CH E 152B Homework 4

Connor Hughes

April 26, 2021

Exercise 9: Transfer function, time series, and state space models

This exercise, reproduced below, is Exercise 1.10 in Rawlings, Mayne, and Diehl (2020).

Define a state vector and realize the following models as state space models **by hand**. One should do a few by hand to understand what the Octave or `MATLAB` calls are doing. Answer the following questions. What is the connection between the poles of G and the state space description? For what kinds of $G(s)$ does one obtain a nonzero D matrix? What is the order and gain of these systems? Is there a connection between order and the numbers of inputs and outputs?

(a)

$$G(s) = \frac{1}{2s+1}$$

(b)

$$G(s) = \frac{1}{(2s+1)(3s+1)}$$

(c)

$$G(s) = \frac{2s+1}{3s+1}$$

(d)

$$y(k+1) = y(k) + 2u(k)$$

(e)

$$y(k+1) = a_1 y(k) + a_2 y(k-1) + b_1 u(k) + b_2 u(k-1)$$

Solution:

(a)

Here, we can see that $G(s) = \frac{1}{2s+1}$ and $y(s) = G(s)u(s) \implies 2sy(s) + y(s) = u(s)$. So, we can take the inverse Laplace transform of this expression to get $2\frac{dy}{dt} + y(t) = u(t)$. Then, we can introduce the state variable x (which is our entire state vector in this 1-dimensional case) and express our system in state space form as follows:

$$\dot{x} = -\frac{1}{2}x + \frac{1}{2}u$$
$$y = x$$

where we can see that our system matrices are given by $A = -\frac{1}{2}$, $B = \frac{1}{2}$, $C = 1$, and $D = 0$.

(b)

Similarly, here we have

$$G(s) = \frac{1}{(2s+1)(3s+1)} = \frac{1}{(6s^2+5s+1)} \text{ and } y(s) = G(s)u(s) \implies 6s^2y(s) + 5sy(s) + y(s) = u(s).$$

So, we can take the inverse Laplace transform of this expression to get $6\frac{d^2y}{dt^2} + 5\frac{dy}{dt} + y(t) = u(t)$. Now, we will define the state vector

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} y \\ \dot{y} \end{bmatrix}$$

and we can then express our system in state space form as follows

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{1}{6} & -\frac{5}{6} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{6} \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

where our system matrices are given by $A = \begin{bmatrix} 0 & 1 \\ -\frac{1}{6} & -\frac{5}{6} \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ \frac{1}{6} \end{bmatrix}$, $C = \begin{bmatrix} 1 & 0 \end{bmatrix}$, and $D = 0$.

(c)

Here, we must begin by converting the given transfer function to the sum of a strictly proper transfer function and a constant, as follows

$$G(s) = \frac{2s+1}{3s+1} = \frac{\frac{1}{3}}{3s+1} + \frac{2}{3} = \frac{1}{9s+3} + \frac{2}{3}$$

Since we have the sum of two transfer functions, we can take the inverse Laplace transform and find their state space representations separately then recombine them after, as follows

$$y = y_1 + y_2$$

$$y_1(s) = \frac{\frac{1}{3}}{3s+1}u(s) \implies 9\frac{dy_1}{dt} + 3y_1(t) = u(t)$$

$$y_2(s) = \frac{2}{3}u(s) \implies y_2(t) = \frac{2}{3}u(t)$$

which, by introducing the state variable x, we can express as

$$\dot{x} = -\frac{1}{3}x + \frac{1}{9}u$$

$$y_1 = x$$

and

$$y_2 = \frac{2}{3}u$$

so we can see

$$\dot{x} = -\frac{1}{3}x + \frac{1}{9}u$$

$$y = x + \frac{2}{3}u$$

and thus we have found a state space representation for the system, with system matrices $A = -\frac{1}{3}$, $B = \frac{1}{9}$, $C = 1$, $D = \frac{2}{3}$.

(d)

Here, we are given a time series model to begin with, so we may go directly to a discrete time state space representation without the inverse Laplace transform. We shall introduce the state variable $x(k) = y(k)$, which gives

$$x(k+1) = x(k) + 2u(k)$$
$$y(k) = x(k)$$

where our system is in state space form with system matrices given by $A_d = 1$, $B_d = 2$, $C_d = 1$, $D_d = 0$.

(e)

Once again, we are given a time series model, so we may go directly to a discrete time state space representation. We shall introduce the state vector

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} y(k-1) \\ y(k) \end{bmatrix}$$

which gives

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ a_2 & a_1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ b_2 & b_1 \end{bmatrix} \begin{bmatrix} u(k-1) \\ u(k) \end{bmatrix}$$
$$y(k) = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}$$

where our system is in state space form with system matrices given by $A_d = \begin{bmatrix} 0 & 1 \\ a_2 & a_1 \end{bmatrix}$, $B_d = \begin{bmatrix} 0 & 0 \\ b_2 & b_1 \end{bmatrix}$, $C_d = \begin{bmatrix} 0 & 1 \end{bmatrix}$, $D_d = 0$.

We can make some observations based on the previous five calculations, regarding the conversion between transfer functions and state space representations. We notice that the poles of the transfer function G(s) are the eigenvalues of the A matrix. When G(s) has the same order polynomial in the numerator as the denominator, we get a nonzero D matrix which is equal to $\lim_{s\to\infty} G(s)$. We can see that the order of the system is equal to the order of the denominator of the transfer function, which corresponds to the number of elements needed for our state vector. The steady state gain of the system is given by $G(0)$ for a transfer function, and can be found from the state space representation by computing $G(s) = C(sI - A)^{-1}B + D$ and plugging in $s = 0$. There is not a connection between the order of a system and its number of inputs and outputs. It is common to have the number of inputs and outputs be equal to or fewer than the number of states, but this need not be the case. We could have arbitrarily many inputs and outputs for even a first order system with a single state variable, or arbitrarily many states for a high-order system with a single input and single output.

Exercise 10: Plant Simulation

Consider some heating process in Figure 1. Assume we have identified a first-order transfer function connecting the steam valve position to the measured temperature

$$y(s) = \frac{k}{\tau s + 1} u(s)$$

in which $k = 5^\circ C$, $\tau = 1$ hr. Now we want to check how well our model fits the actual plant operation, so we examine the recorded valve positions for the last 10 hours of plant operation. These data are plotted in Figure 2 and available on the class website in the homework folder. See file `heating.dat`.

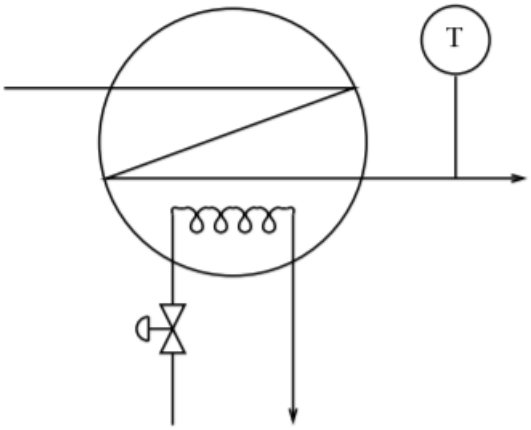

Figure 1: Heating process.

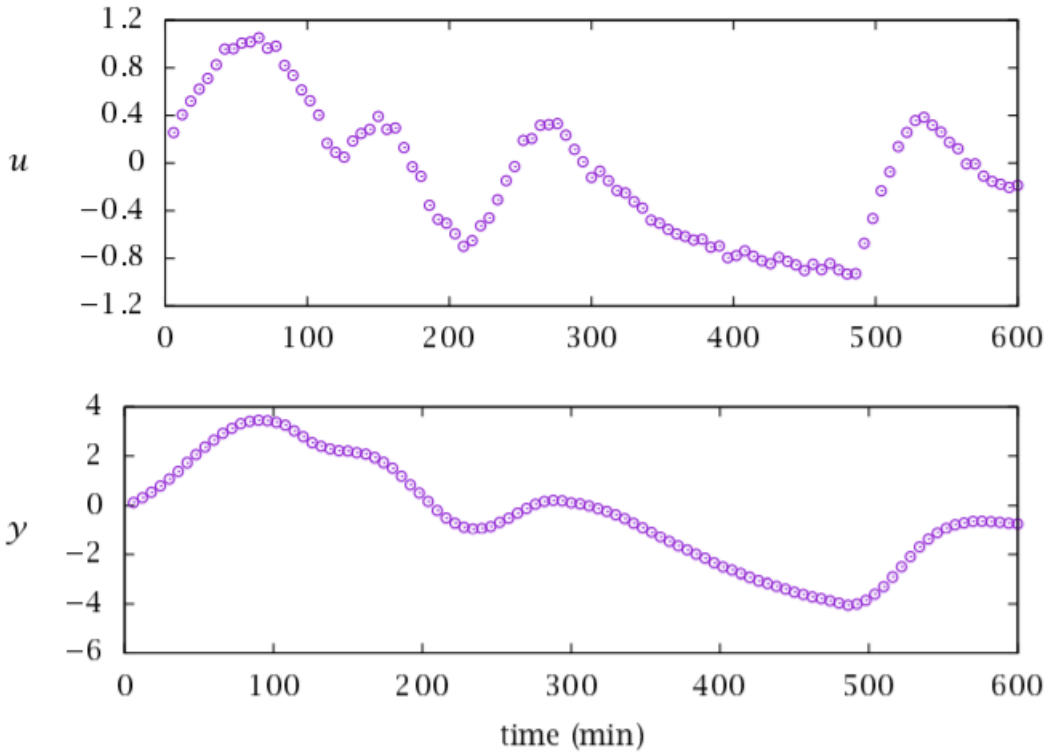

Figure 2: Valve position (input) and temperature (output) versus time (sample number) for the heating process.

Now we want to use the model to predict the temperature response, which we can compare to the actual plant measurements. How do we take this $u(t)$ and use the model to compute $y(t)$? Notice I do not have a nice clean step input, or a nice impulse, or a nice sine wave, or a nice anything to take Laplace transforms and use partial fractions to invert back to the time domain. I have a big table of times and valve positions taken from my process data historian.

Here's one approach. Convert the model to a discrete difference equation

$$x(k+1) = Ax(k) + Bu(k)$$

$$y(k) = Cx(k)$$

(a)

What sample time do you choose?

(b)

Using the difference equation, write a sample program to solve this model and plot $y(t)$ and the measurement on the same plot.

(c)

How does the simulation of $y(t)$ compare to the measurement?

Solution:

(a)

We shall use a sample time equal to that present in the given data, $t_s = 6$ minutes. Assuming that the temperature and valve position measurements we are viewing in this data set are also used by the plant in any applicable control loops, we conclude it is likely that these control loops will only update the valve position once every 6 minutes, when a new temperature reading is available. We will further assume the valve position changes quickly, such that we can approximate the valve position by supposing there is a zero-order hold on the given valve position input data. With this assumption, we know that if we set the sample time for our simulation to 6 minutes as well, then our discrete difference equation will give an exact solution for the evolution of the continuous time system.

(b)

The following `MATLAB` code on the following page was used to simulate the system and plot the results alongside the given data, shown in Figure 3.

```
%Connor Hughes
%CH E 152B HW3

%% Exercise 10: Heating Process
close all;
%read data from file:
T = readtable('heatdat.txt');
M = table2array(T);

%initialize parameters:
tau = 1;
tau = tau*60;   %convert to minutes to match sample time units
k = 5;

%initialize continuous time system matrices:
A = -1/tau;
B = k/tau;
C = 1;
D = 0;

%convert to discrete time:
ts = 6;    %set sample time to 6 min to match given data
A_d = expm(ts*A);
B_d = inv(A)*(expm(ts*A) - 1)*B;
C_d = C;
D_d = D;

%simulate the system:
U = M(:, 2);
Y = NaN(1, 100);
X = NaN(1, 100);
X(1) = 0.1280;
for i =1:99
    X(i + 1) = A_d*X(i) + B_d*U(i);
    Y(i) = C_d*X(i);
end
Y(100) = C_d*X(100);

%plot the results:
subplot(2, 1, 1)
plot(M(:, 1), M(:, 2), 'Marker', 'o', 'LineStyle', 'none');
ylabel('u (Valve Position input)See');
subplot(2, 1, 2)
plot(M(:, 1), M(:, 3), 'Marker', 'o', 'LineStyle', 'none');
hold on
plot(M(:, 1), Y(:), 'Marker', 'o', 'LineStyle', 'none');
legend('Measured Temperature', 'Simulated Temperature')
ylabel('Temperature (deg C)')
xlabel('Time (min)')
```

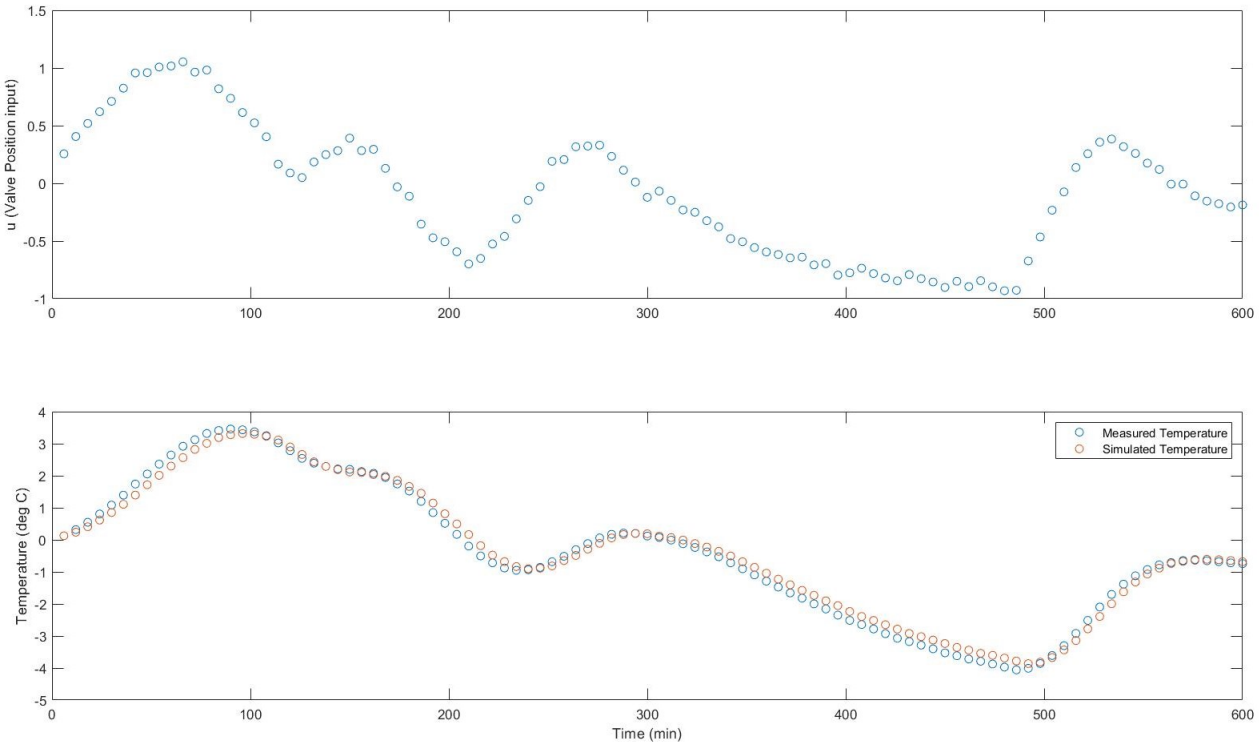

Figure 3: Measured and simulated temperature data, along with valve position input.

(c)

Notice in the above plot that the simulated temperature data matches the measured data closely, though it lags the measured data just a bit. This suggests our model provides a fairly accurate description of the actual system dynamics. The lag may be caused by some time delay inherent in the system which is neglected in our first-order model. Alternatively, perhaps it has resulted because our assumption that the valve position could be considered to change near-instantaneously was incorrect, and as a result the zero-order hold we've assumed for the valve position input data is causing the delay. If in actuality the valve position is slowly and near-continuously changing, and is already moving toward the next set point before we see the corresponding data point, then our assumption of a zero-order hold would mean that the valve position impacts the simulated system just a bit after the actual plant was impacted by the same valve position, hence causing a delay to appear.