

# **Adaptive Control - Assignment 3**

## Stochastic Self Tuning Regulators

**Student:** Murtaza Asaadi, [mergenelos@gmail.com](mailto:mergenelos@gmail.com)

**Lecturer:** Prof. Baqeri, [peyman.bk@gmail.com](mailto:peyman.bk@gmail.com)

May, 2025

# Contents

<b>List of Figures</b>	<b>ii</b>
------------------------	-----------

<b>List of Program Codes</b>	<b>iii</b>
------------------------------	------------

<b>1 Questions 1 to 7</b>	<b>1</b>
1.1 Question 1 . . . . .	3
1.2 Question 2 . . . . .	6
1.3 Question 3 . . . . .	9
1.4 Question 4 . . . . .	12
1.5 Question 5 . . . . .	15
1.6 Question 6 . . . . .	17
1.7 Question 7 . . . . .	21

## List of Figures

1	Original system step response . . . . .	2
2	Discrete stable system step response . . . . .	2
3	Output and control of minimum variance controller . . . . .	4
4	Cummulative loss of Minimum variance controller . . . . .	5
5	Minimum variance controller with delay, output and control . . . . .	7
6	Minimum variance controller with delay, cummulative loss . . . . .	8
7	Output and control of indirect adaptive implementation of minimum variance controller . . . . .	10
8	Cummulative loss of indirect adaptive implementation of minimum variance controller . . . . .	11
9	output and control of direct adaptive implementation of minimum variance controller . . . . .	13
10	Cummulative loss of direct adaptive implementation of minimum variance controller . . . . .	14
11	Estimated parameters of direct adaptive implementation of minimum variance controller . . . . .	14
12	Output and control of moving average controller . . . . .	16
13	Cummulative loss of moving average controller . . . . .	17
14	Output and control of moving average controller with mirrored zeros . . . . .	19
15	Cummulative loss of moving average controller with mirrored zeros . . . . .	20
16	Output and control of direct adaptive implementation of moving average controller with mirrored zeros . . . . .	22
17	Cummulative loss of direct adaptive implementation of moving average controller . . . . .	23
18	Estimated parameters of direct adaptive implementation of moving average controller . . . . .	23

## List of Program Codes

1	Basic system implementation . . . . .	1
2	Minimum variance controller . . . . .	3
3	Minimum variance controller with delay . . . . .	6
4	Indirect adaptive implementation of minimum variance controller . . . . .	9
5	Direct adaptive implementation of minimum variance controller . . . . .	12
6	Moving average controller . . . . .	15
7	Moving average controller with mirrored zeros . . . . .	18
8	Poles & zeros with a mirrored zero . . . . .	21

# 1 Questions 1 to 7

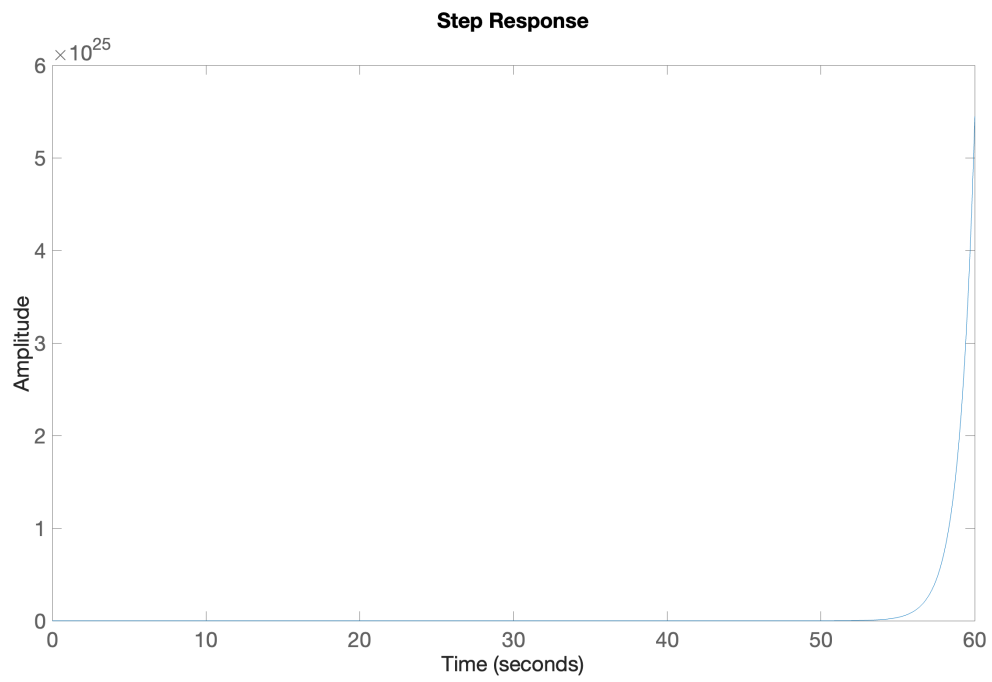
The Matlab implementation of the original system's transfer function is given in Code 1. Its step response, shown in Figure 1, indicates that the original system is unstable.

```
1  %% Setting up
2  clc, clear, close all
3
4  %% Required calculations
5  s = tf('s');
6  G = 3*(0.4*s+1)*(s+0.8)/((3*s+1)^2*(s+1));
7  % Settling time
8  step_info = stepinfo(G);
9  settling_time = step_info.SettlingTime;
10 % Sample time
11 Ts = floor(settling_time)*0.1;
12
13 %% colored noise coefficients
14 C = [1,0.6,0.4];
15
16 %% Original transfer function
17 G_s = 3*(0.4*s+1)*(s+0.8)/((3*s+1)^2*(s-1));
18
19 %% Discrete transfer function
20 G_discrete = c2d(G_s,Ts,'zoh');
21 [B,A] = tfdata(G_discrete,'v');
22 B(1) = [];
```

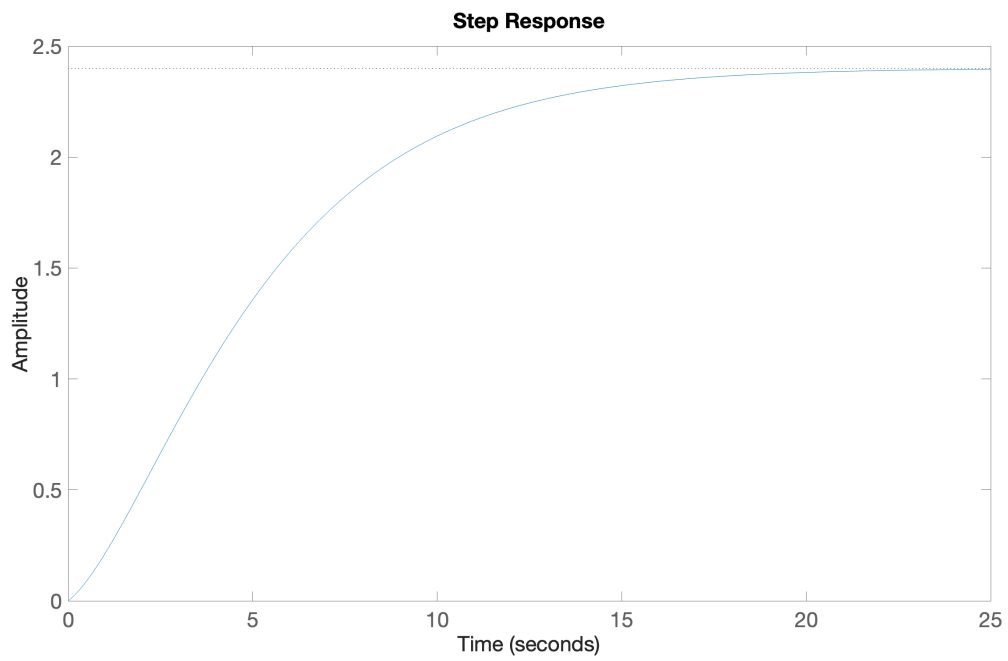
**Code 1:** Basic system implementation

To enable the calculation of the system's settling time, the unstable pole was mirrored and the system was discretized, as implemented in the code shown in Code 1. The step response of the resulting, modified system is presented in Figure 2. The system rise time is 9.81 and settling time is 16.71 seconds.

The code for this section is available at [assignment3/SSTR/SSTR\\_0.m](#).



**Figure 1:** Original system step response



**Figure 2:** Discrete stable system step response

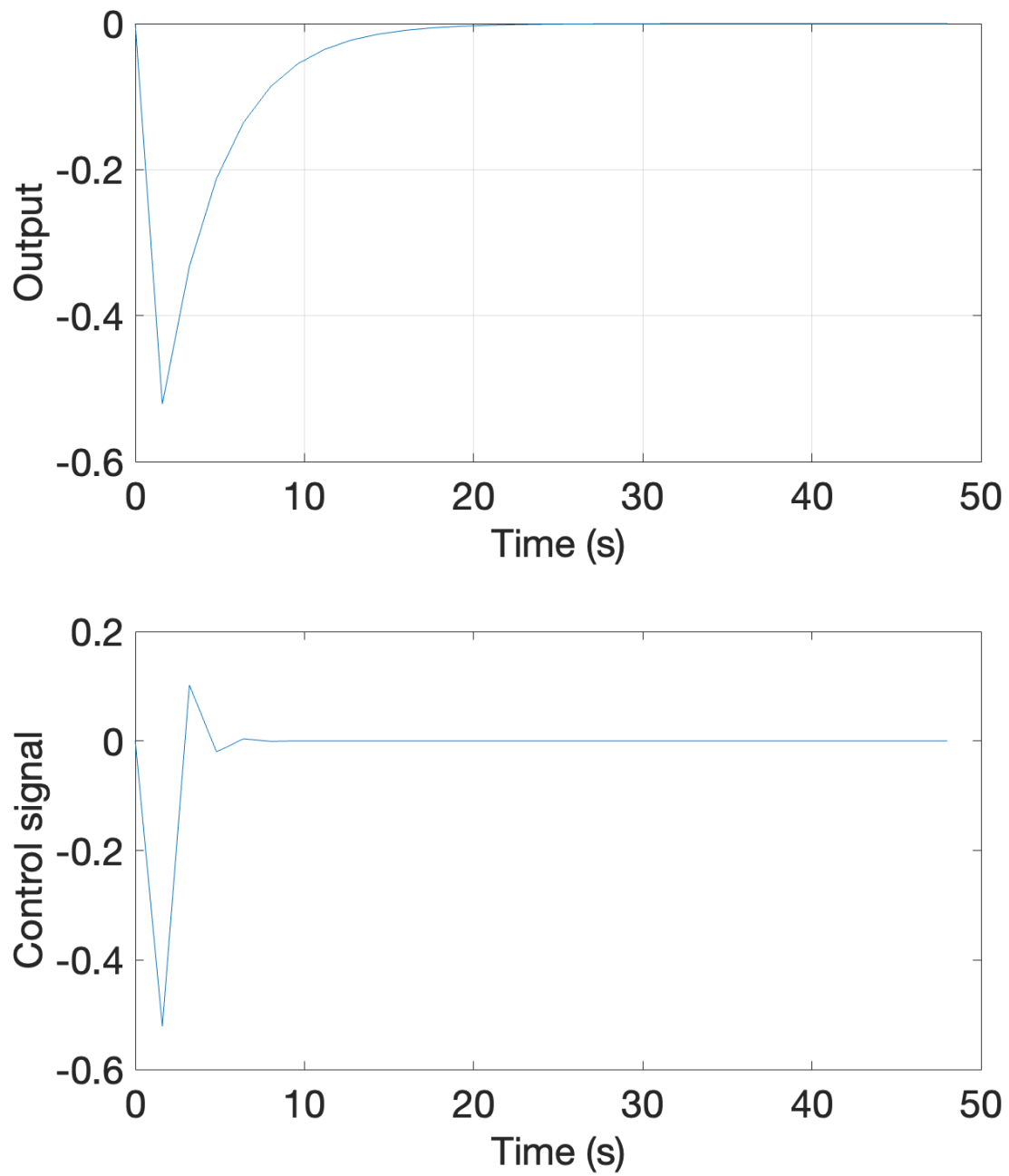
## 1.1 Question 1

First, we implement the open-loop system. To close the loop, we apply a minimum variance controller using Code 2. As shown in Figure 3, the system remains stable and, when actuated, returns to its equilibrium point at 0. As shown in Figure 4, cumulative loss of the system is bounded.

```
1  run('SSTR_0.m');
2
3  %% Diophantine equation
4  [F,G] = diophantine(A,C,1);
5
6  %% Open loop system
7  G_ol = minreal(G_discrete*tf(A,conv(B,F),Ts));
8
9  %% Closed loop system
10 G_cl = feedback(G_ol,1);
```

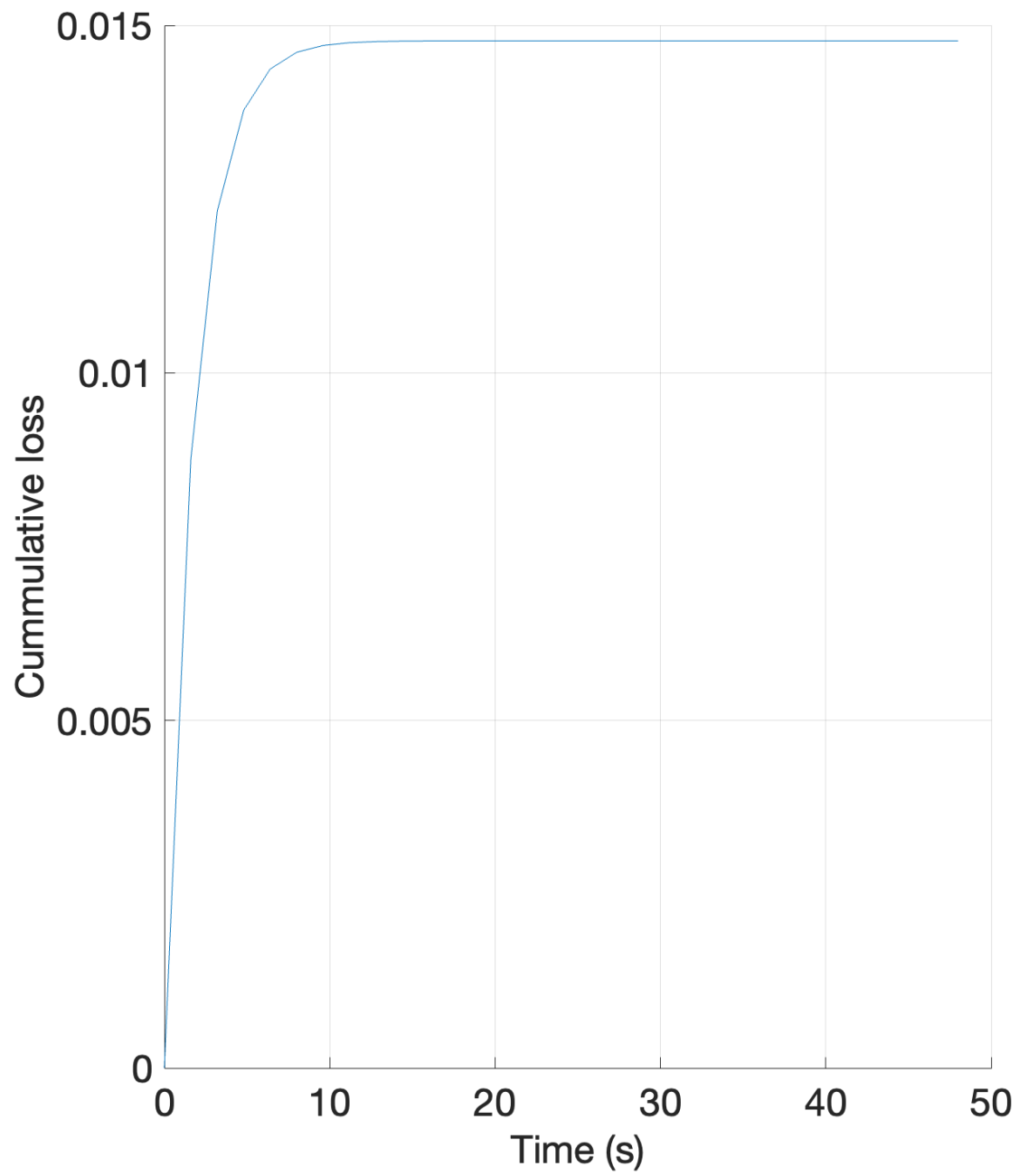
**Code 2:** Minimum variance controller

The code for this section is available at [assignment3/SSTR/SSTR\\_1.m](#).



**Figure 3:** Output and control of minimum variance controller





**Figure 4:** Cumulative loss of Minimum variance controller

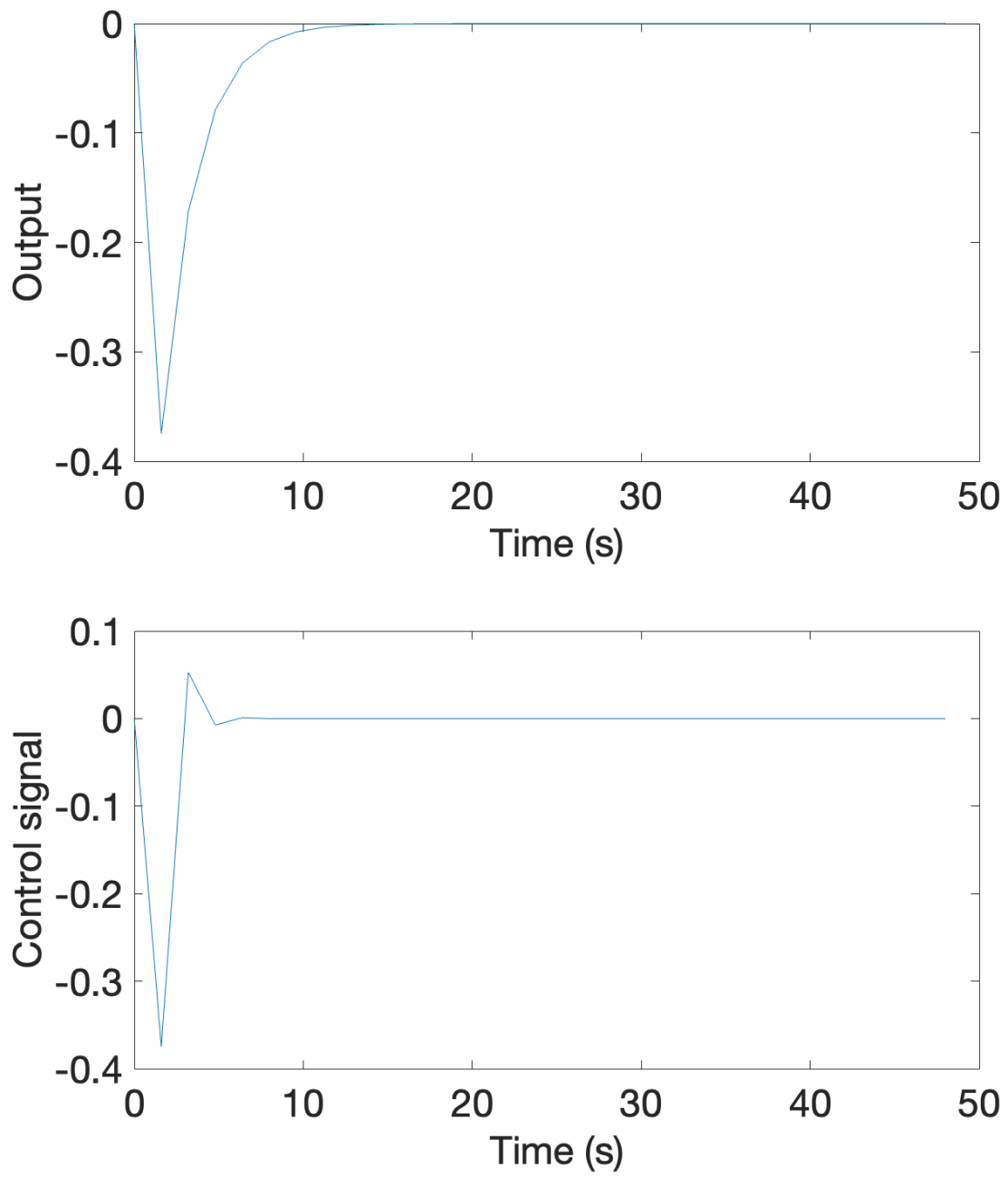
## 1.2 Question 2

The required delay in the transfer function is implemented as shown in Code 3. Rest of the code remains as presented in Code 2. As shown in Figure 5, the implemented system is stable and returns to its equilibrium point at 0. And in Figure 6, cumulative loss of the system is bounded.

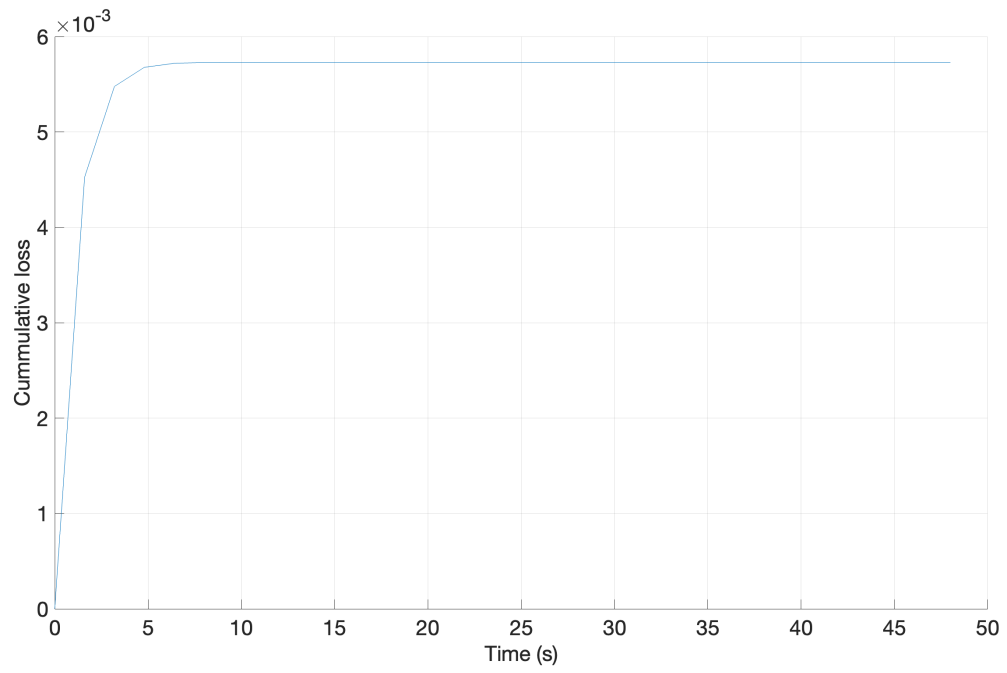
```
10  %% Transfer function
11  G_s =
    ↪  3*(0.4*s+1)*(s+0.8)/((3*s+1)^2*(s-1))*tf(1,[1,0,0]);
```

**Code 3:** Minimum variance controller with delay

The code for this section is available at [assignment3/SSTR/SSTR\\_2.m](#).



**Figure 5:** Minimum variance controller with delay, output and control



**Figure 6:** Minimum variance controller with delay, cummulative loss

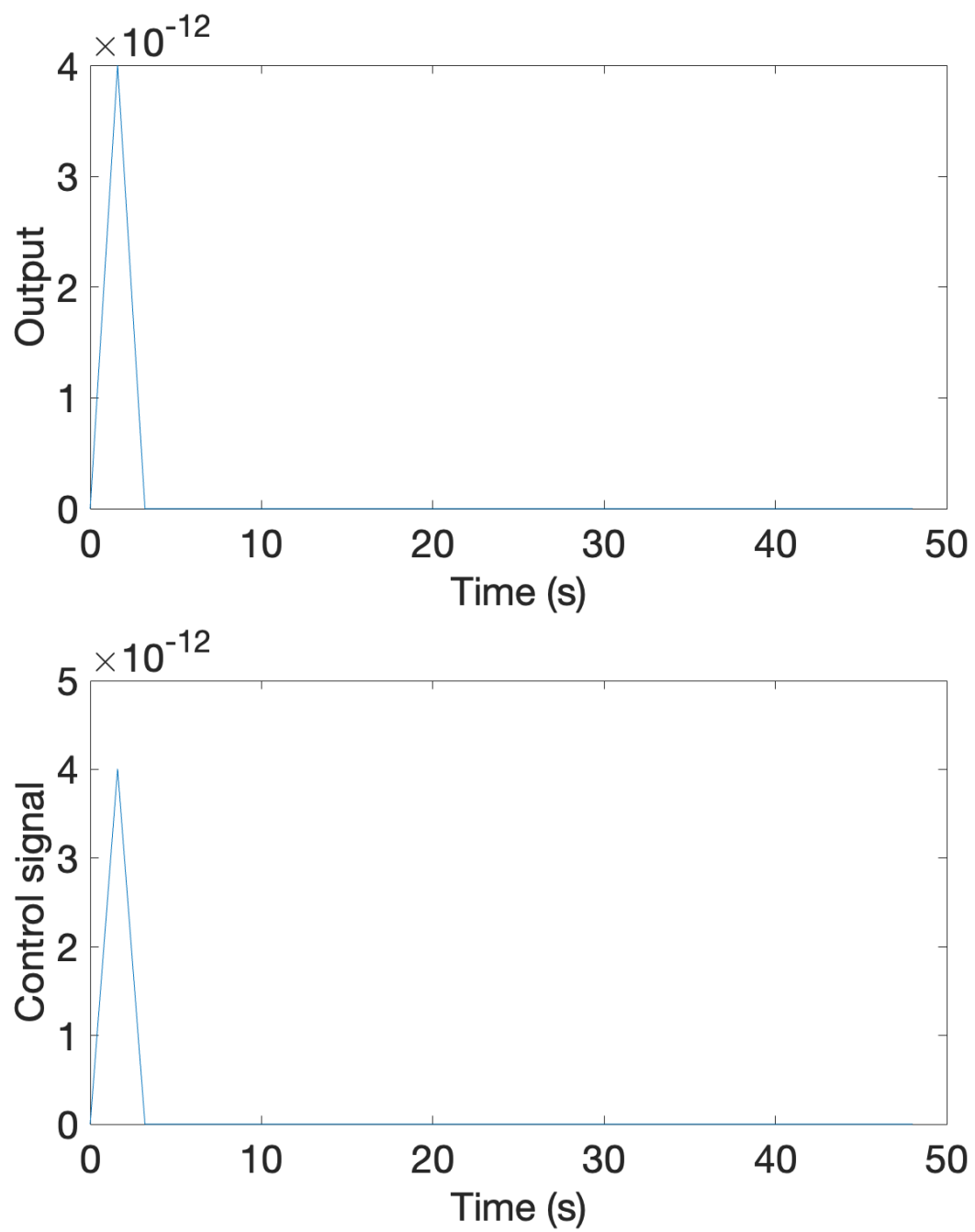
### 1.3 Question 3

We implement the required indirect adaptive controller as shown in Code 4. As shown in Figure 7, the system remains stable and in Figure 6, cumulative loss of the system is bounded.

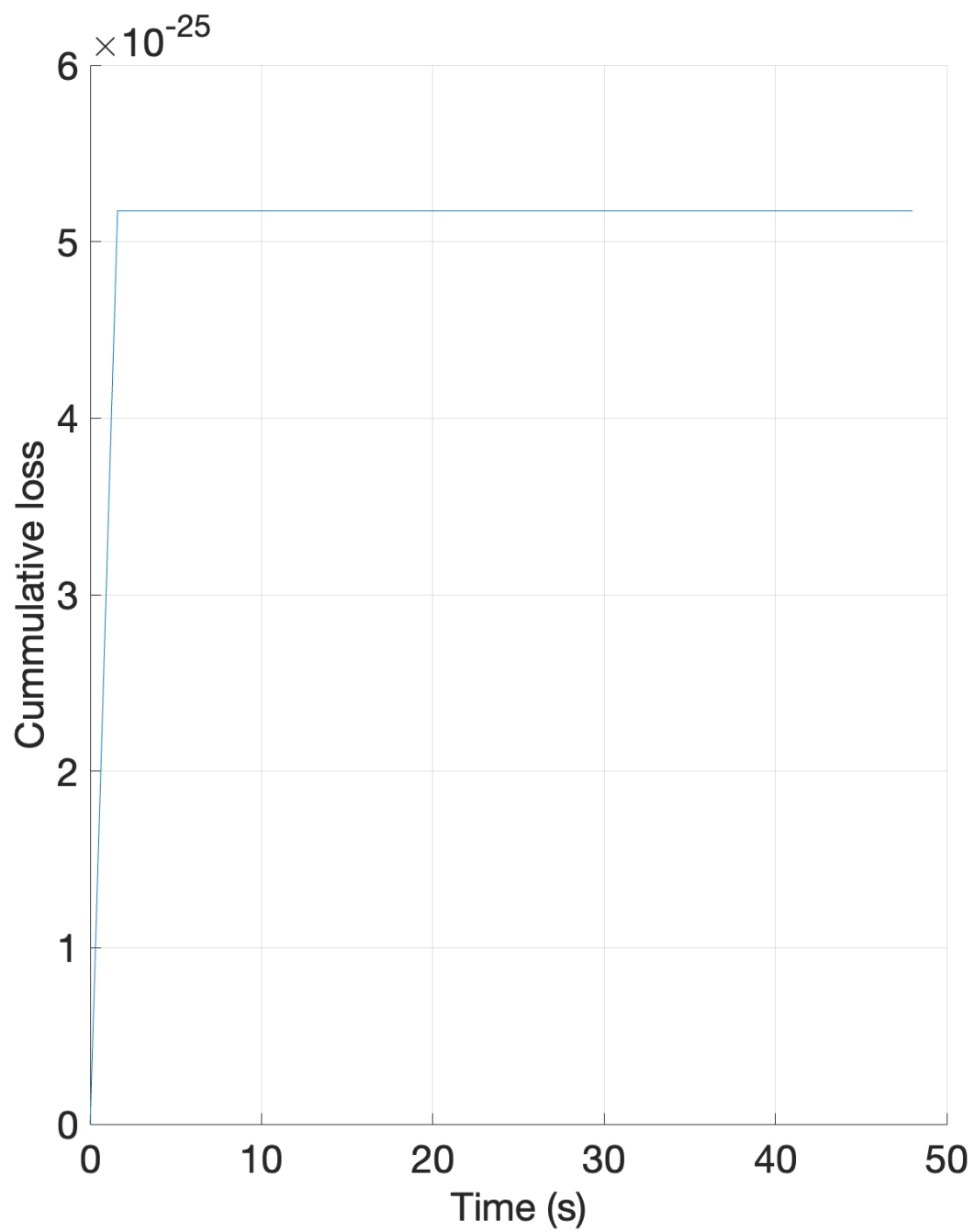
```
1  %% Identification parameters
2  na = 2; nb = 2;
3  theta = zeros(na+nb,1);
4  P = 1000*eye(na+nb);
5  lambda = 0.98;
6  %% Input and output data
7  N = 50;
8  u = randn(1,N);
9  y = zeros(1,N);
10 y_est = zeros(1,N);
11 for k = max(na,nb)+1:N
12 y(k) = -A(2)*y(k-1) - A(3)*y(k-2) + ...
13 B(2)*u(k-1) + B(3)*u(k-2);
14 end
15 %% RLS
16 for k = max(na,nb)+1:N
17 phi = [-y(k-1); -y(k-2); u(k-1); u(k-2)];
18 y_hat = theta'*phi;
19 e = y(k) - y_hat;
20 K = (P*phi)/(lambda + phi'*P*phi);
21 theta = theta + K*e;
22 P = (P - K*phi'*P)/lambda;
23
24 y_est(k) = y_hat;
25 THETA(:,k) = theta;
26 end
27
28 %% Recover A and B
29 A_id = [1; theta(1:na)];
30 B_id = theta(na+1:end);
31
32 %% Adaptive controller
33 [F,G] = diophantine(A_id, C, 1);
34
35 %% Open-loop and closed-loop systems
36 G_z_hat = tf(B_id', A_id', Ts);
37 G_ol = minreal(G_z_hat * tf(A_id', conv(B_id', F), Ts));
38 G_cl = feedback(G_ol, 1);
```

**Code 4:** Indirect adaptive implementation of minimum variance controller

The code for this section is available at [assignment3/SSTR/SSTR\\_3.m](#).



**Figure 7:** Output and control of indirect adaptive implementation of minimum variance controller



**Figure 8:** Cumulative loss of indirect adaptive implementation of minimum variance controller

## 1.4 Question 4

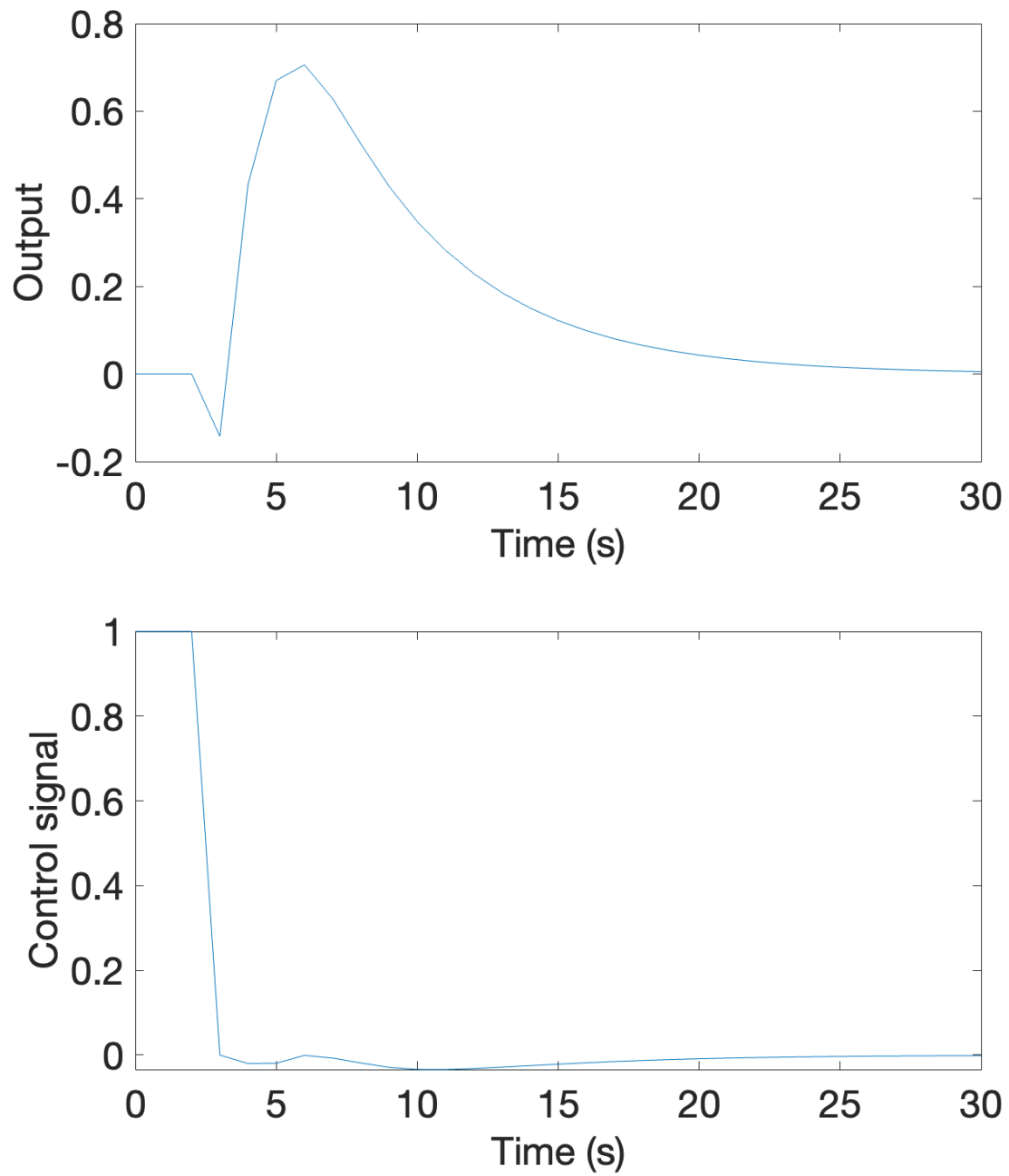
Direct adaptive controller codebase is presented in Code 5. As shown in Figure 9, the system is stable and in Figure 10, cumulative loss of the system is bounded. Figure 11 presents the parameters estimation plot of direct adaptive controller.

```
1  run('SSTR_0.m');
2
3  %% Solve Diophantine equation
4  [F,G] = diophantine(A,C,1);
5
6  %% Closed loop system
7  G_ol = minreal(G_discrete*tf(A,conv(B,F),Ts));
8
9  %% closed loop system
10 G_cl = feedback(G_ol,1);
11 [B_true,A_true] = tfdata(G_cl,'v');
12
13 %% Simulation settings
14 na = length(A_true)-1; nb = length(B_true)-1;
15 N = 31;
16 theta = zeros(nb + na, 1);
17 gamma = 0.01;
18
19 u = ones(1,N);
20 y = zeros(1,N);
21 y_ref = ones(1,N);
22 phi = zeros(nb + na, 1);
23
24 %% Simulation loop
25 for k = 4:N
26 e = y_ref(k) - y(k);
27 phi = [-y(k-1); -y(k-2); -y(k-3); u(k-1); u(k-2);
28        ↪ u(k-3)];
29 u(k) = theta' * phi;
30 y(k) = -A_true(2)*y(k-1) - A_true(3)*y(k-2) -
31        ↪ A_true(4)*y(k-3) + ...
32 B_true(2)*u(k-1) + B_true(3)*u(k-2) + B_true(4)*u(k-3);
33
34 theta = theta - gamma * e * phi;
35 THETA(:,k) = theta;
36 end
```

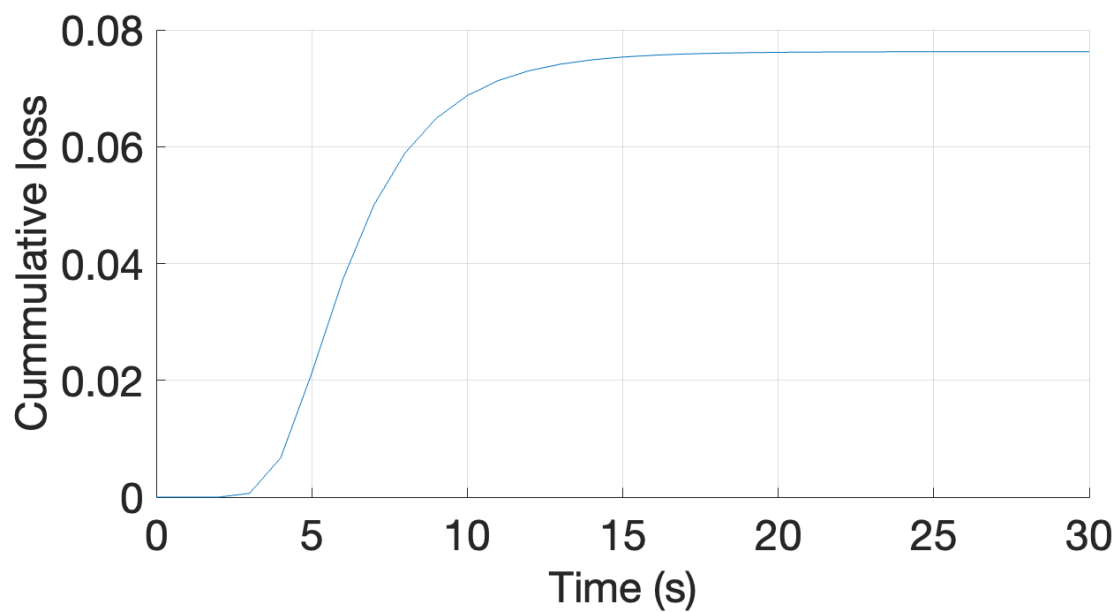
**Code 5:** Direct adaptive implementation of minimum variance controller

The code for this section is available at [assignment3/SSTR/SSTR\\_4.m](#).

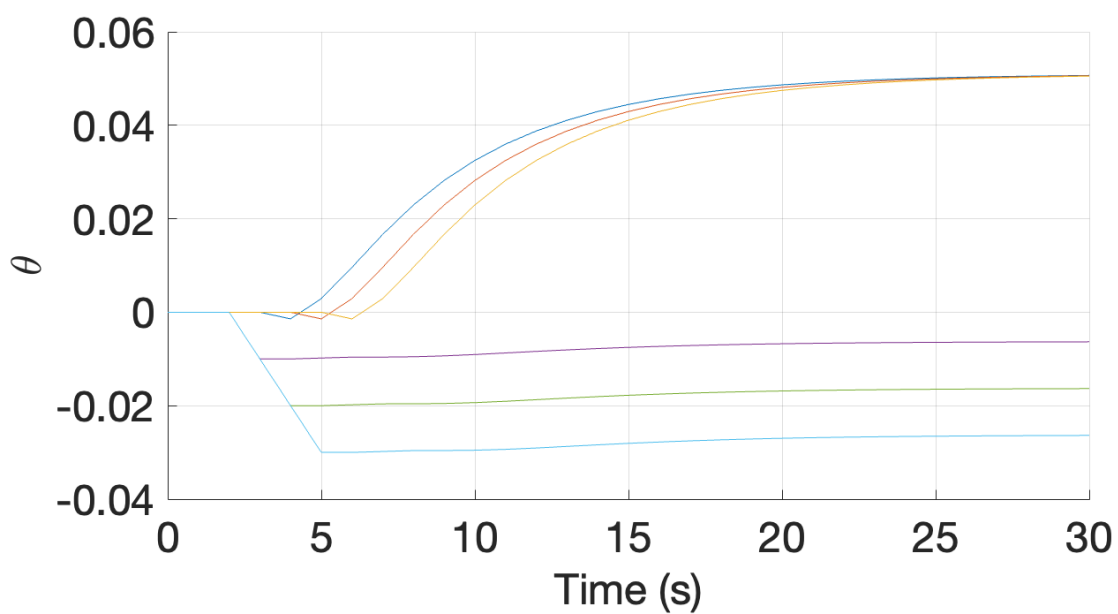




**Figure 9:** output and control of direct adaptive implementation of minimum variance controller



**Figure 10:** Cumulative loss of direct adaptive implementation of minimum variance controller



**Figure 11:** Estimated parameters of direct adaptive implementation of minimum variance controller

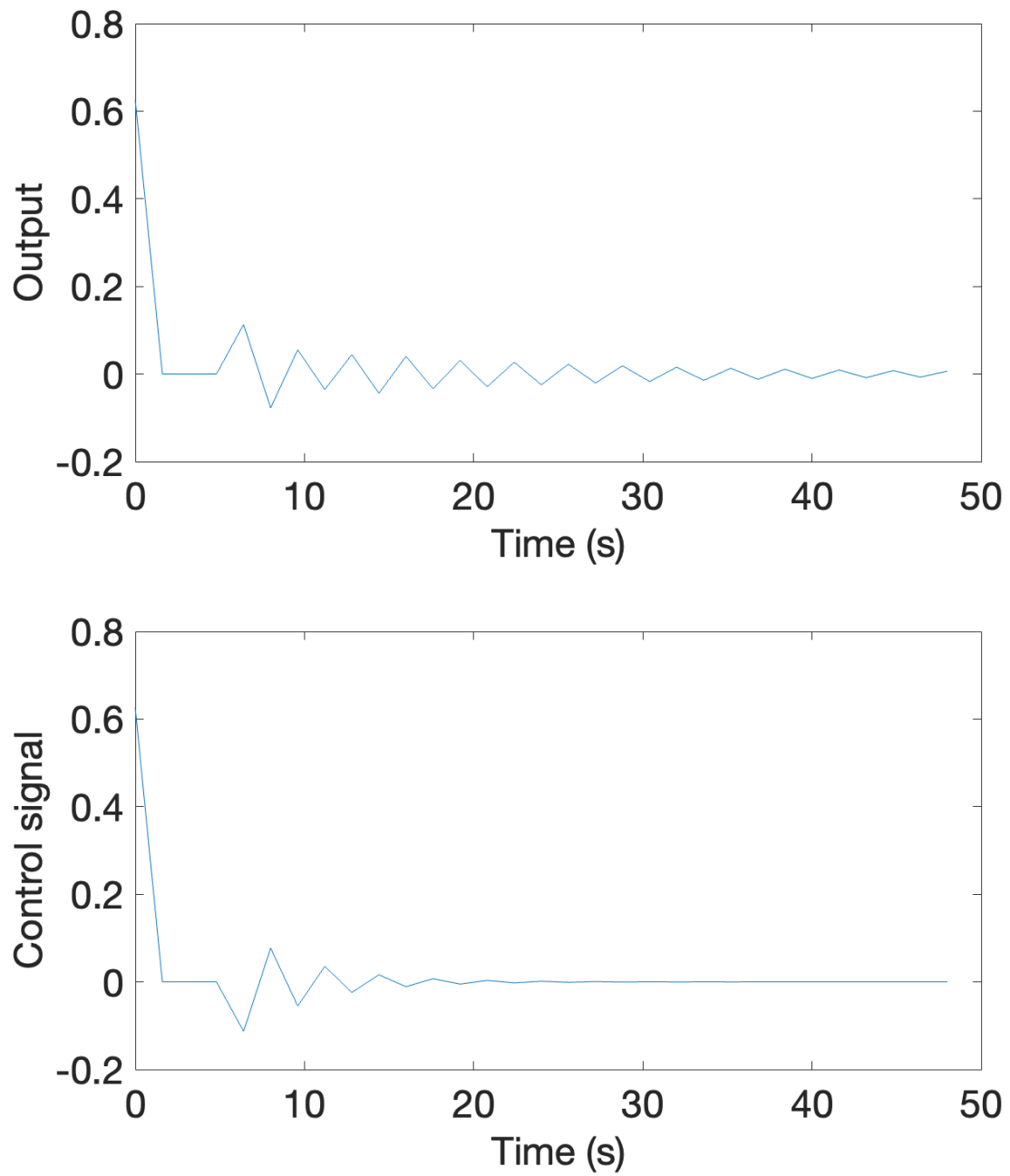
## 1.5 Question 5

We implement the open-loop system. To close the loop, we apply a moving average controller using Code 6. As shown in Figure 12, the system remains stable and, when actuated, returns to its equilibrium point at 0. As shown in Figure 13, cumulative loss of the system is bounded.

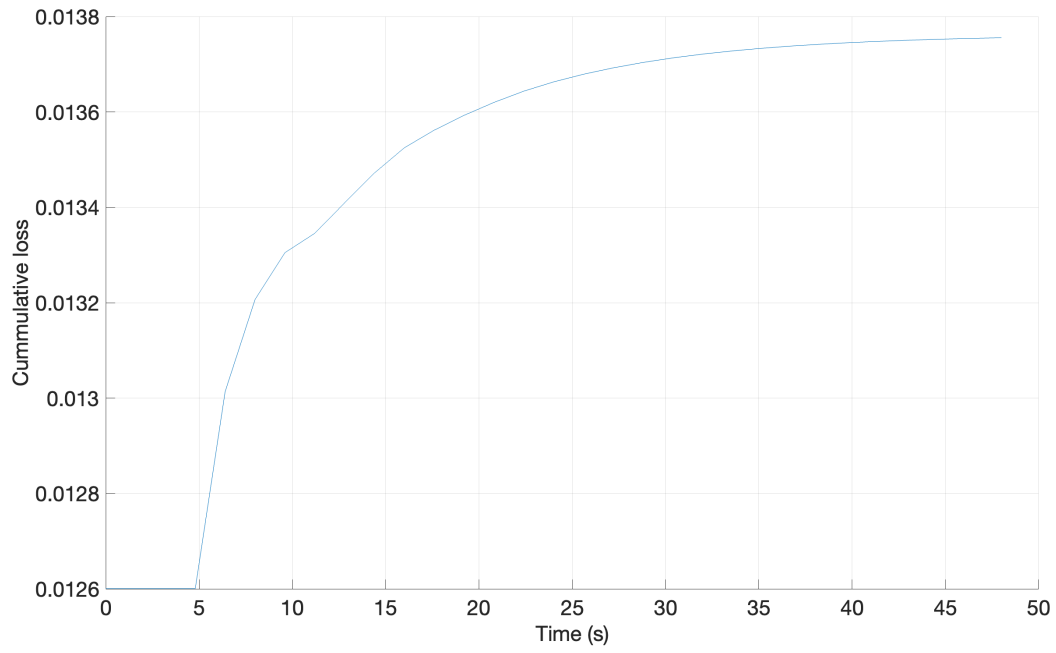
```
1  run('SSTR_0.m');
2
3  %% Solve Diophantine equation for d = 2
4  d = 2; % d0=1
5  [F, G] = diophantine(A, C, d);
6
7  %% Define the MA controller
8  MA_controller = tf(conv(F,A), 1, Ts);
9
10 %% Define the open loop system
11 G_ol = minreal(G_discrete * MA_controller);
12
13 %% Closed-loop system
14 G_cl = feedback(G_ol, 1);
```

**Code 6:** Moving average controller

The code for this section is available at [assignment3/SSTR/SSTR\\_5.m](#).



**Figure 12:** Output and control of moving average controller



**Figure 13:** Cumulative loss of moving average controller

## 1.6 Question 6

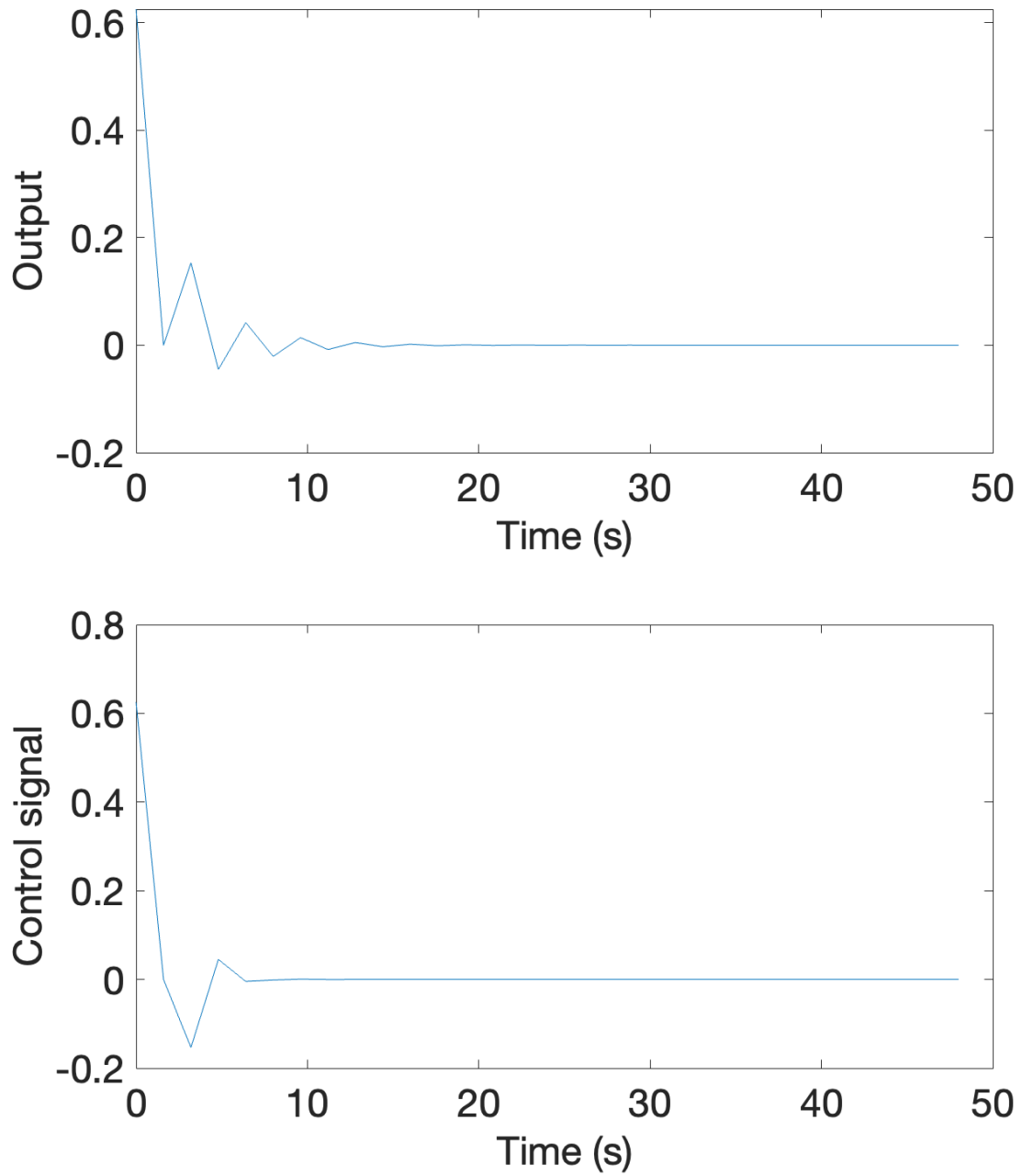
We mirror zeros of the transfer function then we apply a moving average controller using Code 7. As shown in Figure 14, the system remains stable and, when actuated, returns to

its equilibrium point at 0. As shown in Figure 15, cumulative loss of the system is bounded.

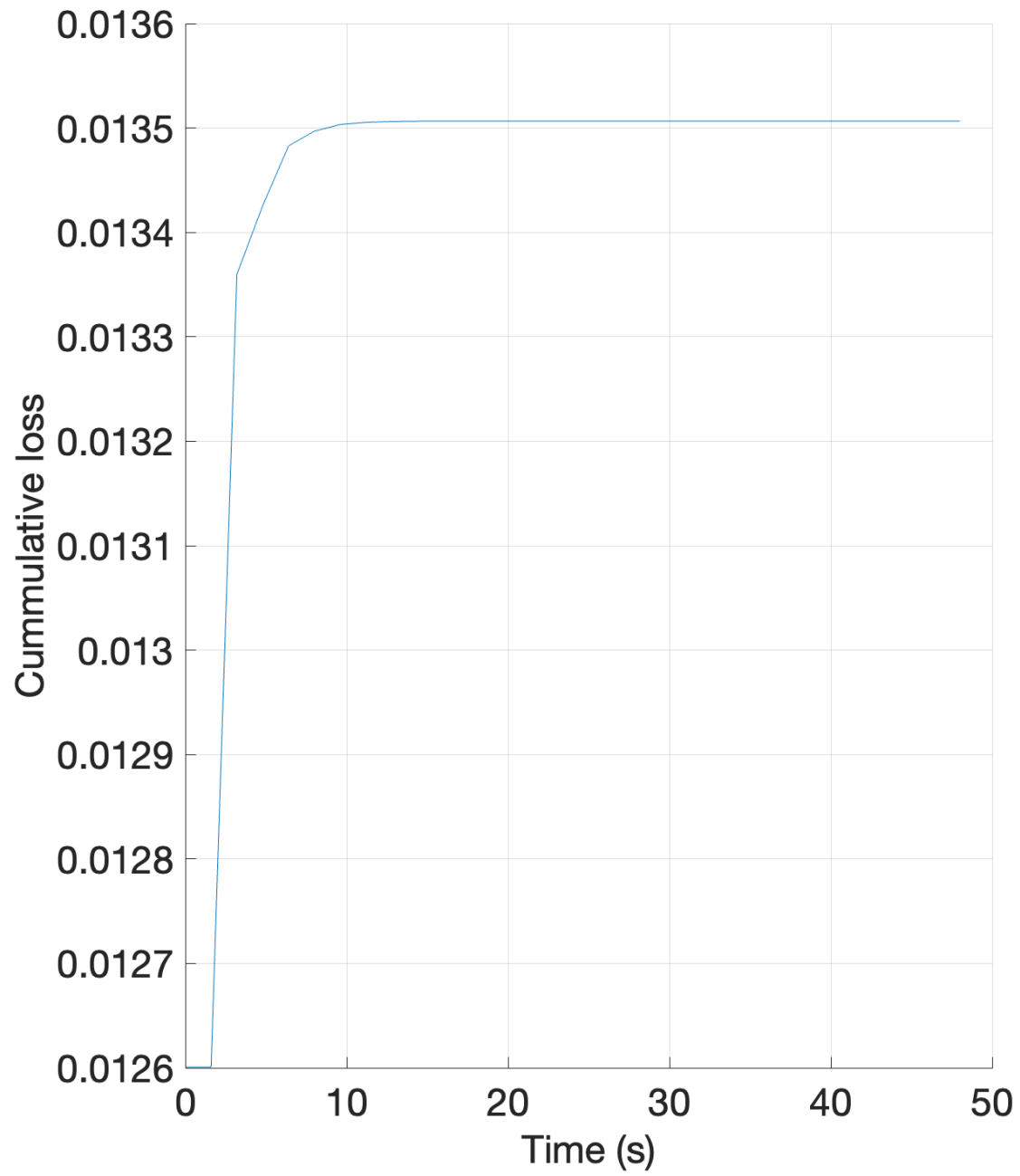
```
1  %% Original unstable system
2  G_s = 3*(0.4*s-1)*(s-0.8)/((3*s+1)^2*(s-1));
3
4  %% Discretize the system
5  G_z = c2d(G_s, Ts, 'zoh');
6  [B, A] = tfdata(G_z, 'v');
7  B(1) = [];
8
9  %% Solve Diophantine equation for d = 2
10 d = 2; % d0=1
11 [F, G] = diophantine(A, C, d);
12
13 %% Define the MA controller
14 MA_controller = tf(conv(F,A), B, Ts);
15
16 %% Define the open loop system
17 G_ol = minreal(G_z * MA_controller);
18
19 %% Closed-loop system
20 G_cl = feedback(G_ol, 1);
21
```

**Code 7:** Moving average controller with mirrored zeros

The code for this section is available at [assignment3/SSTR/SSTR\\_6.m](#).



**Figure 14:** Output and control of moving average controller with mirrored zeros



**Figure 15:** Cumulative loss of moving average controller with mirrored zeros

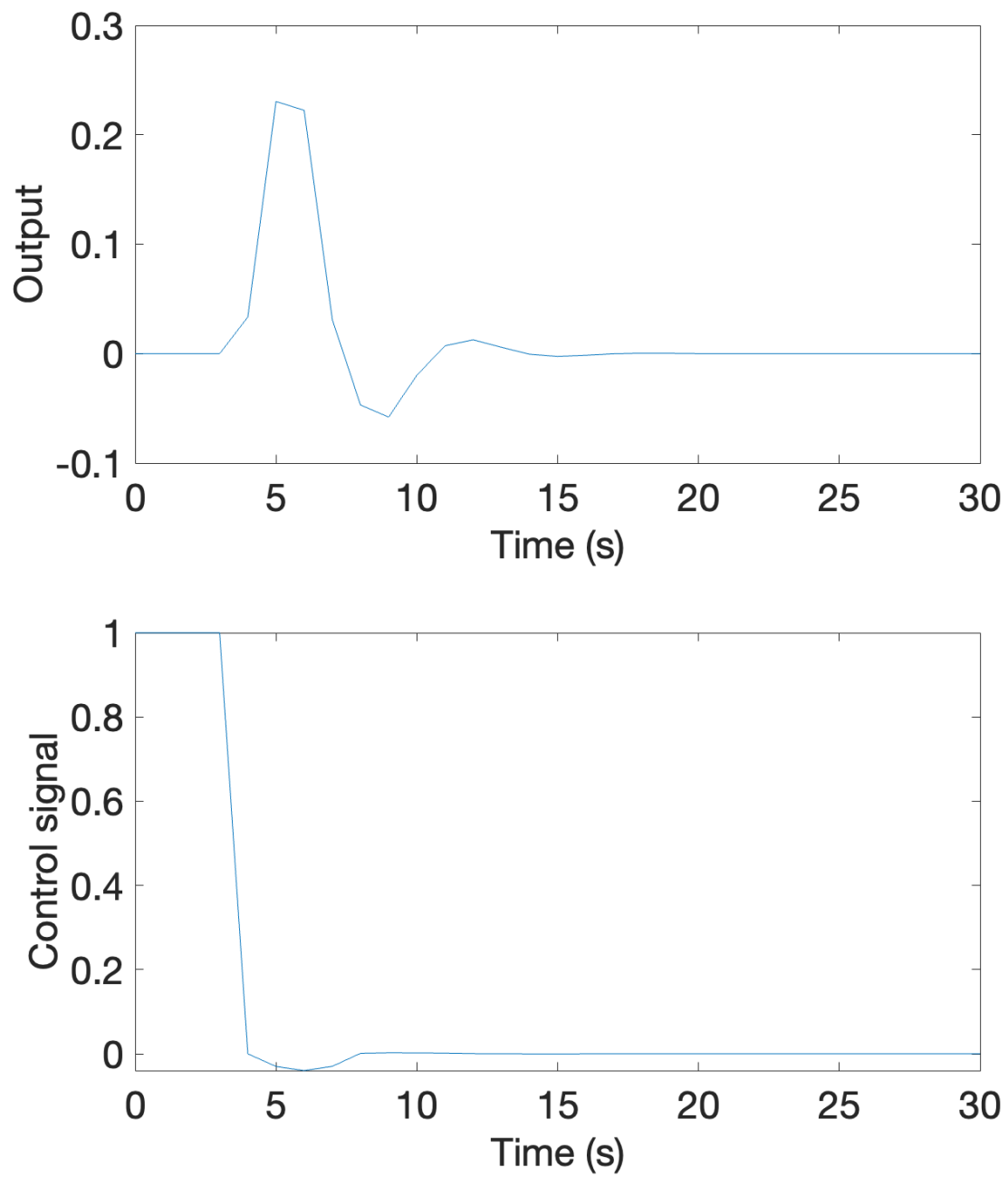


## 1.7 Question 7

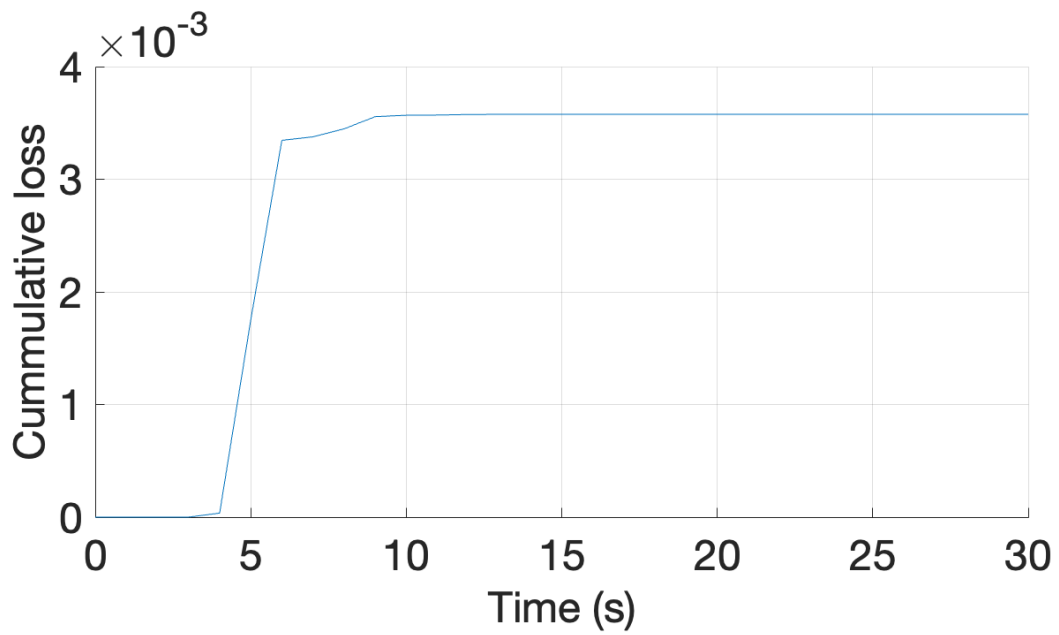
?? implements the direct adaptive moving average controller. As shown in Figure 16, the system remains stable and, when actuated, returns to its equilibrium point at 0. As shown in Figure 17, cumulative loss of the system is bounded. Estimated parameters are presented in Figure 18. The code for this section is available at [assignment3/SSTR/SSTR\\_7.m](#).

```
1  %% Solve Diophantine equation
2  d = 2; % d0=1
3  [F, G] = diophantine(A, C, d);
4
5  %% Define the MA controller
6  MA_controller = tf(conv(F,A), 1, Ts);
7
8  %% Open loop
9  G_ol = minreal(G_discrete * MA_controller);
10
11 %% Closed loop
12 G_cl = feedback(G_ol, 1);
13 A_true = [-2.5373    -0.1906    0.0497    0.1744
14           ↪ 1.0341];
15 B_true = [-2.5373    -0.1906    0.0497    0.1744
16           ↪ 0.0341];
17
18 %% Simulation settings
19 na = length(A_true)-1; nb = length(B_true)-1;
20 N = 31;
21 theta = zeros(nb + na, 1);
22 gamma = 0.01;
23 u = ones(1,N);
24 y = zeros(1,N);
25 y_ref = ones(1,N);
26 phi = zeros(nb + na, 1);
27
28 for k = 5:N
29     e = y_ref(k) - y(k);
30     phi = [-y(k-1); -y(k-2); -y(k-3); -y(k-4); ...
31            u(k-1); u(k-2); u(k-3); u(k-4)];
32     u(k) = theta' * phi;
33     y(k) = -A_true(2)*y(k-1) - A_true(3)*y(k-2) -
34            ↪ A_true(4)*y(k-3) + ...
35            B_true(2)*u(k-1) + B_true(3)*u(k-2) + B_true(4)*u(k-3);
36
37     theta = theta - gamma * e * phi;
38     THETA(:,k) = theta;
39 end
```

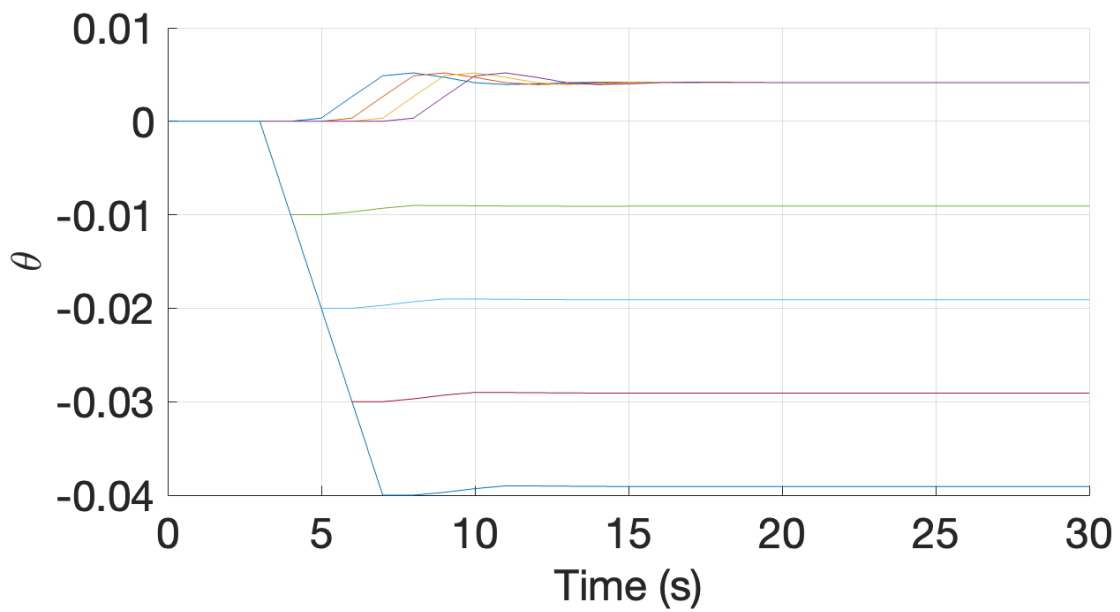
**Code 8:** Poles & zeros with a mirrored zero



**Figure 16:** Output and control of direct adaptive implementation of moving average controller with mirrored zeros



**Figure 17:** Cumulative loss of direct adaptive implementation of moving average controller



**Figure 18:** Estimated parameters of direct adaptive implementation of moving average controller