Problem 1.
Consider the pendulum model with input the torque applied at the pivot point and output the angle of the pendulum. (Assume that the pendulum is a rigid rod of length 0.5m, mass 200g evenly distributed, and its rotation around the pivot point is frictionless.)

1. Design a state observer to estimate the angle and angular velocity from noisy angle measurements.
2. Collect 20s of simulation data at 100Hz with random 10Hz excitation around the stable equilibrium such that the amplitude of oscillation does not exceed 6degrees. Implement a 12-bit quantization on the angle measurement for the 360degree range and a 10-bit quantization on the torque for the range [-1, 1]. Formulate the parameter estimation problem and use the batch least-squares algorithm to estimate the parameters of the corresponding transfer function. Illustrate your findings with a few well-chosen simulations.

We start with the pendulum model
\[ J \ddot{\theta} = T - \frac{mgl}{2} \sin \theta - \epsilon \dot{\theta} | \dot{\theta} | \]
Where m is the mass, L is the length, \( J = \frac{ml^2}{3} \) is the inertia, and \( \epsilon \) is the friction coefficient for the pendulum, and \([T, \theta] \) is the I/O pair. The torque T is proportional to the current driving the pendulum motor, but since we have no further data, we will assume a proportionality constant of 1. Linearizing the model around the stable equilibrium \([0, 0] \), we obtain the transfer function
\[ P(s) = \frac{60}{s^2 + 29.43} \]
And the state-space realization in terms of angle and angular velocity
\[ \dot{x} = \begin{bmatrix} 0 & 1 \\ -29.43 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 60 \end{bmatrix} u, \]
\[ y = \begin{bmatrix} 1 & 0 \end{bmatrix} x \]

For the discrete-time model, to be used for state estimation, we find the ZOH equivalent:
\[ x_{k+1} = (I + 10^{-2} \begin{bmatrix} -0.1471 & 0.9995 \\ -29.42 & -0.1471 \end{bmatrix}) x_k + 10^{-2} \begin{bmatrix} 0.2999 \\ 59.97 \end{bmatrix} u_k, \]
\[ y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} x_k \]

For this we define the state observer
\[ \hat{x}_{k+1} = A\hat{x}_k + Bu_k + L(y_k - \hat{y}_k), \]
\[ \hat{y}_k = C\hat{x}_k \]
where L is the observer gain which can computed using a variety of approaches. One, particularly attractive method is by using the Kalman Filter equations in their steady-state solution, given by the discrete Riccati equation \( L = A\Sigma C^T [C\Sigma C^T + R]^{-1}, \Sigma = A\Sigma A^T + GQG^T - A\Sigma C^T [C\Sigma C^T + R]^{-1}C\Sigma A^T. \) While this equation, taken as a recursion, will converge to the steady-state solution, MATLAB also implements efficient numerical methods to solve it:
\[ >> L = dlqe(A,G,C,Q,R) \]
Here, G,Q are the input and intensity (covariance) matrices for the state noise and R is the intensity of the output noise. Since we do not have any additional information to model the noise, or optimize specific aspects of the Kalman Filter response, we will simply choose, \( G = I \), \( Q = BB' \), and \( R = a \) small scalar, to be iterated until a "reasonable" speed of convergence is obtained. For example,
\[ >> Hd = c2d(H,0.01) \]
\[ >> L=dlqe(Hd.a,eye(2,2),Hd.c,Hd.b*Hd.b',0.01),abs(eig(Hd.a-L*Hd.c)) \]
This yieds
\[ L = [0.2865, 4.823] \]
and magnitude of the observer error system eigenvalues 0.873; the latter implies convergence of the error system in 20 samples, or 0.2 sec, which is a reasonable time from a feedback control perspective. (In a quick design, the crossover of the feedback system would be selected around 10-20 rad/s, a factor of 2-4 above the bandwidth of the system poles, both for the stable and the unstable equilibrium case.)

Finally, for implementation purposes, it is often a good idea to use a controller to stabilize the system so that its response stays bounded for any possible test condition. (Especially, for system identification applications.) Omitting the details, here we design a PID to provide 50deg phase margin at 13rad/s:

\[
[K_p, K_i, K_d] = [1.3429e+000 \quad 2.6944e+000 \quad 1.6398e-001]
\]

Next, we construct a simulation model to solve the nonlinear pendulum equation, and connect the observer to the system I/O.

Pendulum Subsystem:
Observer Subsystem:

This simulation model allows the study of observer and identification problems under a variety of conditions. We list some below:

- Convergence for different initial conditions (defined in the Pendulum mask)
- Convergence with and without the PID controller, with and without random excitation, with and without output noise
- Use of different observer gains, obtained with different output noise weights (R) in the Riccati equation
- Stable and unstable equilibrium (requires adjustment of the observer model).

Example: Uncontrolled system (for the unstable equilibrium such tests can be performed only for short time intervals), starting with I.C. [0.1, 0]. Here, the angle output is noisy but the velocity is not. Their estimates present a “smoothed” version of the angle, but the velocity estimate is noisy. For a 20s interval, the two traces overlap. With a zoom-in during the initial transient, we can observe the convergence, which takes roughly 0.2s as predicted from the eigenvalues of the observer error subsystem.
For the identification experiment, we connect the excitation at the pendulum input. With zero I.C., and after some trial-and-error we find a gain for the excitation (0.02) which causes the angle deviations to be below 6 degrees. (This is necessary to keep the system near the linearization point where \( \sin \theta \approx \theta \).)

We collect the data \((U,Y)\) and form a regressor for a second order system. For a generalization, we define the filter \( F \) (e.g., a delay) and then write the regressor
\[
w = [FY, FFTY, Fu, FFu]
\]
(for a general case of regressor construction, see a system identification text). Then, the LS approximation problem has a solution
\[
q = w \backslash y = (w^T w)^{-1} w^T y
\]
From which the identified system can be expressed as
\[
H = \frac{q(3)F + q(4)FF}{1 - q(1)F - q(2)FF}
\]
The MATLAB implementation of this algorithm is shown below
\[
\text{>> F=c2d(tf(1,[.1 1]),.01)}
\]
\[
\text{>> w=[lsim(F,Y(:,2)),lsim(F*F,Y(:,2)),lsim(F,U),lsim(F*F,U)];q=w\backslash Y(:,2)}
\]
\[
\text{>> Hd=minreal((q(3)*F+q(4)*F*F)/(1-q(1)*F-q(2)*F*F)), H=d2c(Hd)}
\]
Then
\[
H_d(z) = \frac{0.001572 z + 0.004358}{z^2 - 1.997 z + 0.9995}, \quad H_c(s) = \frac{-0.1394 s + 59.33}{s^2 + 0.04907 s + 29.36}
\]
Notice that the model coefficients are fairly “close” to the true linearization \((P)\). However, the identification of the resonance is usually a difficult task and some “smearing” of the peak occurs. A similar result is obtained with the controller in feedback, but now the excitation must be increased by an order of magnitude to achieve the same range of output variation. Otherwise, the output noise causes the signal to noise ratio (SNR) to decrease and the accuracy of the identification deteriorates.

Finally, identification with the pure ARX regressor (delay, \( F = tf([1,1],[1 0]),.01)\)) is unsuccessful for this case, because it puts too much emphasis on the high frequencies.
\[
H_d(z) = \frac{-0.006659 z + 0.02488}{z^2 - 0.6264 z - 0.3609}
\]