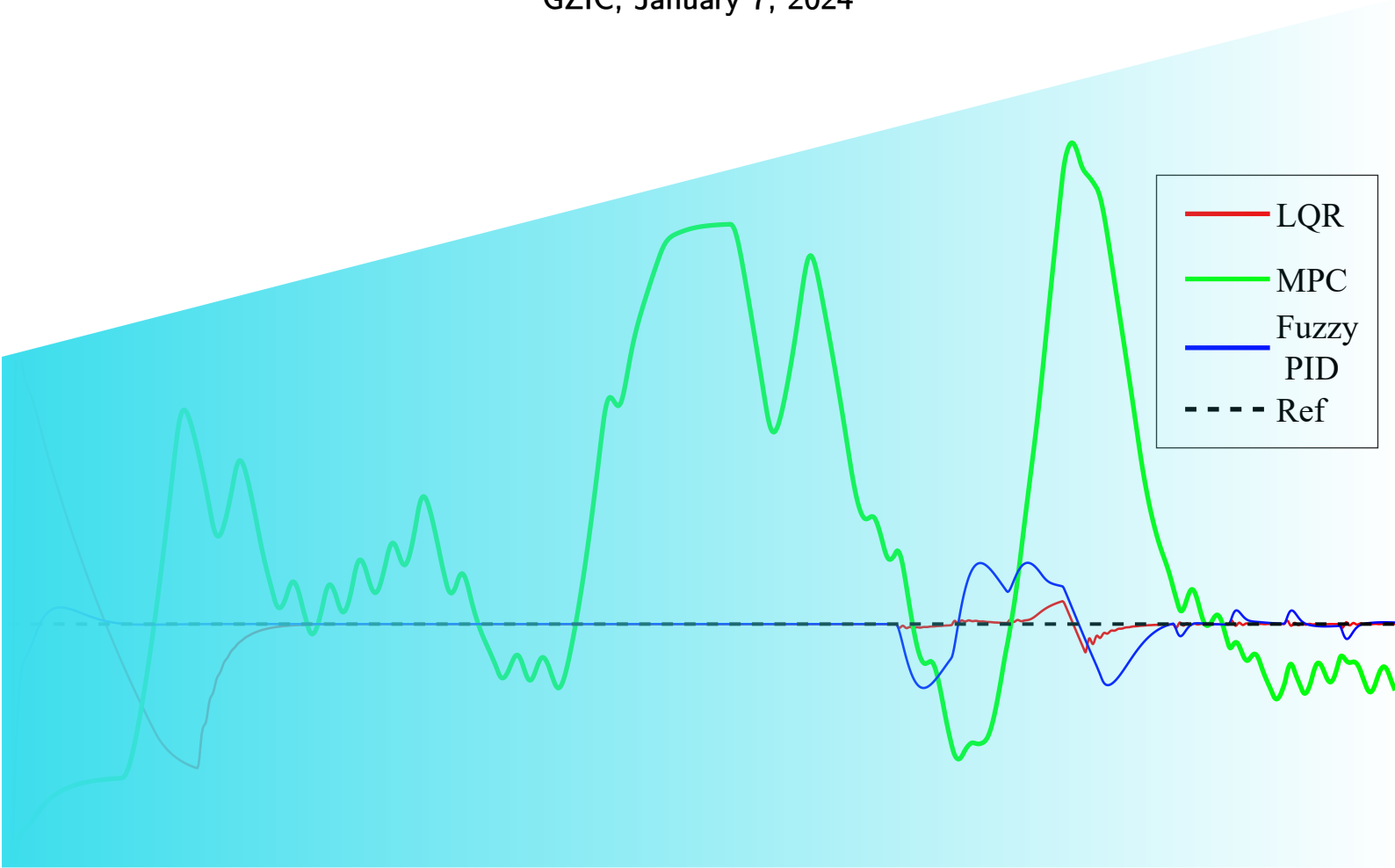
Group SharpShooters

Full Name	Student ID
Xinlei Zhang	202030101256
Ze'an He	202030020342
Yile Shen	202030020205
Yuli Yang	202030322149
Jinan Guo	201930031052

Github Page

Tutor: Dr.Gang Chen

GZIC, January 7, 2024



Contents

1 | Task Introduction

1.1 | Task Background

This project aims to use the knowledge acquired from the senior undergraduate course, Modern Control Theory, to address the real-world gas separation problem and improve the plant process response. In reality, gas separation problems commonly exist in the scenario where a particular gas is desired to be extracted from a mixed gas consisting of more than 2 components. This project takes the oxygen enrichment as the background, where the oxygen is the desired gas to be separated from a mixed gas. In a real-world plant, permeable membrane is usually employed to conduct gas separation, since the permeability through the membrane differs in different gases, leading to different penetration rate under the same separator condition. Moreover, the penetration rate of gases through the membrane can be controlled by crafting the pressure difference between the two sides of the membrane. Hence, with the permeable membrane and crafted pressure difference, oxygen enrichment can be realized and the response during this enrichment process can be further controlled by adjusting the pressure difference during the process.

1.2 | Task Details Description

As shown in Fig. 1.1, in our problem, there existed three cascaded selectively permeable membrane subsystems composing the whole gas separator system. In each subsystem, the inlet gas is imported from the material part, penetrates the selectively permeable membrane to the permeation part and is exported to the next subsystem. The initial mixed gas is imported from the left subsystem and is exported from the right subsystem after three times extraction, or called oxygen enrichment. Agitator in each subsystem makes the gas concentration uniform by agitating inside. Pressure sensors are placed in both the material and permeation parts in each subsystem to measure the real-time pressure value, to provide the feedback information for closed-loop pressure control.Moreover, density sensors are placed in the permeation parts of three subsystems to measure the real-time oxygen density. With information from those sensors, the

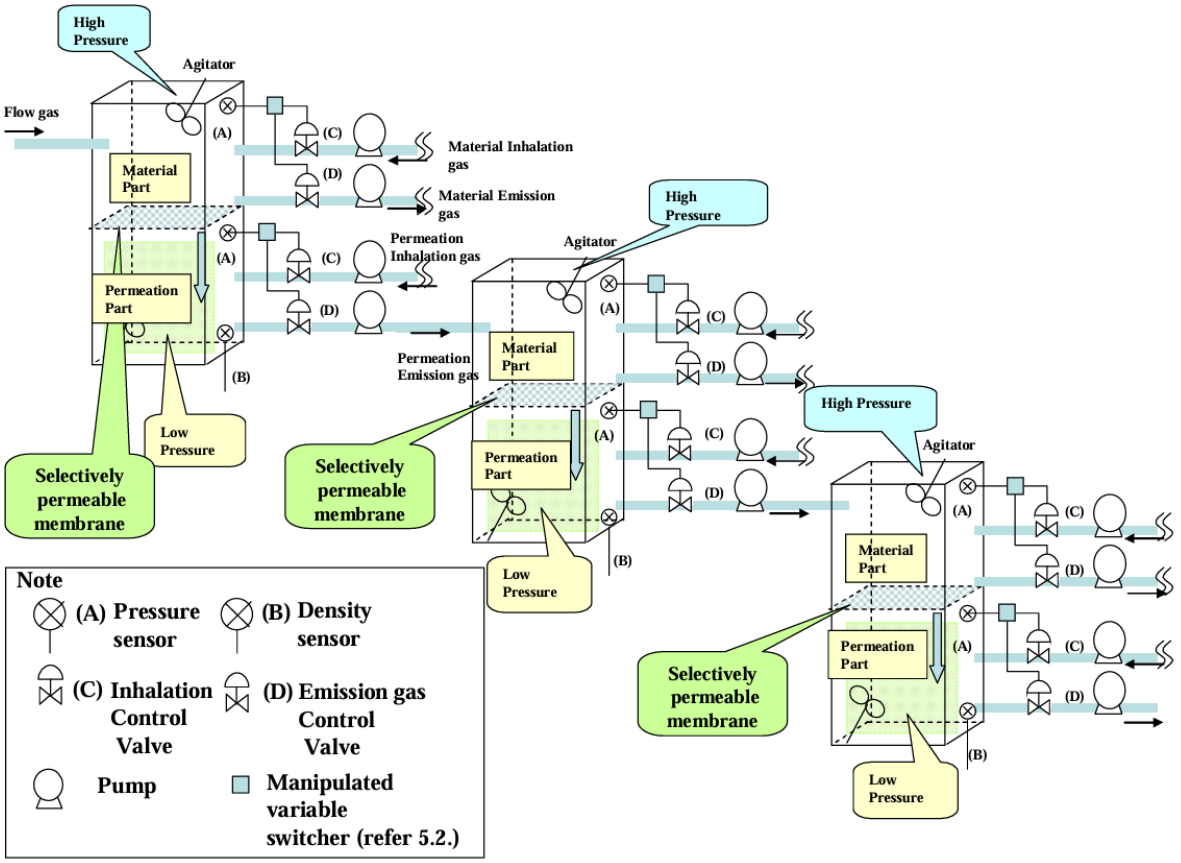


Figure 1.1: Illustration for the gas separator plant[1]

pressure difference could be manipulated with pumps to control the oxygen density in each permeation part, so as to realize oxygen enrichment, to harvest the exported gas with the desired oxygen density.

1.3 | Our Works

In the MATLAB official document[1], the procedures to address this problem, including mathematical modeling, system identification, controller design and results evaluation are elaborated and discussed in detail. We follow this reference material and realize the results given in the material. Furthermore, we propose our methods to address this problem. Our works in this project can be summarized as follows:

1. Acquire the knowledge and methods given in the reference material and implement the whole procedure;
2. Propose other methods to address the problem based on the knowledge acquired from the class, specifically from the aspects of system modeling and controller design;
3. Record all the implementation steps in detail and present those steps in this report, in order to make this report as a detailed implementation manual in addition to the reference material. Our Simulink model and MATLAB script are attached in the appendix, and also can be downloaded on the website.

1.4 | Project Assignment

Plant Modeling: Xinlei Zhang, Yile Shen, Jinan Guo, Ze'an He
Control Law Design: Jinan Guo, Xinlei Zhang, Yuli Yang
Project Report: Xinlei Zhang, Jinan Guo, Yile Shen, Ze'an He, Yuli Yang

1.5 | Report Organization

This report is organized as follows. We start in Section 2 to introduce the modeling procedure for the plant based on physical laws, and discuss both state-space modeling and system identification methods; In Section 3, we present the details to implement system identification based on the MATLAB toolboxes used in the reference material; In Section 4, we elaborate on the proposed methods to design the controller to address the problem; The results from our methods are evaluated and compared with those from the reference material in Section 5; At last, the project is concluded in Section 6 and the implementable code is given in Appendices.

1.6 | Nomenclature

Variable	Explanation
F	Material gas quantity flown into the material part per unit time
x_{AF}	Mol fraction of component A in F
N_{tWi}	Number of mol in all gases in the material part in subsystem i
N_{AWi}	Number of mol in gas A in the material part in subsystem i
N_{tQi}	Number of mol in all gases in the permeation part in subsystem i
N_{AQi}	Number of mol in gas A in the permeation part in subsystem i
$\bar{P}_{tWi}$	Total nominal pressure of gas in the material part in subsystem i
$\bar{P}_{tQi}$	Total nominal pressure of gas in the permeation part in subsystem i
y_{AQi}	Mol fraction of component A in the permeation part in subsystem i
W_i	Emission quantity of gas i from the material part per unit time
Q_i	Emission quantity of gas i from the permeation part per unit time
Parameter	**Explanation**
$Area$	Membrane area
$C_{A,B}$	Transfer coefficient of gas A or B
k	Gas constant 8.314 times temperature divided by volume

Table 1.1: Variables and parameters explanation

2 | Plant Modeling

2.1 | Modeling of the Gas Separator

The physical mechanism in this section is mainly summarized from the reference material[1]. Please refer to that material for mroe details. Moreover, state-space representation of the system is proposed here to develop more insights into the system dynamics. All symbols have been defined in the Table 1.1.

2.1.1 | Separation Mechanism

Permeable membrane is barrier diffusion that selectively permeates according to material type. Concentration difference, pressure difference and difference in potential are used to drive mass transfer.

2.1.2 | Mathematical Modeling for the membrane separation

Take the permeation process of gas A as an example. The gas quantity penetrates the membrane from the material side to the permeation side in the subsystem i, denoted by N_{Ai}, can be calculated as,

$$N_{Ai} = C_A k(N_{AWi} - N_{AQi}) \tag{2.1}$$

2.1.3 | Mathematical Modeling for the separator plant

Since the three subsystems are cascaded together and inherently similar, here we only present the modeling result of the first subsystem, and the other results can be easily obtained by replacing the input gas variable with the output gas variable in the last subsystem.

$$\dot{N_{tWi}} = F - W_i - Area \cdot (N_{Ai} - N_{Bi}) \tag{2.2}$$

$$\dot{N_{AWi}} = F \cdot x_{AF} - W_i \cdot \frac{N_{AWi}}{N_{tWi}} - N_{Ai} \cdot Area \tag{2.3}$$

$$\dot{N_{tQi}} = (N_{Ai} + N_{Bi}) \cdot Area - Q_i \tag{2.4}$$

$$\dot{N_{AQi}} = N_{Ai} \cdot Area - Q_i \cdot \frac{N_{AQi}}{N_{tQi}} \tag{2.5}$$

$$\bar{P}_{tWi} = \frac{N_{tWi} \cdot k}{101300} \tag{2.6}$$

$$\bar{P}_{tQi} = \frac{N_{tQi} \cdot k}{101300} \tag{2.7}$$

$$y_{AQi} = \frac{N_{AQi}}{N_{tQi}} \tag{2.8}$$

2.2 | State-Space Modeling

From the above mathematical modeling for the separator plant, it's obvious that there exists the dynamics regarding the variables, N_{tWi}, N_{AWi}, N_{tQi} and N_{AQi}. Hence, we continue to take the first subsystem as the example to formulate its state-space model.
Select the state variable $\mathbf{x}$, control input $\mathbf{u}$ and the output $\mathbf{y}$ as

$$\mathbf{x} = [N_{tWi}, N_{AWi}, N_{tQi}, N_{AQi}]^{\mathrm{T}} \tag{2.9}$$

$$\mathbf{y} = [\bar{P}_{tWi}, \bar{P}_{tQi}, y_{AQi}]^{\mathrm{T}} \tag{2.10}$$

$$\mathbf{u} = [F, x_{AF}, W_i, Q_i]^{\mathrm{T}} \tag{2.11}$$

The system process function $f(\cdot)$ can be given by

$$\dot{N_{tWi}} = f_1(\mathbf{x}, \mathbf{u}) = -AkC_B(N_{tWi} + N_{tQi}) + Ak(C_B - C_A)(N_{AWi} + N_{AQi}) + F - W_i \tag{2.12}$$

$$\dot{N_{AWi}} = f_2(\mathbf{x}, \mathbf{u}) = Fx_{AF} - W_i \frac{N_{AWi}}{N_{tWi}} - AC_A k(N_{AWi} - N_{AQi}) \tag{2.13}$$

$$\dot{N_{tQi}} = f_3(\mathbf{x}, \mathbf{u}) = -Q_i + AC_B k(N_{tWi} - N_{tQi}) + Ak(C_A - C_B)(N_{AWi} + N_{AQi}) \tag{2.14}$$

$$\dot{N_{AQi}} = f_4(\mathbf{x}, \mathbf{u}) = -Q_i \frac{N_{AQi}}{N_{tQi}} + AC_A k(N_{AWi} - N_{AQi}) \tag{2.15}$$

where the parameter $Area$ is abbreviated as A.

The system output function $h(\cdot)$ can be given by

$$\bar{P}_{tWi} = h_1(\mathbf{x}) = \frac{N_{tWi} \cdot k}{101300} \tag{2.16}$$

$$\bar{P}_{tQi} = h_2(\mathbf{x}) = \frac{N_{tQi} \cdot k}{101300} \tag{2.17}$$

$$y_{AQi} = h_3(\mathbf{x}) = \frac{N_{AQi}}{N_{tQi}} \tag{2.18}$$

After obtaining the state-space representation of the system, $f(\cdot)$ and $h(\cdot)$, there are some remarks for this system:

- Multi-input Multi-output (MIMO). With total three subsystems, $i = 1, 2, 3$, we can conclude the dimensions of the state space, output space and also the input space.

$$\mathbf{x} \in \mathbb{R}^{12},\ \mathbf{y} \in \mathbb{R}^{9},\ \mathbf{u} \in \mathbb{R}^{8}$$

- Time-invariant. This system is a time-invariant system, or called autonomous system, since all the quantity except the state variable and control input are time-independent.
- Nonlinear. There exists non-linearity in the both system dynamics and output functions, including the nonlinear term, Fx_{AF}, $\frac{N_{AWi}}{N_{tWi}}$ and $\frac{N_{AQi}}{N_{tQi}}$.

3 | System Identification

3.1 | Introduction

In this section, the system identification method is provided to model the system in addition to the state-space model derived in the last section. These two modeling methods will be compared and discussed in the end of this section.
In designing control system, parametric model or non-parametric model of controlled object (plant) is necessary. Then, parametric model is acquired by employing system identification method to Simulink model. As control system design requires both outer and inner loops, system identification is performed for outer and inner loops.

3.2 | Identification of Inner Loop

The steps to implement the system identification with Simulink model in MATLAB are given as follows. Although most of the steps are following those in the reference material, we add more details and take screenshot for each step. More importantly, we use the up-to-date version of system identification toolbox in MATLAB which is more implementable with MATLAB 2023 comparing to the decade ago version used in the reference material.

1. Adding disturbances or outputs into the loop

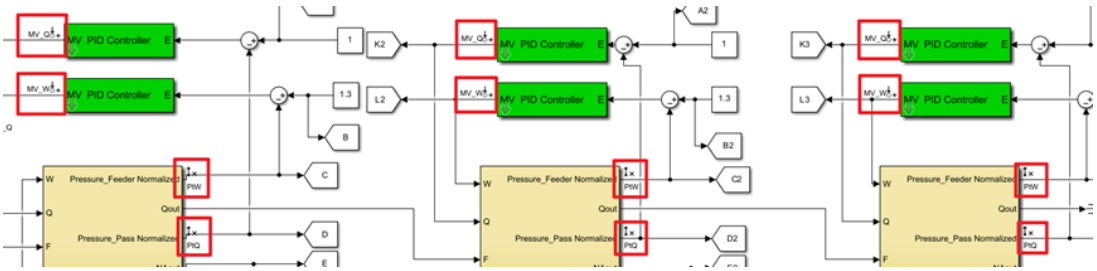


Figure 3.1: Step 1

2. Step response of permeation part in separator of subsystem 1

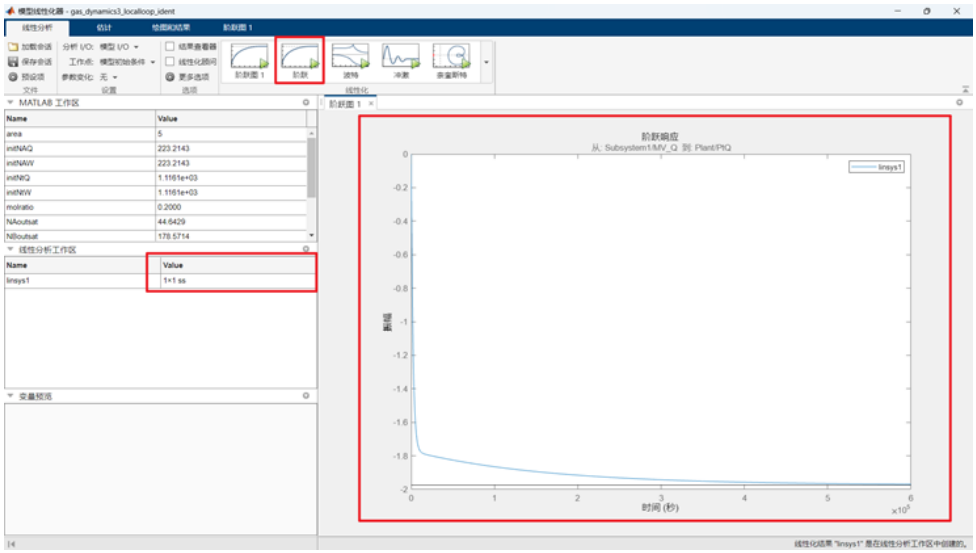


Figure 3.2: Step 2

3. Linearization result of permeation part in separator of subsystem 1

线性化结果:

从输入 "u1" 到输出 "y1":

$$\frac{-0.0008967\ s^4 - 8.989e\text{-}05\ s^3 - 1.594e\text{-}06\ s^2 - 1.966e\text{-}08\ s - 1.353e\text{-}13}{s^5 + 0.1097\ s^4 + 0.001988\ s^3 + 2.519e\text{-}05\ s^2 + 1.123e\text{-}08\ s + 6.854e\text{-}14}$$

Figure 3.3: Step 3

4. Reduce the order to simplify the transfer function

$$\frac{-0.0008967}{s} \approx -0.0009/s$$

Figure 3.4: Step 4

5. Linearization results of each inner loop

Input and output points of the inner loop	Transfer Function
Gas separator 1 material part pressure(y)—material part valve opening (u)	-0.0008/s
Gas separator 1 permeation part pressure (y)—permeation part valve opening (u)	-0.0009/s
Gas separator 2 material part pressure (y)—material part valve opening (u)	-0.0009/s
Gas separator 2 permeation part pressure (y)—permeation part valve opening (u)	-0.0008/s
Gas separator 3 material part pressure(y)—material part valve opening (u)	-0.0008/s
Gas separator 3 permeation part pressure(y)—permeation part valve opening (u)	-0.0008/s

Figure 3.5: Step 5

3.3 | Identification of Outer Loop

In this part, model is acquired by linearization and model order reduction for the outer loop system identification. The purpose is to construct inner loop which can track rapidly change of pressure set points that the outer loop outputs. Simulink Control Design is used for linearization in Simulink.

1. Build the first order delay + dead time system

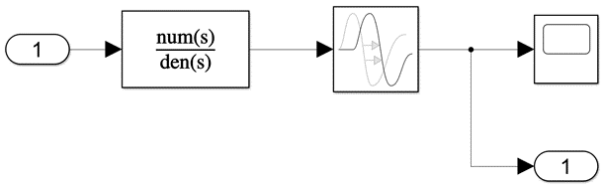


Figure 3.6: Step 1

2. Open the Parameter Estimator and choose output and input signals and parameters

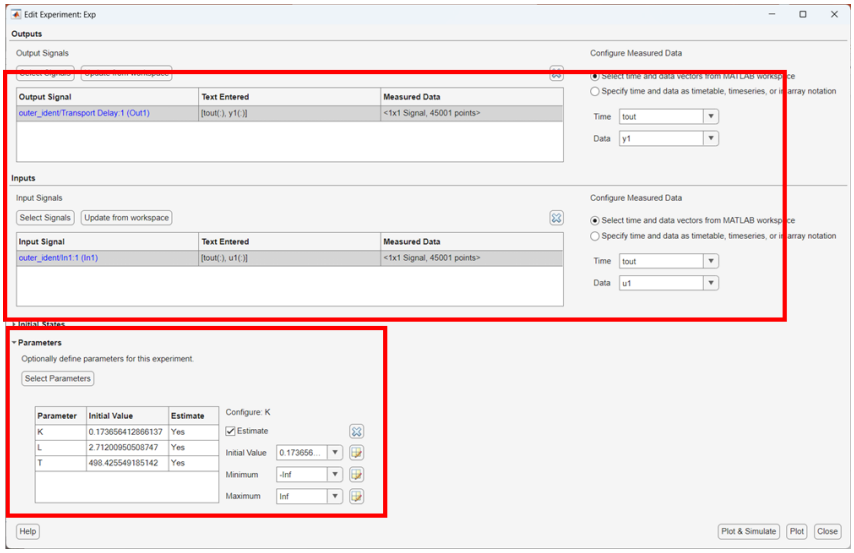


Figure 3.7: Step 2

3. Plot the output signal and the input signal

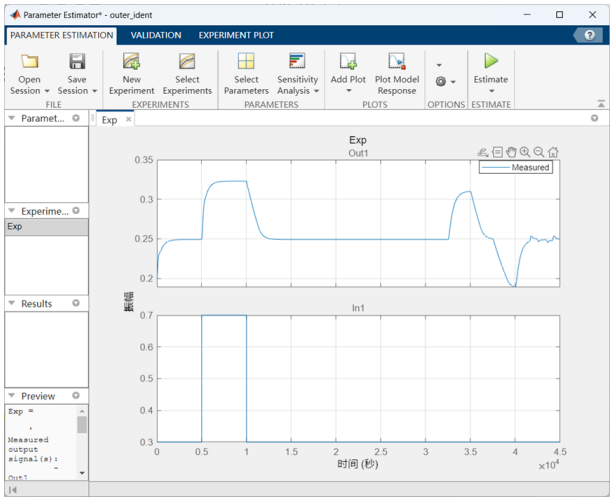


Figure 3.8: Step 3

4. Extract data of step response part, using the DATA PROCESSING toolbox

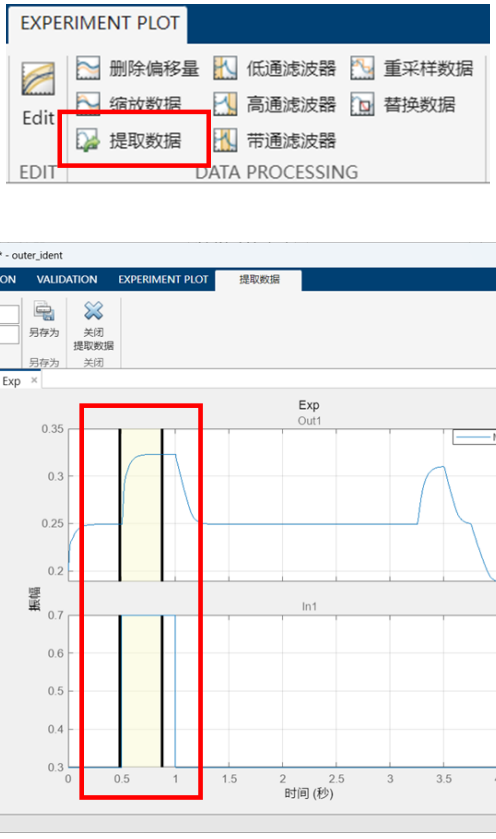


Figure 3.9: Step 4

5. Remove the offset in the data, the initial value of the output signal, using the DATA PROCESSING toolbox

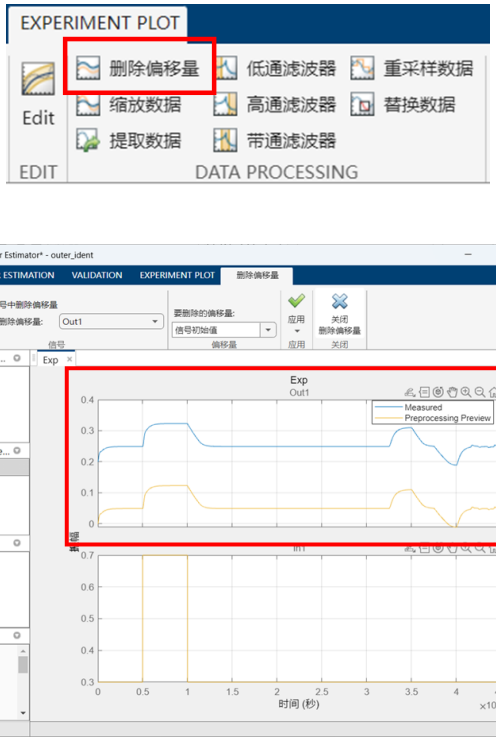


Figure 3.10: Step 5

6. The simulated value is close to the measured value

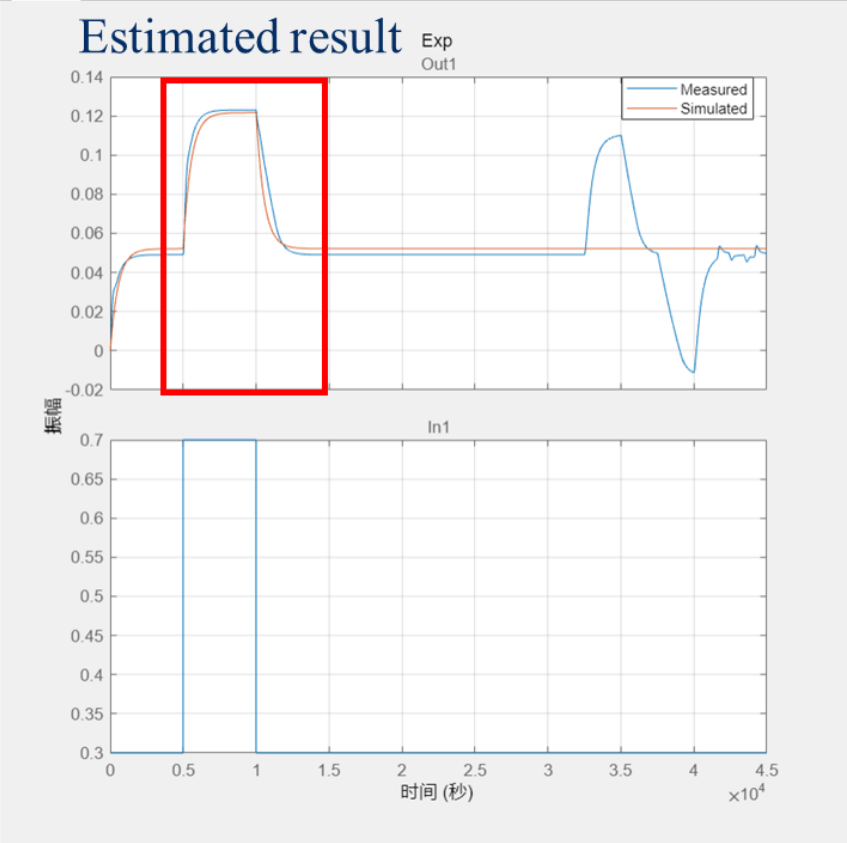


Figure 3.11: Step 6

7. Transfer function matrix of the derived model

$$\begin{bmatrix} Y1(s) \\ Y2(s) \\ Y3(s) \end{bmatrix} = \begin{bmatrix} \frac{0.182}{313.2s+1}e^{(-3.6s)} & 0 & 0 \\ \frac{0.2348}{736.7s+1} & \frac{0.2462}{205.5s+1}e^{(-43.8s)} & 0 \\ \frac{9.2372*10^{-7}}{s^2+0.0032s+3.4489*10^{-6}} & \frac{3.641*10^{-6}}{s^2+0.0071s+1.23*10^{-5}} & \frac{0.2884}{234.2s+1}e^{(-41s)} \end{bmatrix} \begin{bmatrix} U1(s) \\ U2(s) \\ U3(s) \end{bmatrix}$$

Figure 3.12: Step 7

3.4 | State-Space Modeling vs System Identification

Following the above detailed steps, we can obtain the identified system model. There are some remarks regarding this system identification method to model the system.

- The identified model is computed based on the input-output data inspection. At first, we need to build the Simulink block of the system based on the equations in Section 2. Then, imposing specific test signals, like the impulse input or the step input, the corresponding output response is collected. Through inspection into the characteristic output response, the system can be identified as a certain-order model.
- This method only produces an approximated model of the original system, since we compute the system model by comparing the original system response to that of a certain-order model. There exists modeling error which may affect the final performance of the designed controller.
- This method only produces a linearized model of the original system, as shown in Fig. 3.12. The lost non-linearity in the original model introduces extra modeling error.
- The physical meaning of each state variable is unspecified, which may bring potential risks in real-world implementation.

Comparing to the derived state-space model in the last section, the identified model is a simplified version where the linear system control design strategies could be employed due to its linearity, with loss of modeling accuracy in the non-linearity. However, the unspecified physical meaning of the state variable in the identified model may bring other potential risks and difficulties in real-world implementation, while the state variables in the state-space model have exact and clear physical meaning.

4 | Controller Design

In this section, we elaborate on the design details of fuzzy PID controller and linear quadratic regulator to address the target problems. Basic working principles for both controller are presented and their design specific parameters are also summarized.

4.1 | Fuzzy Control

4.1.1 | Working Principle of Fuzzy PID

Fuzzy PID (Proportional-Integral-Derivative) controllers are a variant of PID control that utilizes fuzzy logic to enhance the performance of traditional PID controllers. PID controllers are a common form of feedback control systems that regulate stability and responsiveness by adjusting proportional, integral, and derivative components. However, in certain nonlinear or difficult-to-precisely-model systems, traditional PID controllers might exhibit suboptimal performance.
Fuzzy PID, by introducing fuzzy logic, enables controllers to make more intelligent decisions when confronted with ambiguous or imprecise inputs. It employs fuzzy logic's fuzzy sets, fuzzy rules, and fuzzy inference to adjust the parameters of the PID controller, allowing it to better adapt to complex systems or uncertain environments. The advantage of a fuzzy PID controller lies in its ability to handle ambiguity and uncertainty, enhancing system robustness and stability, and delivering superior control effectiveness for systems that are challenging to precisely model.
Different from traditional PID, fuzzy PID does not need to specify the value of PID, nor does it need to specify the initial value like other optimizers. It can optimize the PID parameter in real time. Its working logic is shown in the figure below.

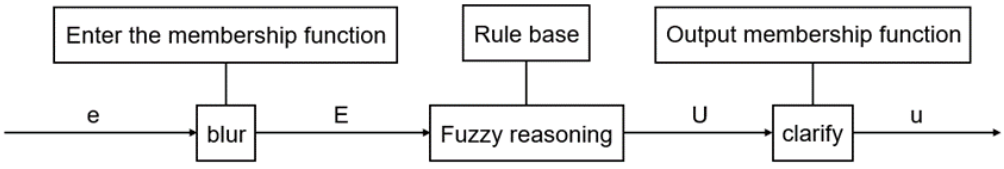


Figure 4.1: Working logic of Fuzzy PID

4.1.2 | Basic Concepts of Fuzzy PID

Basic Concepts of Fuzzy PID	
Amount of ambiguity	E,EC
Discussion domain	-240~240
Partition	-240~-180;-180~-120;-120~-60;60~0;0~60;60~120;120~180;180~240.
Degree of affiliation	The degree of subordination of a fuzzy subset.
Fuzzy subset	NB(Negative Big);NM(Negative Medium);NS(Negative small);ZO(Zero);PS(Positive Small);PM(Positive Medium);PB(Positive Big).

Figure 4.2: Basic Concepts of Fuzzy PID

4.1.3 | Fuzzy Controller Settings

Fuzzy Controller Settings			
Inputs	E,EC	**Or method**	Max
Outputs	KP,KI,KD	**Implication**	Min
FIS type	Mamdani	**Aggregation**	Max
And method	Min	**Defuzzification**	Centroid

Figure 4.3: Fuzzy Controller Settings

4.1.4 | Fuzzy Control Parameters

```
[Input1]
Name='E'
Range=[-3 3]
NumMFs=7
MF1='NB':'gaussmf',[0.7002 -3]
MF2='NM':'trimf',[-3 -2 0]
MF3='NS':'trimf',[-3 -1 1]
MF4='Z':'trimf',[-2 0 2]
MF5='PM':'trimf',[0 2 3]
MF6='PB':'gauss2mf',[0.7002 2.9 0.3405 3.099]
MF7='PS':'trimf',[-1 1 3]
```

```
[Input2]
Name='EC'
Range=[-3 3]
NumMFs=7
MF1='NB':'gaussmf',[0.7 -3]
MF2='NM':'trimf',[-3 -2 0]
MF3='NS':'trimf',[-3 -1 1]
MF4='Z':'trimf',[-2 0 2]
MF5='PS':'trimf',[-1 1 3]
MF6='PM':'trimf',[0 2 3]
MF7='PB':'gauss2mf',[0.6 2.9 0.6 3.1]
```

```
[Output3]
Name='KD'
Range=[-3 3]
NumMFs=7
MF1='NB':'gaussmf',[0.7002 -3]
MF2='NM':'trimf',[-3.002 -2.002 0]
MF3='NS':'trimf',[-3 -1.02 1]
MF4='Z':'trimf',[-2 0 2]
MF5='PS':'trimf',[-1 1 3]
MF6='PM':'trimf',[0 2 3]
MF7='PB':'gauss2mf',[0.6 2.9 0.3404 3.1]
```

```
[Output1]
Name='KI'
Range=[-0.06 0.06]
NumMFs=7
MF1='NB':'gaussmf',[0.015 -0.06]
MF2='NM':'trimf',[-0.06 -0.04 0]
MF3='NS':'trimf',[-0.06 -0.02 0.02]
MF4='Z':'trimf',[-0.04 0 0.04]
MF5='PS':'trimf',[-0.02 0.02 0.06]
MF6='PM':'trimf',[0 0.04 0.06]
MF7='PB':'gauss2mf',[0.014 0.058 0.006794 0.062]
```

```
[Output2]
Name='KP'
Range=[-0.3 0.3]
NumMFs=7
MF1='NB':'gaussmf',[0.07 -0.3]
MF2='NM':'trimf',[-0.3 -0.2 0]
MF3='NS':'trimf',[-0.3 -0.1 0.1]
MF4='Z':'trimf',[-0.2 0 0.2]
MF5='PS':'trimf',[-0.1 0.1 0.3]
MF6='PM':'trimf',[0 0.2 0.3]
MF7='PB':'gauss2mf',[0.06 0.29 0.03404 0.3102]
```

Figure 4.4: Fuzzy Control Parameters

4.1.5 | Rules

模糊PID规则表								
KP\KI\KD		EC						
		NB\1	NM\2	NS\3	ZO\4	PS\5	PM\6	PB\7
E	NB\1	PB\NB\PS\715	PB\NB\NS\713	PM\NM\NB\621	PM\NM\NB\621	PS\NS\NB\531	ZO\ZO\NM\442	ZO\ZO\PS\445
	NM\2	PB\NB\PS\715	PB\NB\NS\713	PM\NM\NB\621	PS\NS\NM\532	PS\NS\NM\532	ZO\ZO\NS\443	NS\ZO\ZO\344
	NS\3	PM\NB\ZO\614	PM\NM\NS\623	PM\NS\NM\632	PS\NS\NM\532	ZO\ZO\NS\443	NS\PS\NS\353	NS\PS\ZO\354
	ZO\4	PM\NM\ZO\624	PM\NM\NS\623	PS\NS\NS\533	ZO\ZO\NS\443	NS\PS\NS\353	NM\PM\NS\263	NM\PM\ZO\264
	PS\5	PS\NM\ZO\524	PS\NS\ZO\534	ZO\ZO\ZO\444	NS\PS\ZO\354	NS\PS\ZO\354	NM\PM\ZO\264	NM\PB\ZO\274
	PM\6	PS\ZO\PB\547	ZO\ZO\NS\443	NS\PS\PS\355	NM\PS\PS\255	NM\PM\PS\265	NM\PB\PS\275	NB\PB\PB\177
	PB\7	ZO\ZO\PB\447	ZO\ZO\PM\446	NM\PS\PM\256	NM\PM\PM\266	NM\PM\PS\265	NB\PB\PS\175	NB\PB\PB\177

Figure 4.5: Rules of Fuzzy Controller

4.1.6 | Block Diagrams

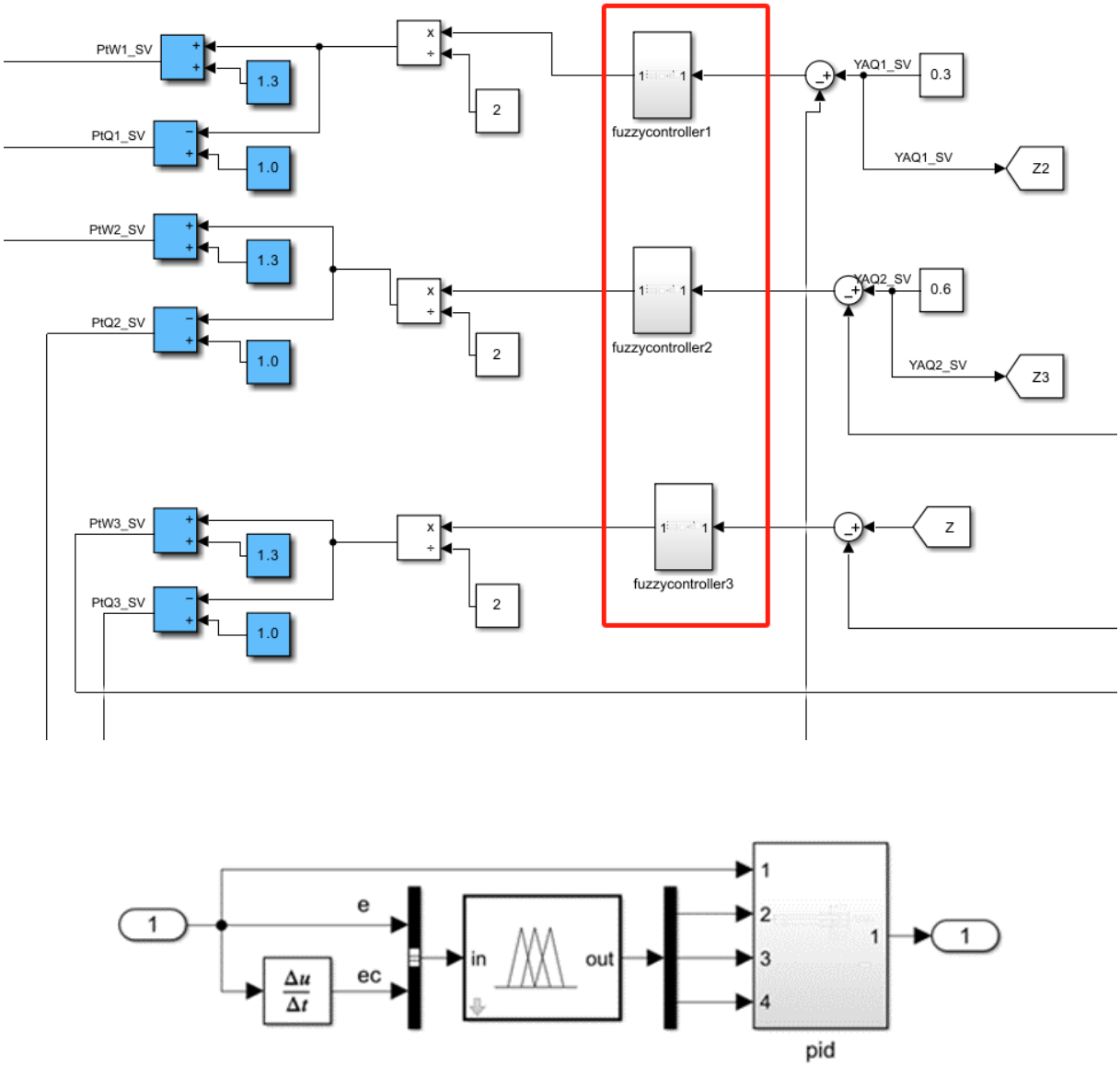


Figure 4.6: Block Diagram of Fuzzy Controller

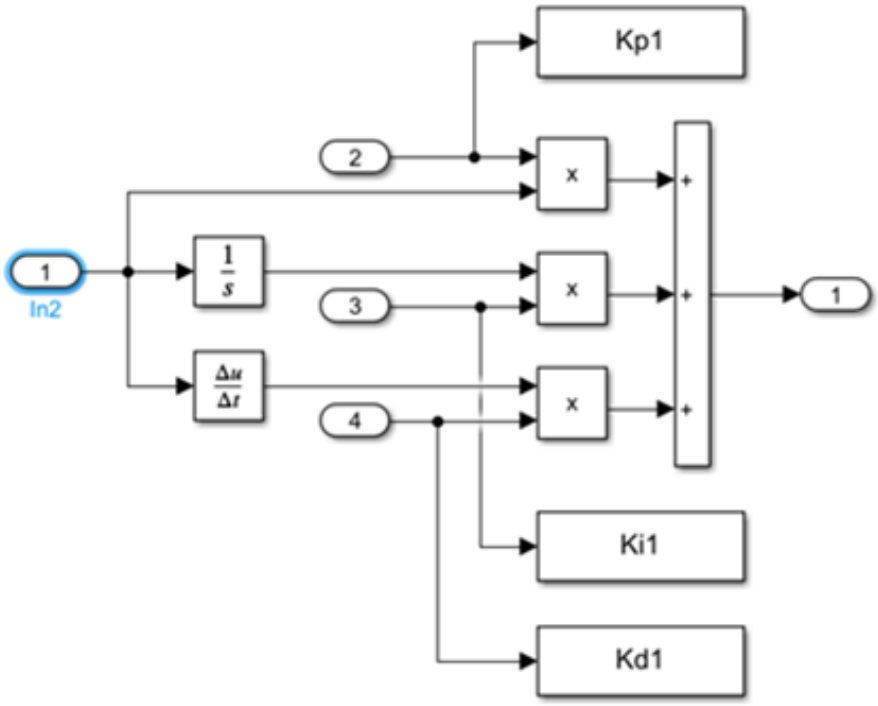


Figure 4.7: Block Diagram of PID Block

4.2 | Linear Quadratic Regulator

4.2.1 | Principle of LQR

LQR is one of the most frequently used optimal controllers in both industry and many research fields, which is an optimal controller for linear time-invariant system derived from the Bellman equation. By employing this LQR controller, the cost in the from of quadratic could be minimized along the state trajectory. The quadratic cost function is commonly written as

$$J(\mathbf{x}) = \int_0^\infty (\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \mathbf{u}^{\mathrm{T}}\mathbf{R}\mathbf{u})\mathrm{d}t \tag{4.1}$$

where the matrix $\mathbf{Q}$ and $\mathbf{R}$ are design parameters, which represent the weight of state cost and input cost respectively. As for the linear system, if we've already specified the state-space model of the system, namely the matrix A, B, C and D, we can directly obtain the optimal control law $\mathbf{K}$ by the MATLAB function *lqr(A,B,Q,R)*.

4.2.2 | Design Procedure

In this project, the original control problem is to manipulate the separator pressures to set the oxygen density in the permeation part at the desired values, which is an output tracking problem using the language of control engineering. Normally there exist two methods to address this kind of problem in the domain of linear control system design:

1. Derive the dynamics of the output value and reformulate the state vector to include the output. Then, the problem is transformed into the state tracking problem.
2. The other method is to schedule the desired state trajectory from the target output trajectory at first, and then impose the state tracking design procedure on the original dynamical system.

Although these two methods are intrinsically identical, difference exists in the implementation procedure, and the second method is employed in this project.

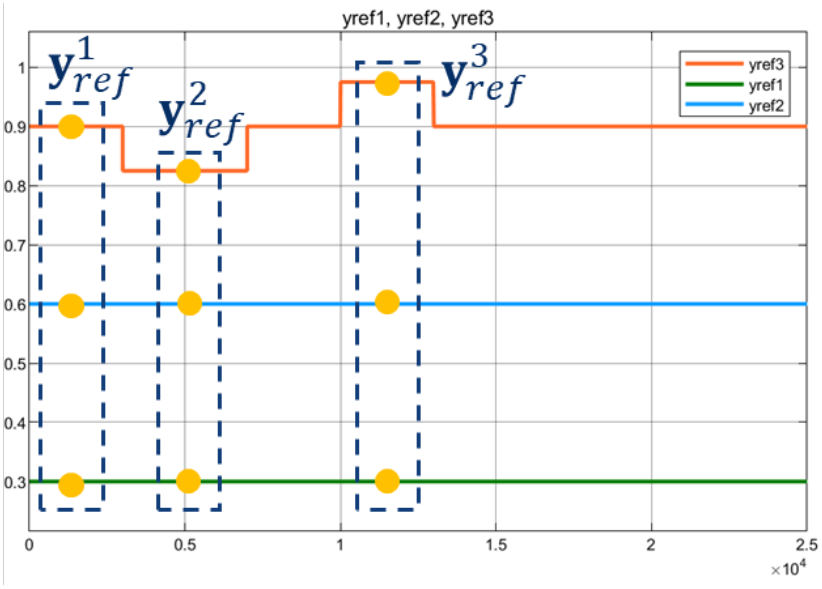


Figure 4.8: State Scheduling from the Desired Output Trajectory

As shown in Fig. 4.8, only three pairs of state are needed to be scheduled according to the special form of the output trajectory. Nonlinear optimization algorithm is employed to realize the scheduling with the state-space model derived in Section 2, as shown below.

$$[\mathbf{y}^1_{ref}, \mathbf{y}^2_{ref}, \mathbf{y}^3_{ref}]^{\mathrm{T}} \overset{Scheduling}{\longrightarrow} [\mathbf{x}^1_{ref}, \mathbf{x}^2_{ref}, \mathbf{x}^3_{ref}]^{\mathrm{T}} \tag{4.2}$$

After scheduling, we've obtained the desired state trajectory. The next task is to design a control system to realize the state tracking. In order to employ the linear control design methods, the system is linearized at the three scheduled state through Jacobian linearization, and it's fortunate to find that the linearized systems at three different scheduled state are all controllable and observable by the Kalman controllability and observability conditions. Hence, luenberger observer is designed to observe the full-state to construct the closed-loop state feedback control. The procedure above can be described with the equations below.

$$f(\cdot),\ h(\cdot) \overset{Linearization}{\longrightarrow} \mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D} \tag{4.3}$$

$$\dot{x} = \mathrm{A}\mathbf{x} + \mathrm{B}\mathbf{u} \tag{4.4}$$
$$\mathbf{u} = -\mathrm{K}(\mathbf{x}_{ref} - \hat{\mathbf{x}}) \tag{4.5}$$
$$\mathbf{y} = \mathrm{C}\mathbf{x} \tag{4.6}$$
$$\dot{\hat{\mathbf{x}}} = \mathrm{A}\hat{\mathbf{x}} + \mathrm{B}\mathbf{u} + \mathrm{L}(\mathbf{y} - \hat{\mathbf{y}}) \tag{4.7}$$
$$\hat{\mathbf{y}} = \mathrm{C}\hat{\mathbf{x}} \tag{4.8}$$
$$\mathrm{K} \longleftarrow lqr(\mathrm{A}, \mathrm{B}, \mathrm{Q}, \mathrm{R}) \tag{4.9}$$

In summary, the design parameters include L, Q and R, while these values designed in this project can be found in the Appendix II. The parameter L controls the dynamical characteristic of the Luenberger observer, and the parameters Q and R control the performance of the state tracking. By tuning these three parameters, we can find a satisfying result, which is shown in the next section.

5 | Evaluation

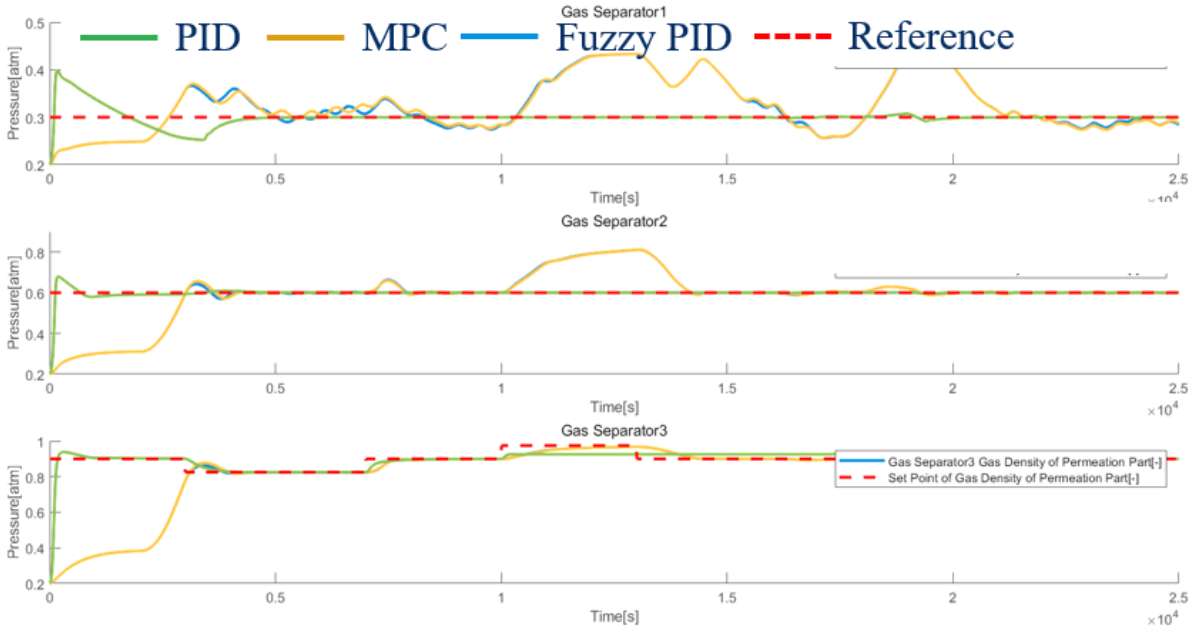


Figure 5.1: Performance comparison between PID, MPC and Fuzzy PID controller

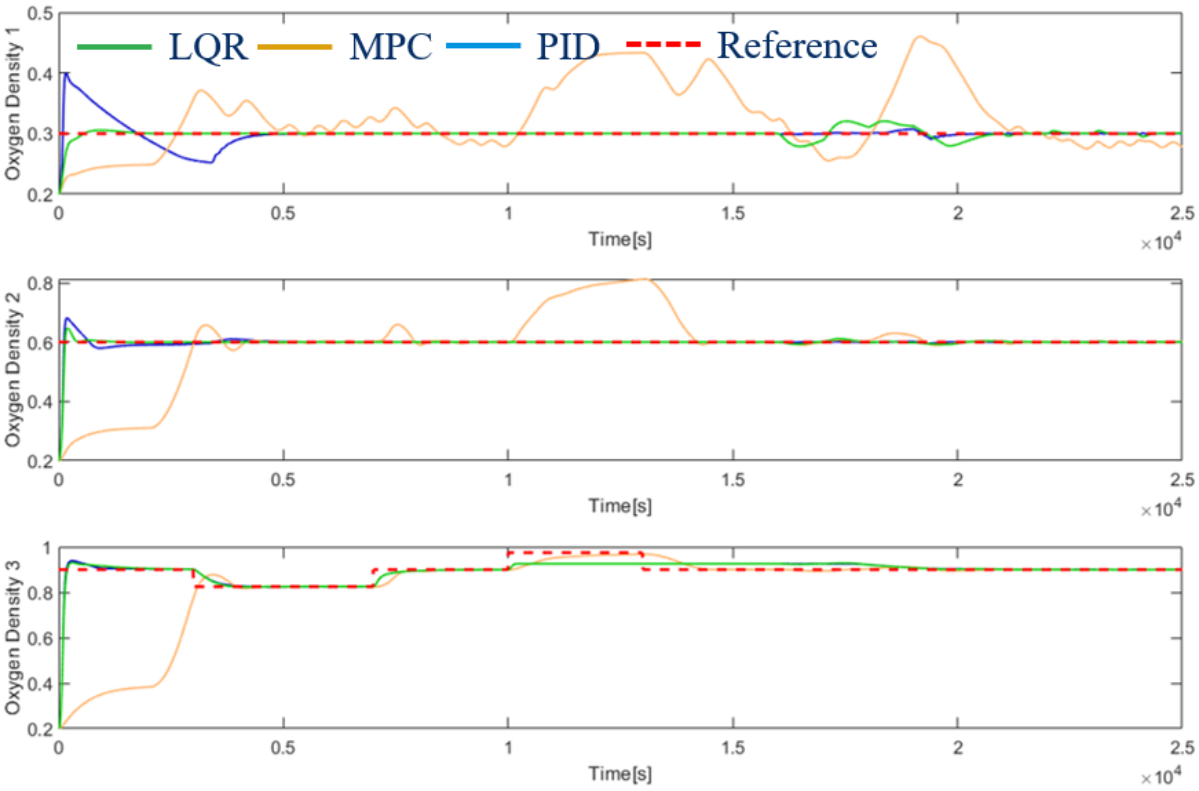


Figure 5.2: Performance comparison between PID, MPC and LQR controller

By comparing the final tracking results shown in the Fig. 5.1 and 5.2 from PID, MPC, Fuzzy PID and LQR controller, several remarks could be given as follows.

- As for the transient response, the LQR controller converges to the desired value with the fastest speed, since the cost function is minimized and the state error cost is optimized.
- As for the steady state error, the MPC and Fuzzy Controller achieves the minimal steady-state error during the time period from 10000s to 15000s. The steady state error in the LQR controller should be induced by the scheduling error.
- All controllers are successfully converge to the desired value after the disturbance, while the MPC controller exhibits the largest overshoot.
- Different trade-off exists in different controller.

6 | Conclusions

We have achieved the desired oxygen enrichment problem, from the problem formulation, to the system modeling, the system identification, the controller design, and we compare the different results obtained from different controller and have a detailed discussion. Throughout this project, we implement the knowledge acquired from the courses in the practical problem, which not only enhances our understanding on these knowledge, but also improves our problem-solving skills.
Finished this project, we've progressed a lot, not only our personal skills in many aspects, but also the team coherence. Better performance in future could be expected.

7 | Acknowledgement

All works of this project are finished by the team *SharpShooters*, supported by SHIEN-MING WU SCHOOL OF INTELLIGENT ENGINEERING, Dr. Gang Chen and all people who provided valuable assistance. The official document[1] is the major reference material regarding all technical details.

8 | References

[1] Hiroumi Mita. Simulation of cascade pid control and model predictive control for a gas separation plant. In *MATLAB Central File Exchange*, Retrieved January 6, 2024.

Appendix I: Simulink Model

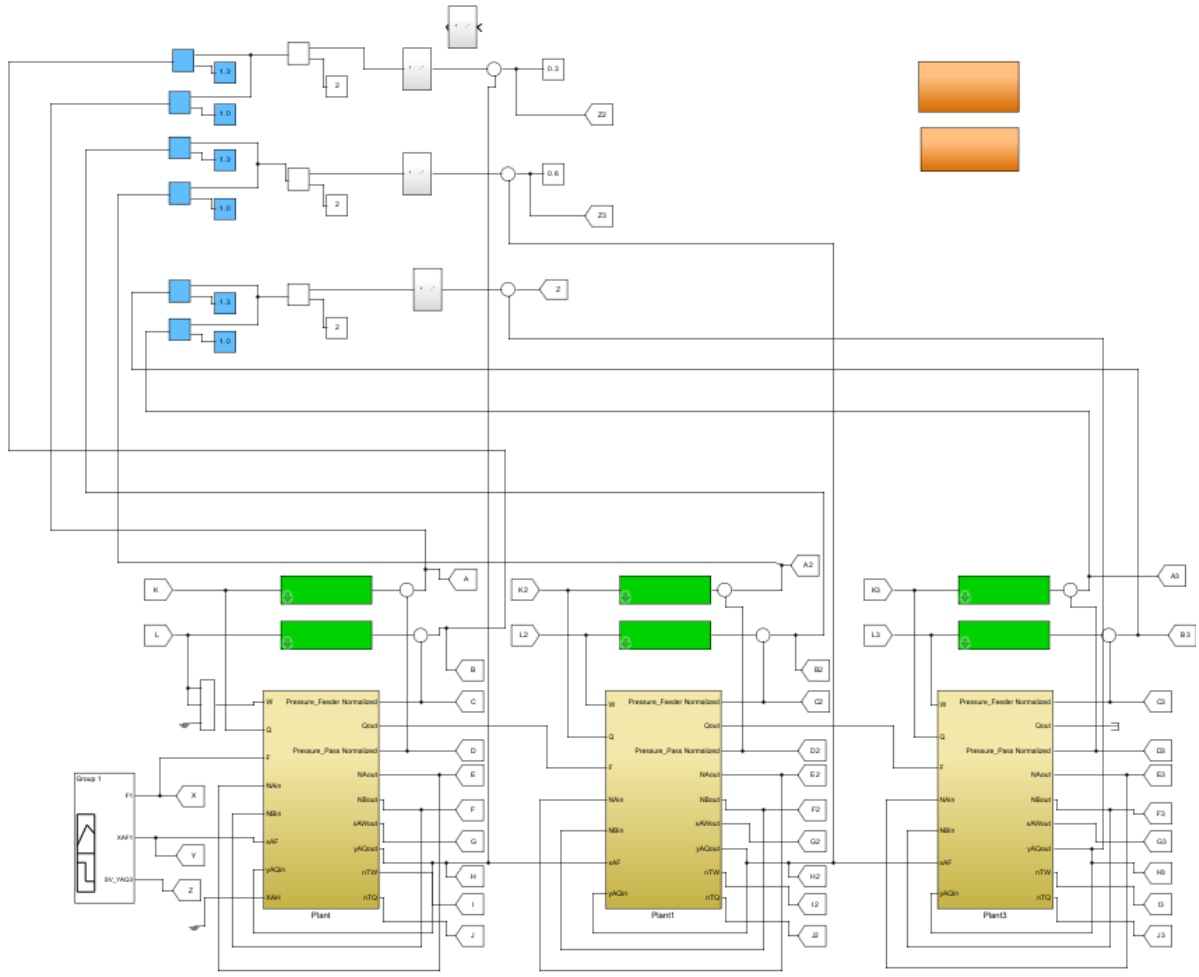


Figure 8.1: Illustration for the Simulink model of Fuzzy PID

Appendix II: MATLAB Script

```
clear;close all;clc
%% State-Space Modeling of the gas separator system
C_A = 0.00001; C_B = 0.000001;
Area = 5;
k = 8.314*273.15/25;
N_AQ_1_init = 223.2143; N_AW_1_init = 223.2143; N_tQ_1_init = 1116.1; N_tW_1_init = 1116.1;
N_AQ_2_init = 223.2143; N_AW_2_init = 223.2143; N_tQ_2_init = 1116.1; N_tW_2_init = 1116.1;
N_AQ_3_init = 223.2143; N_AW_3_init = 223.2143; N_tQ_3_init = 1116.1; N_tW_3_init = 1116.1;
pre_constant = 1013*100;
% PID Inner Loop Parameters
Kp_1W = -20; Ki_1W = -20*0.014925;Kp_1Q = -17; Ki_1Q = -17*0.014925;
Kp_2W = -20; Ki_2W = -20*0.014925;Kp_2Q = -20; Ki_2Q = -20*0.014925;
Kp_3W = -30; Ki_3W = -30*0.014925;Kp_3Q = -30; Ki_3Q = -30*0.014925;

syms F x_AF...
     N_tW1 N_tQ1 N_AW1 N_AQ1 P_tW1 P_tQ1 y_AQ1 W12 Q1...
     N_tW2 N_tQ2       N_AQ2 P_tW2 P_tQ2 y_AQ2     Q2...
     N_tW3 N_tQ3       N_AQ3 P_tW3 P_tQ3 y_AQ3     Q3...
     eta1 eta2 eta3 eta4 eta5 eta6...
     SV1 SV2 SV3 SV4 SV5 SV6

f1_sym  =  SV1 - N_tW1.*k./pre_constant;
f2_sym  =  SV2 - N_tQ1.*k./pre_constant;
f3_sym  =  SV3 - N_tW2.*k./pre_constant;
f4_sym  =  SV4 - N_tQ2.*k./pre_constant;
f5_sym  =  SV5 - N_tW3.*k./pre_constant;
f6_sym  =  SV6 - N_tQ3.*k./pre_constant;

f7_sym  = -Area.*k.*C_B.*N_tW1 + Area.*k.*C_B.*N_tQ1 + Area.*k.*(C_B-C_A).*N_AW1...
          + Area.*k.*(C_A-C_B).*N_AQ1 + F - Kp_1W.*(SV1 - N_tW1.*k./pre_constant) - Ki_1W.*eta1;

f8_sym  =  Area.*k.*C_B.*N_tW1 - Area.*k.*C_B.*N_tQ1 - Area.*k.*(C_B-C_A).*N_AW1...
          - Area.*k.*(C_A-C_B).*N_AQ1 - Kp_1Q.*(SV2 - N_tQ1.*k./pre_constant) - Ki_1Q.*eta2;

f9_sym  = -W12.*N_AW1./N_tW1 - Area.*k.*C_A.*N_AW1 + Area.*k.*C_A.*N_AQ1 + F.*x_AF;

f10_sym  = -(Kp_1Q.*(SV2 - N_tQ1.*k./pre_constant) + Ki_1Q.*eta2).*N_AQ1./N_tQ1 + Area.*k.*C_A.*N_AW1...
           - Area.*k.*C_A.*N_AQ1;

f11_sym  = -Area.*k.*((C_A - C_B).*N_AQ1./N_tQ1 + C_B).*N_tW2 + Area.*k.*C_B.*N_tQ2...
           + Area.*k.*(C_A-C_B).*N_AQ2 + Kp_1Q.*(SV2 - N_tQ1.*k./pre_constant) + Ki_1Q.*eta2...
           - Kp_2W.*(SV3 - N_tW2.*k./pre_constant) - Ki_2W.*eta3;

f12_sym  =  Area.*k.*((C_A - C_B).*N_AQ1./N_tQ1 + C_B).*N_tW2 - Area.*k.*C_B.*N_tQ2...
            - Area.*k.*(C_A-C_B).*N_AQ2 - Kp_2Q.*(SV4 - N_tQ2.*k./pre_constant) - Ki_2Q.*eta4;

f13_sym  = -(Kp_2Q.*(SV4 - N_tQ2.*k./pre_constant) + Ki_2Q.*eta4).*N_AQ2./N_tQ2...
           + Area.*k.*C_A.*(N_AQ1./N_tQ1).*N_tW2 - Area.*k.*C_A.*N_AQ2;

f14_sym  = -Area.*k.*((C_A - C_B).*N_AQ2./N_tQ2 + C_B).*N_tW3 + Area.*k.*C_B.*N_tQ3...
           + Area.*k.*(C_A-C_B).*N_AQ3 + Kp_2Q.*(SV4 - N_tQ2.*k./pre_constant)...
           + Ki_2Q.*eta4 - Kp_3W.*(SV5 - N_tW3.*k./pre_constant) - Ki_3W.*eta5;

f15_sym  =  Area.*k.*((C_A - C_B).*N_AQ2./N_tQ2 + C_B).*N_tW3 - Area.*k.*C_B.*N_tQ3...
            - Area.*k.*(C_A-C_B).*N_AQ3 - Kp_3Q.*(SV6 - N_tQ3.*k./pre_constant) - Ki_3Q.*eta6;


f16_sym  = -(Kp_3Q.*(SV6 - N_tQ3.*k./pre_constant) + Ki_3Q.*eta6).*N_AQ3./N_tQ3...
           + Area.*k.*C_A.*(N_AQ2./N_tQ2).*N_tW3 - Area.*k.*C_A.*N_AQ3;

f_sym   = [f1_sym;f2_sym;f3_sym;f4_sym;f5_sym;f6_sym;f7_sym;f8_sym;f9_sym;f10_sym;...
           f11_sym;f12_sym;f13_sym;f14_sym;f15_sym;f16_sym];
f_fun   = matlabFunction(f_sym);

h1_sym = N_tW1.*k./pre_constant;
h2_sym = N_tQ1.*k./pre_constant;
h3_sym = N_AQ1./N_tQ1;
h4_sym = N_tW2.*k./pre_constant;
h5_sym = N_tQ2.*k./pre_constant;
```

```
h6_sym = N_AQ2./N_tQ2;
h7_sym = N_tW3.*k./pre_constant;
h8_sym = N_tQ3.*k./pre_constant;
h9_sym = N_AQ3./N_tQ3;
h_sym  = [h1_sym;h2_sym;h3_sym;h4_sym;h5_sym;h6_sym;h7_sym;h8_sym;h9_sym];
h_fun  = matlabFunction(h_sym);

%% Jacobian Linearization
A_sym = jacobian(f_sym,[eta1;eta2;eta3;eta4;eta5;eta6;N_tW1;N_tQ1;N_AW1;N_AQ1;N_tW2;...
                        N_tQ2;N_AQ2;N_tW3;N_tQ3;N_AQ3]);
B_sym = jacobian(f_sym,[SV1;SV2;SV3;SV4;SV5;SV6]);
P_sym = jacobian(f_sym,[F;x_AF]);
C_sym = jacobian(h_sym,[N_tW1;N_tQ1;N_AW1;N_AQ1;N_tW2;N_tQ2;N_AQ2;N_tW3;N_tQ3;N_AQ3]);

A = matlabFunction(A_sym);
B = matlabFunction(B_sym);
P = matlabFunction(P_sym);
C = matlabFunction(C_sym);


%% Process
load("data_setup\input.mat");u = u';
load("data_setup\mpc\WQ.mat");
load("data_setup\mpc\F_mpc.mat");
load("data_setup\mpc\x_AF_mpc.mat");
load('data_setup\mpc\SV.mat');
load("data_setup\mpc\z.mat");z = z';
load("SV_LQR.mat");
load("y_ref.mat");

F = F_mpc'; x_AF = x_AF_mpc';

tout = 25000;
W12 = zeros(1,tout);
n = 16; m = 9;
x = zeros(n,tout);
y_true = zeros(9,tout);
x(:,1) = [0;0;0;0;0;0;...
          N_tW_1_init;N_tQ_1_init;N_AW_1_init;N_AQ_1_init;...
          N_tW_2_init;N_tQ_2_init;N_AQ_2_init;...
          N_tW_3_init;N_tQ_3_init;N_AQ_3_init];
x_dot_true = zeros(n,tout);
y_true(:,1) = h_fun(x(10,1),x(13,1),x(16,1),x(8,1),x(12,1),x(15,1),x(7,1),x(11,1),x(14,1));
dt = 1;

SV = SV_LQR(:,1:tout);

for i = 1:tout-1

    % system propagation

    W12(i) = Kp_1W*(SV(1,i) - x(7,i).*k./pre_constant) + Ki_1W*x(1,i);
    if  W12(i) < 0
        W12(i) = 0;
    end

    x_dot_true(:,i) = f_fun(F(i),x(10,i),x(13,i),x(16,i),x(9,i),x(8,i),x(12,i),x(15,i),x(7,i),...
                            x(11,i),x(14,i),SV(1,i),SV(2,i),SV(3,i),SV(4,i),SV(5,i),SV(6,i),...
                            W12(i),x(1,i),x(2,i),x(3,i),x(4,i),x(5,i),x(6,i),x_AF(i));
    x(:,i+1) = x_dot_true(:,i)*dt + x(:,i);
    y_true(:,i+1) = h_fun(x(10,i+1),x(13,i+1),x(16,i+1),x(8,i+1),x(12,i+1),x(15,i+1),x(7,i+1),...
                          x(11,i+1),x(14,i+1));
end
%% Output Performance

z = z';
figure

subplot(3,1,1);hold on;grid on;box on;
plot(y_true(3,:),'b','LineWidth',1.5);plot(z(1,1:25000),'--r','LineWidth',1.5);
legend('True Concentration','Desired Concentration');
xlabel('t (s)'); ylabel('y1');
xlim([1,tout]);
```

```
143
144 subplot(3,1,2);hold on;grid on;box on;
145 plot(y_true(6,:),'b','LineWidth',1.5);plot(z(2,1:25000),'--r','LineWidth',1.5);
146 legend('True Concentration','Desired Concentration');
147 xlabel('t (s)'); ylabel('y2');
148 xlim([1,tout]);
149
150 subplot(3,1,3);hold on;grid on;box on;
151 plot(y_true(9,:),'b','LineWidth',1.5);plot(z(3,1:25000),'--r','LineWidth',1.5);
152 legend('True Concentration','Desired Concentration');
153 xlabel('t (s)'); ylabel('y3');
154 xlim([1,tout]);
```