Summer Semester (2019)

Assignment on Bioreactor (5)

Course Name: Advanced Control Technology

Assignment Part: 01

Submitted By:

Sayed Rafay Bin Shah
Matriculation No.: 30043073

Md. Saiful Islam Sajol
Matriculation No.: 30042836

Submitted To:

Prof. Dr.-Ing. Andreas Schwung

Table of Contents

1. Introduction

Bioreactor is a system where a vessel carries out a biological reaction and is used to process aerobic cells for conducting cellular or enzymatic immobilization [1]. Bioreactors are used in various applications including food, beverages and pharmaceuticals industries. Mathematical modelling plays a vital role in bioreactor technology. In this assignment a simple bioreactor system is given to analyze state space modelling where a vessel containing glucose is connected to a pump to pass the glucose to the bioreactor for tissue culturing or enzymatic immobilization.

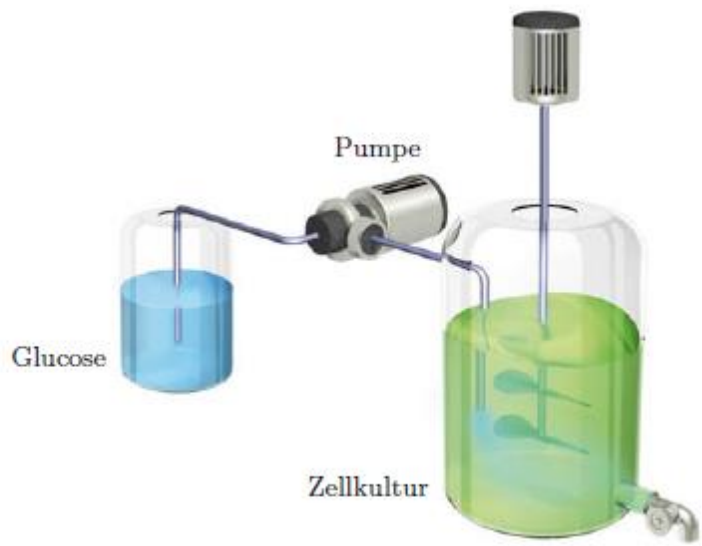

Fig. 1: Bioreactor System.

The given state space system is modeled as follows:

$$\dot{\boldsymbol{x}} = \boldsymbol{a}(\boldsymbol{x}) + \boldsymbol{b}(\boldsymbol{x}) \cdot u = \begin{bmatrix} \mu(x_2) \cdot x_1 \\ -\dfrac{1}{\alpha}\mu(x_2) \cdot x_1 \end{bmatrix} + \begin{bmatrix} -x_1 \\ K - x_2 \end{bmatrix} u,$$
$$y = g(\boldsymbol{x}) = \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{x} .$$

Where,

$$\mu(x_2) = \frac{\mu_0 x_2}{k_1 + x_2 + k_2 x_2^2}$$

Maximal growth rate, $\mu_0 = 2$
Affinity constant, $k_1 = 0.06$
Affinity constant, $k_2 = 0.3$
Feed concentration of glucose, K = 2
Yield constant, $\alpha = 0.7$
Concentration of biomass and substrate = x_1 and x_2 respectively.

2. Linearization of the System Model in a suitable Operation Point:

The model provided in the task is in non-linear state space equation form. By finding suitable operating points, the system can be linearized.
The linear system representation should be of the form:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}x(t) + \boldsymbol{B}u(t)$$
$$\boldsymbol{y}(t) = \boldsymbol{C}x(t) + \boldsymbol{D}u(t)$$

Where,

A is the state matrix,

B is the input matrix,

C is the output matrix,

D is the direct transmission matrix.

By taking the system dynamics as zero, the operating points are calculated using the MATLAB script shown in Fig. 2. The input value of u is arbitrary chosen equal to 1.3.

```
syms x1 x2
u0 = 2;
u = 1.3;
k1 = 0.06;
k2 = 0.3;
k = 2;
alpha = 0.7;

fn1 = ((u0*x1*x2)/(k1+x2+k2*(x2^2))) - (x1*u);
fn2 = (-(u0*x1*x2)/(alpha*(k1+x2+k2*(x2^2)))) + ((k-x2)*u);
[solx1, solx2] = solve([fn1 == 0, fn2 == 0], [x1,x2], 'real', true);
x1_roots = zeros;
x1_roots = vpa(solx1);
x2_roots = zeros;
x2_roots = vpa(solx2);
x1_value = x1_roots(end);
x2_value = x2_roots(end);
fprintf('The operating points are x1 = %f, x2 = %f\n', x1_value, x2_value);
```

Fig. 2: MATLAB Script to obtain operating points.

The values are obtained as follows:

x_1= 0.22714

x_2 =1.675505

With the help of Taylor´s Series we can linearize the non-linear model around the operating points.

The matrix can be obtained into the form as shown below:

A=$\begin{bmatrix} \left.\frac{\partial f_1}{\partial x_1}\right|_{x_1,x_2,u} & \left.\frac{\partial f_1}{\partial x_2}\right|_{x_1,x_2,u} \\ \left.\frac{\partial f_2}{\partial x_1}\right|_{x_1,x_2,u} & \left.\frac{\partial f_2}{\partial x_2}\right|_{x_1,x_2,u} \end{bmatrix}$

$$B = \begin{bmatrix} -x_1 \\ k - x_2 \end{bmatrix}$$

C= [1 0]

D=0

The derivatives of the matrix **A** are found using the MATLAB script in Figure 6. The resulting expressions are shown in Figure7.

```
syms u0 u k1 k2 k alpha x1 x2
fn1 = ((u0*x1*x2)/(k1+x2+k2*(x2^2))) - (x1*u);
fn2 = (-(u0*x1*x2)/(alpha*(k1+x2+k2*(x2^2)))) + ((k-x2)*u);
A11 = diff(fn1,x1)
A12 = diff(fn1,x2)
A21 = diff(fn2,x1)
A22 = diff(fn2,x2)
```

Fig. 6: MATLAB Script to obtain differential elements of matrix **A**.

```
A11 =

(u0*x2)/(k2*x2^2 + x2 + k1) - u


A12 =

(u0*x1)/(k2*x2^2 + x2 + k1) - (u0*x1*x2*(2*k2*x2 + 1))/(k2*x2^2 + x2 + k1)^2


A21 =

-(u0*x2)/(alpha*(k2*x2^2 + x2 + k1))


A22 =

(u0*x1*x2*(2*k2*x2 + 1))/(alpha*(k2*x2^2 + x2 + k1)^2) - (u0*x1)/(alpha*(k2*x2^2 + x2 + k1)) - u
```

Fig. 7: Elements of matrix **A**.

Hence the matrix becomes,

$$A = \begin{bmatrix} \dfrac{\mu_0\, x_2}{k_1 + x_2 + k_2\, x^2{}_2} - \mathrm{u} & \dfrac{(k_1 + x_2 + k_2\, x^2{}_2)\mu_o\ x_1 - \mu_o\ x_1\ x_2\,(1 + 2k_2\, x_2)}{(k_1 + x_2 + k_2\, x^2{}_2)^2} \\ \dfrac{-1}{\alpha}\ \dfrac{\mu_0\, x_2}{k_1 + x_2 + k_2\, x^2{}_2} & \dfrac{-\alpha(k_1 + x_2 + k_2\, x^2{}_2)\mu_o\ x_1 - (-\mu_o\ x_1\ x_2)(\alpha + \alpha 2k_2\, x_2)}{(k_1 + x_2 + k_2\, x^2{}_2)^2\alpha^2} - u \end{bmatrix}$$

Substituting the values of x_1, x_2, k_1, k_2, K and u, the matrices **A** and **B** are found. Thus the linear state space representation can be shown as follows:

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} -5.1193 * 10^{-8} & -0.0535 \\ -1.8571 & -1.2236 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} -0.227147 \\ x_2\, 0.324495 \end{bmatrix} u$$

$$y = [1\ \ 0] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + 0$$

3. Simulink Model of Bioreactor System

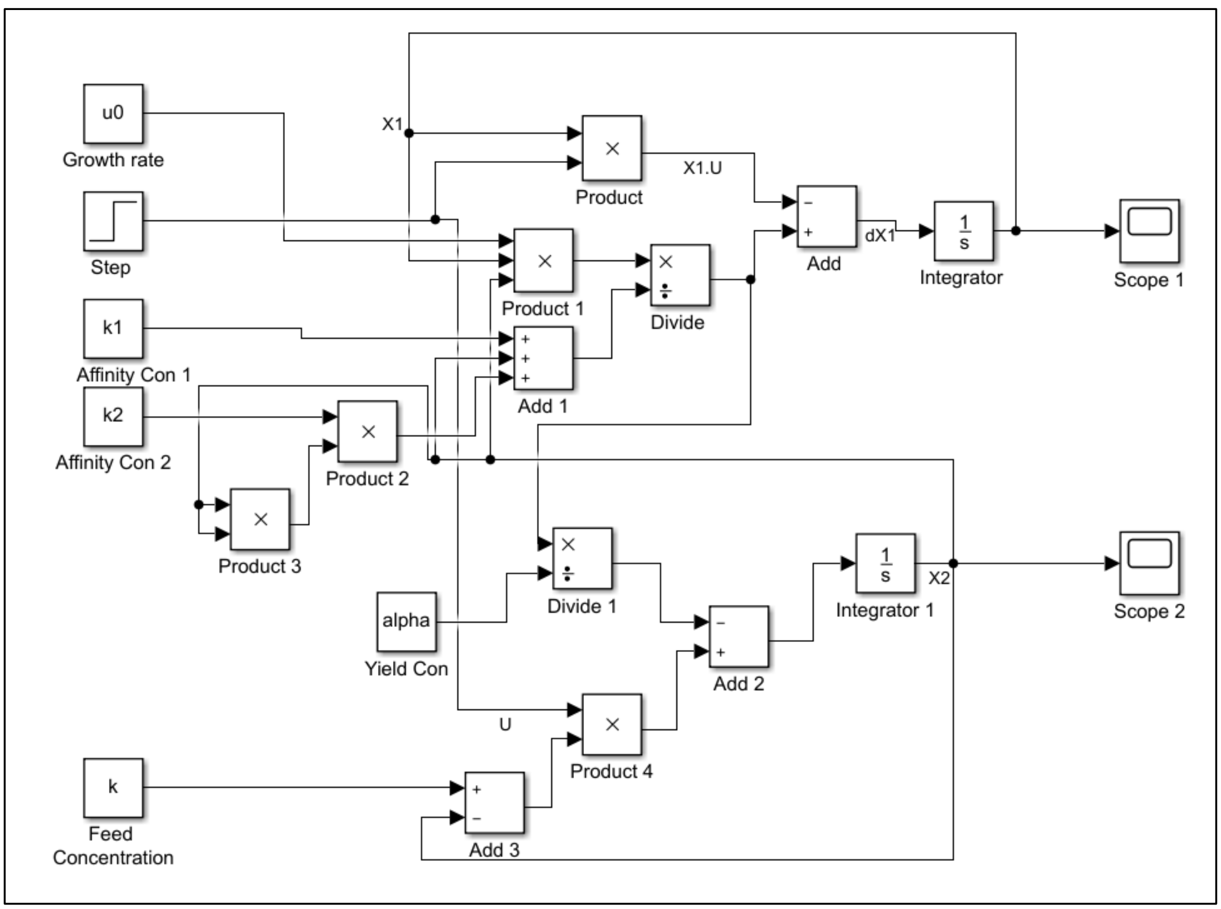

Fig. 8: Nonlinear model of the Bioreactor system in Simulink

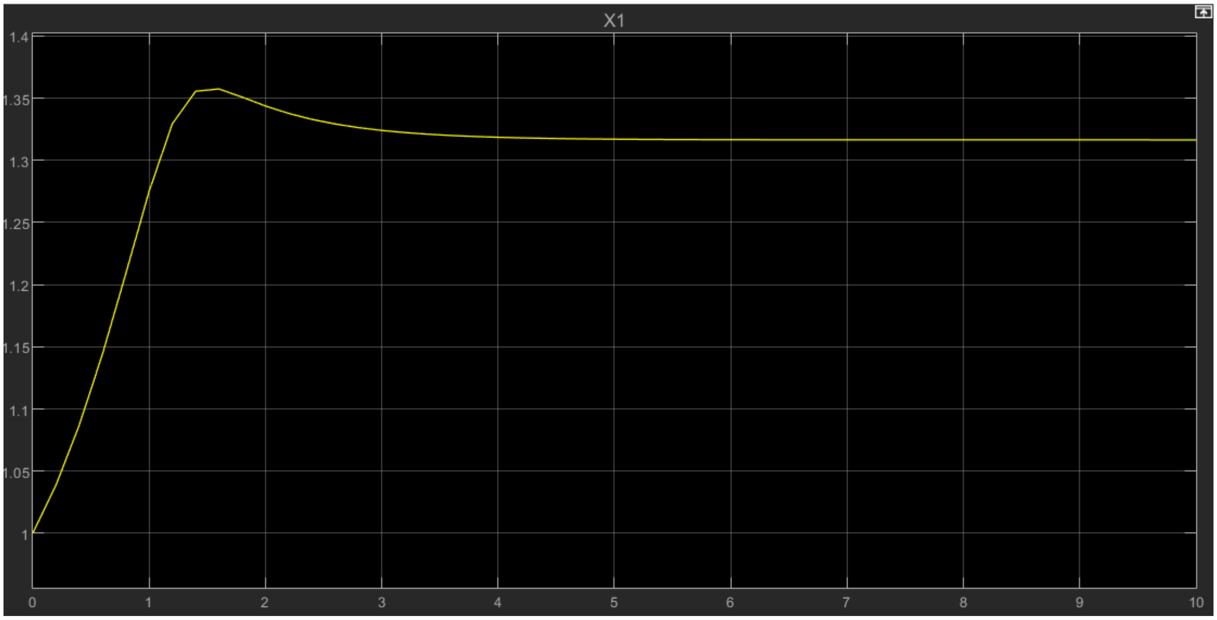

Fig. 9: Output graph of scope 1 (Value of x_1).

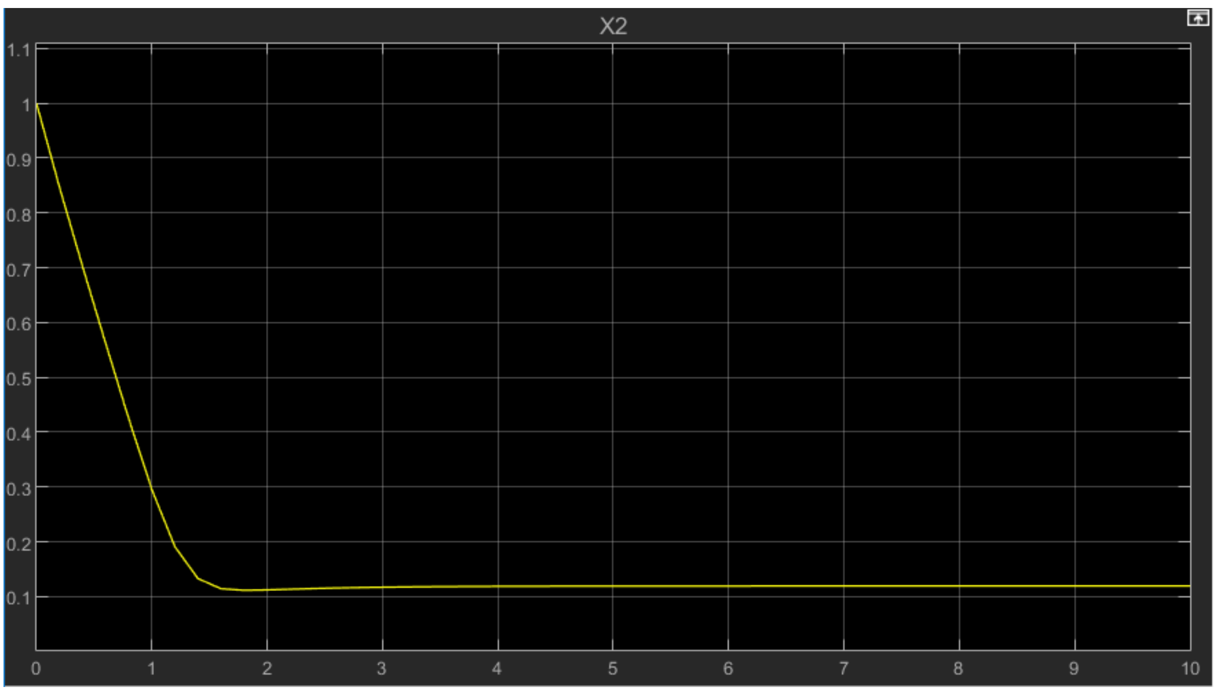

Fig. 9: Output graph of scope 2 (Value of x_2).

4. Analysis of the System Behaviour: Stability, Controllability & Observabilty

4.1 Stability

To evaluate the stability of the system, the eigenvalues of the matrix **A** are determined using the MATLAB script shown in Fig. 10.

```
A = [-5.1193e-08 -0.0535; -1.8571 -1.2236];
B = [-0.227147; 0.324495];
C = [1 0];
D = 0;

e=eig(A)                    %For Stability
```

Fig. 10: MATLAB Script to determine eigenvalues of matrix **A**.

The eigenvalues obtained are

```
e =

    0.0764
   -1.3000
```

It can be seen that one value is positive and the other is negative. This means the system is **unstable**.

4.2 Controllability

To determine the controllability of the system, the rank of controllability matrix is first determined using the script shown in Fig. 10. If the rank of controllability matrix is equal to the length of matrix **A**, then the system is controllable, otherwise not [2].

```
Con_mat = ctrb(A,B);     %For Controllability
length(A);
rank(Con_mat);
fprintf('Length of Matrix A = %d\n', length(A));
fprintf('Rank of Controllability Matrix = %d\n', rank(Con_mat));
```

Fig. 10: MATLAB Script to determine the rank of Controllability Matrix.

The result obtained are as follows:

```
Length of Matrix A = 2
Rank of Controllability Matrix = 2
```

Hence, the system is **controllable**.

4.3 Observability

To determine the observability of the system, the rank of observability matrix is first determined using the script shown in Fig. 11. If the rank of observability matrix is equal to the rank of matrix **A**, then the system is observable, otherwise not [2].

```
Obs=obsv(A,C);            %For Observability
rank(A);
rank(Obs);
fprintf('Rank of Matrix A = %d\n', rank(A));
fprintf('Rank of Observability Matrix = %d\n', rank(Obs));
```

Fig. 11: MATLAB Script to determine the rank of Observability Matrix.

The result obtained are as follows:

```
Rank of Matrix A = 2
Rank of Observability Matrix = 2
```

Hence, the system is **observable**.

5. Design of State Space Controller and Observer

5.1 Design of State Space Controller using Pole Placement Approach

For the design of the State Space Controller, the Pole-Placement approach is chosen. Hence, by choosing poles arbitrarily and evaluating the response for those poles, the state space controller is designed. The poles for which a faster response time with relatively less distortion can be obtained will be selected.

Firstly, the feedback gain matrix **K** is determined for various poles using the MATLAB script shown in Fig. 12.

For Poles, P_c = -2+0.3j, -2-0.3j

```
Pc = [-2+j*0.3 -2-j*0.3];     %POLES
K = place(A,B,Pc)            %Feedback Gain Matrix
```

Fig. 12: MATLAB Script to determine Feedback Gain Matrix **K.**

The **K** matrix is,

K = [-4.6024   -3.2208]

Similarly,

for Poles, P_c = -2+5j, -2-5j

K = [-2.0223   -1.4156]

The response of the controller to these poles are obtained by running the MATLAB script shown in Fig. 13. The analysis of the responses is described in Art 6.1 The controller is designed using the poles P_c = -2+0.3j, -2-0.3j.

```
A = [-5.1193e-08 -0.0535; -1.8571 -1.2236];
B = [-0.227147; 0.324495];
C = [1 0];
D = 0;

Pc = [-2+j*0.3 -2-j*0.3];     %POLES
K = place(A,B,Pc)         %Feedback Gain Matrix

sys = ss(A-B*K, eye(2), eye(2), eye(2));
t= 0:0.01:5;
x = initial(sys, [0.227147; 1.675505], t);
x1 = [1,0]*x';
x2 = [0,1]*x';

title('Response of Controller to Initial Condition');
subplot(2,1,1);
plot(t,x1);
ylabel('State Variable x1');
grid
subplot(2,1,2);
plot(t,x2);
ylabel('State Variable x2');
grid
```

Fig. 13: MATLAB Script to obtain initial response of state space controller to varying poles values.

The controller model is also drawn using MATLAB/Simulink as shown in Fig. 14.

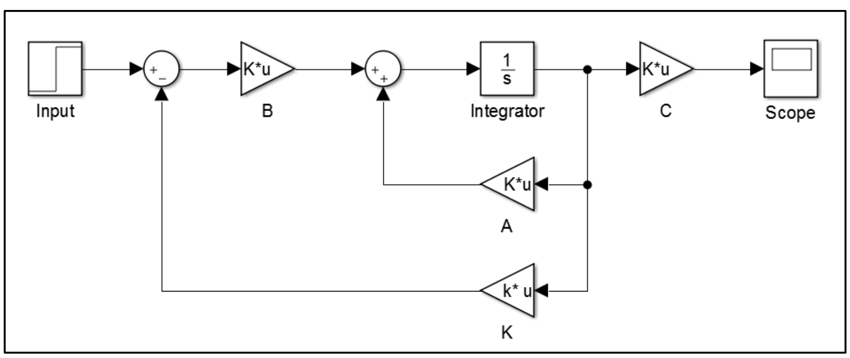

Fig. 14: Simulink model for Controller.

5.2 Design of State Space Observer

The response time of the observer must be several times faster than that of the controller. Hence, the observer poles are chosen more towards the left to obtain a faster stabilizing time than the controller. In designing the observer, error estimation is also performed to compensate for the inaccuracies of the matrices **A** and **B**.

At first, the observer feedback gain matrix is determined for two arbitrarily chosen set of poles using the script shown in Fig. 15.

For Poles, P_o = -10, -11

```
Po = [-10 -11];
Ko = place(A',C',Po)'
```

Fig. 15: MATLAB Script to obtain Observer Feedback Gain Matrix.

The feedback gain matrix is,

$$\mathbf{K_o} = \begin{bmatrix} 0.0198 \\ -1.6056 \end{bmatrix}$$

Similarly for *Poles, P_o* = -5, -6

$$\mathbf{K_o} = \begin{bmatrix} 9.7764 \\ -339.0084 \end{bmatrix}$$

The response of the observer to these poles are obtained by running the MATLAB script shown in Fig. 16. The analysis of the responses is described in Art 6.2. The observer is designed using the poles P_o = -10, -11.

```
A = [-5.1193e-08 -0.0535; -1.8571 -1.2236];
B = [-0.227147; 0.324495];
C = [1 0];
D = 0;

Pc = [-2+j*0.3 -2-j*0.3];
K = place(A,B,Pc)
Po = [-10 -11];
Ko = place(A',C',Po)'

sys = ss([A-B*K B*K; zeros(2,2) A-Ko*C], eye(4), eye(4), eye(4));
t=0:0.01:3;
x= initial(sys,[0.227147;0;1.675505;0],t);
x1= [1 0 0 0]*x';
x2= [0 1 0 0]*x';
e1= [0 0 1 0]*x';
e2= [0 0 0 1]*x';

subplot(2,2,1); plot(t,x1); grid
title('Response of Observer to Initial Condition')
ylabel('State Variable x1')
subplot(2,2,2); plot(t,x2); grid
ylabel('State Variable x2')
subplot(2,2,3); plot(t,e1); grid
ylabel('Error State Variable e1')
subplot(2,2,4); plot(t,e2); grid
ylabel('Error State Variable e1')
```

Fig. 16: MATLAB Script to obtain initial response of state space observer to varying poles values.

The observer model is also drawn using MATLAB/Simulink as shown in Fig. 17.

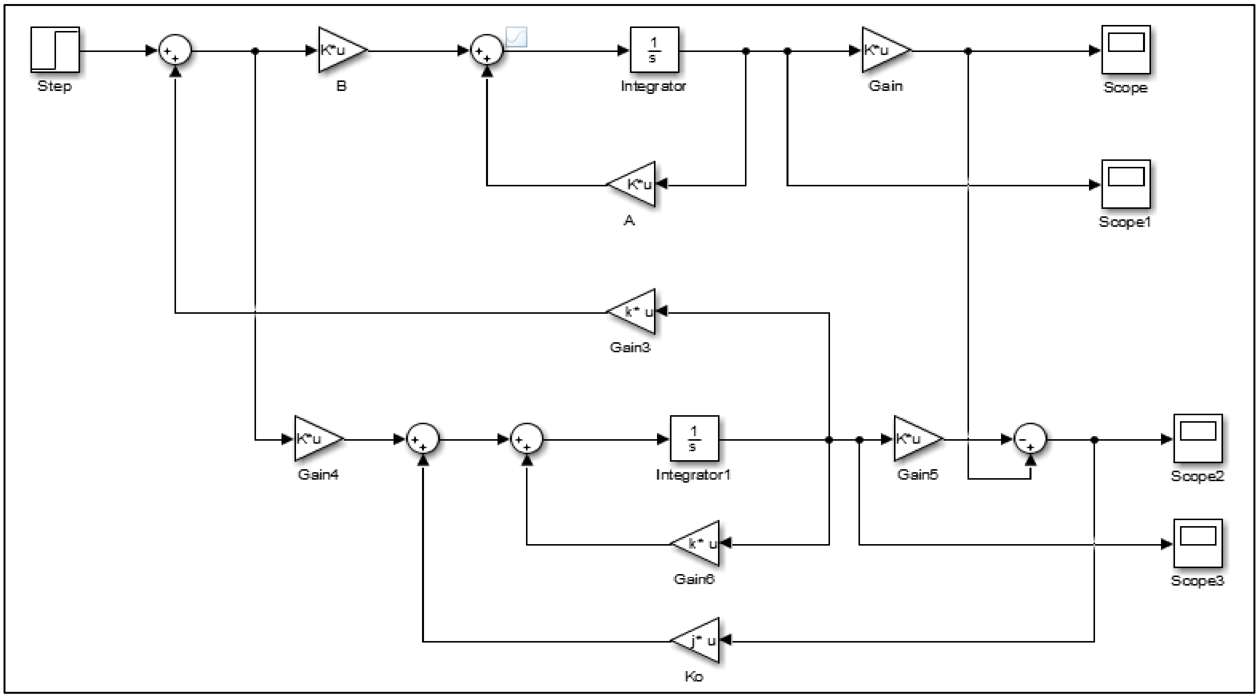

Fig. 17: Simulink model for Observer.

6. Analysis of Controller and Observer Response

6.1 Analysis of State Space Controller

The response from the designed controller for different poles are shown in Fig. 18 and Fig. 19.

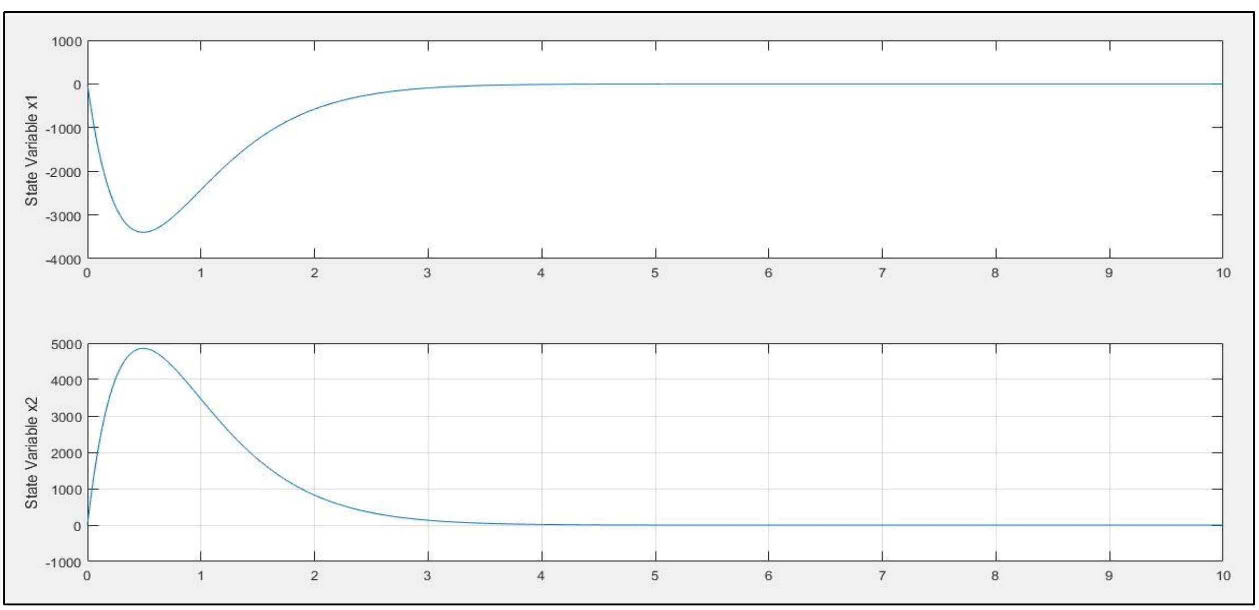

Fig. 18: Controller Response for Poles -2+0.3j, -2-0.3j.

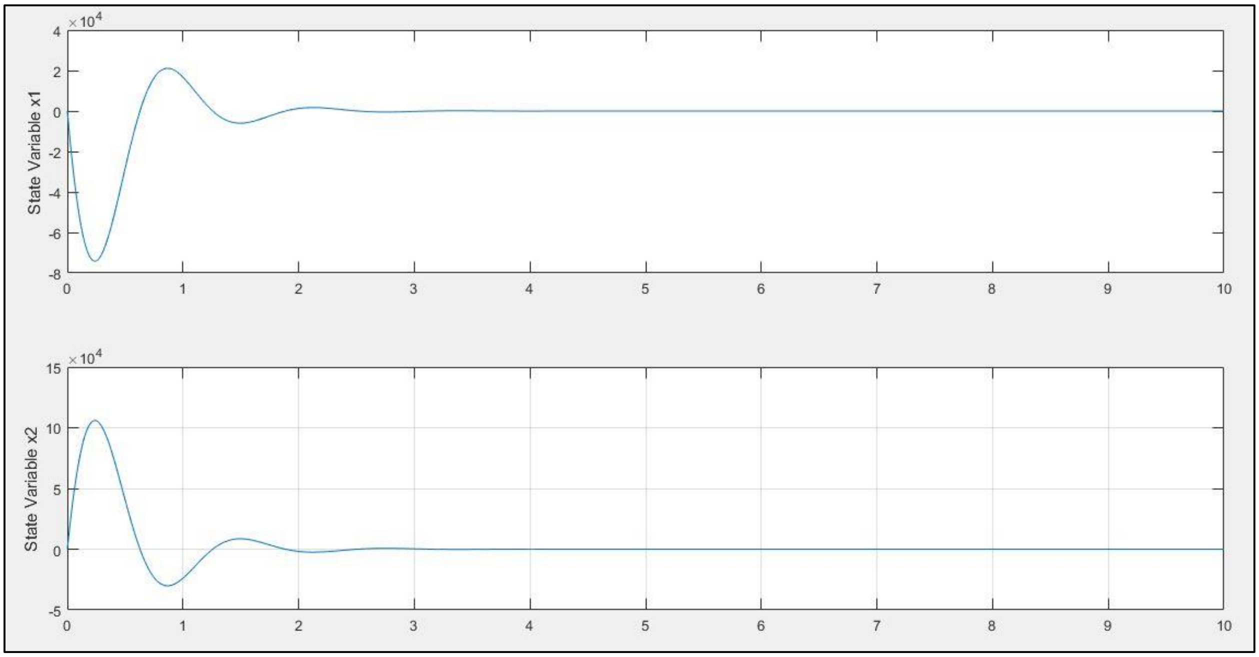

Fig. 19: Controller Response for Poles -2+5j, -2-5j.

Here it is seen that the controller response achieves steady state at approx. 3.2 sec for poles at -2+0.3j, -2-0.3j and at approx. 2.5 sec for poles at -2+5j, -2-5j. However, the latter response shows disturbances before achieving steady state. Hence, the poles at 2+0.3j, -2-0.3j are selected for designing the state space controller.

6.2 Analysis of State Space Observer

The initial response of the observer for different pole position are shown in Fig. 20 and Fig. 21.

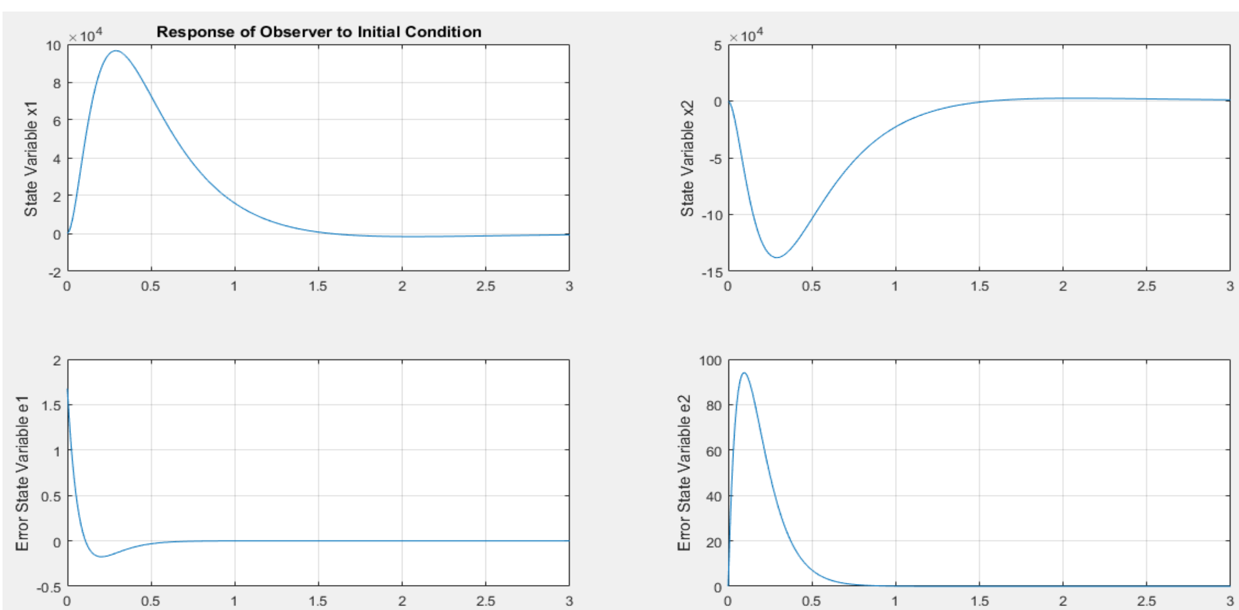

Fig. 20: Observer Response to Poles at -10, -11.

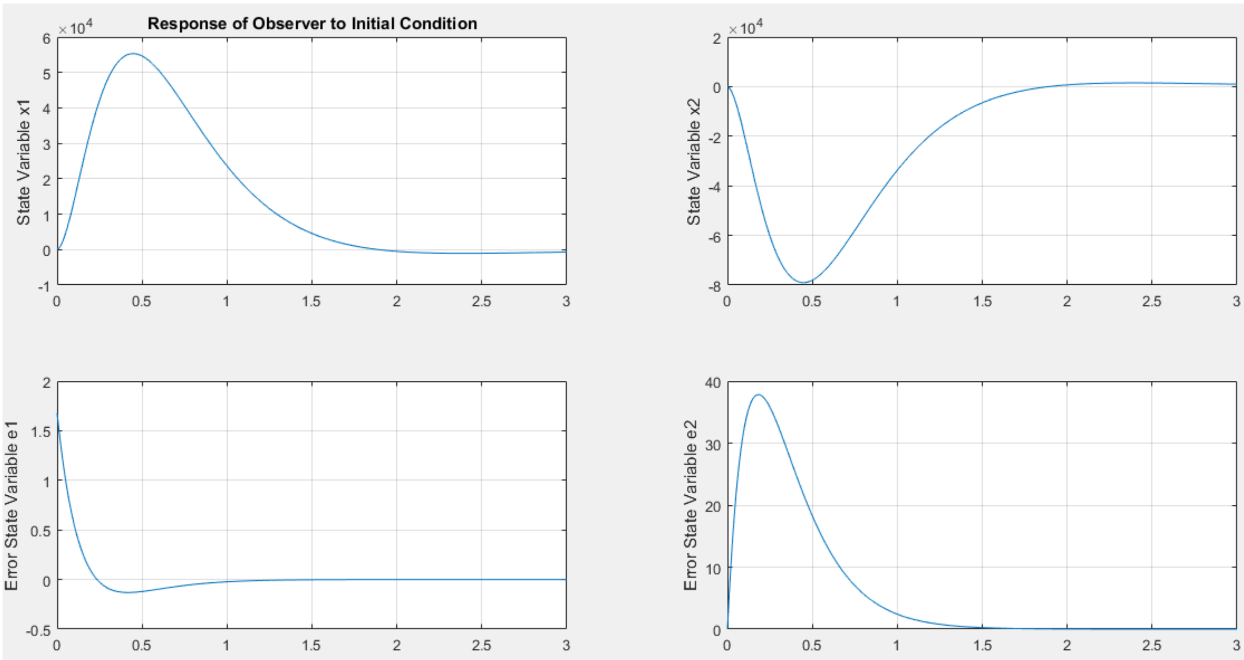

Fig. 21: Observer Response to Poles at -5, -6.

It can be seen from the graphs that the observer responses stabilize at approx. 1.5 sec for poles located at -10, 11 and they stabilize at approx. 2 sec when designed with located at -5, -6. Both the responses possess very less or no disturbances. Since, the time required by observer response to reach steady state must be several times faster than the controller response, hence, the observer is designed with poles located at -10, 11.

7. Conclusion

In this task, the non-linear behavior of a Bioreactor system is analyzed and linearization is performed on the state space representation. The system is found out to be unstable, controllable and observable. A state space controller is designed in MATLAB using Pole-Placement approach by selecting poles arbitrarily and their responses are analyzed. Additionally, a state space observer is also designed in such a way that the response time of the observer is at least 2 times faster than the controller. Simulink models of the system, the controller and the observer are drawn using MATLAB/Simulink and documented.

8. References

[1] Bioreactors, Science Direct: https://www.sciencedirect.com/topics/neuroscience/bioreactors

[2] Ogata, K. (2010). Modern Control Engineering 5th Edition. Pearson.