This homework is due on Thursday, October 29, 2020, at 10:59PM. Self-grades are due on Thursday, November 5, 2020, at 10:59PM.

Clarification: You will have to submit code for Q2 and Q3. You can either attach a screenshot to your written submission or you can submit the .ipynb to the HW 8 Code Submission assignment on Gradescope.

1 Eigenvalue Placement

Consider the following linear discrete time system

$$\vec{x}(t+1) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -9 & -6 \end{bmatrix} \vec{x}(t) + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u(t) \tag{1}$$

a) What are the eigenvalues of this system? Is this system stable?

Solution

We can compute the eigenvalues through the characteristic polyomial

$$\det(A - \lambda I) = \det \begin{bmatrix} -\lambda & 1 & 0 \\ 0 & -\lambda & 1 \\ 0 & -9 & -6-\lambda \end{bmatrix} = -\lambda \begin{vmatrix} -\lambda & 1 \\ -9 & -6-\lambda \end{vmatrix} = -\lambda(\lambda^2 + 6\lambda + 9) = 0$$

Therefore, $\lambda = 0, -3$. The system is unstable since $|-3| \geq 1$ is outside of the unit circle.

b) Show that this system is controllable.

Solution

The controllability matrix can be computed as

$$C = \begin{bmatrix} B & AB & A^2B \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & -6 \\ 1 & -6 & 27 \end{bmatrix}$$

Since this matrix has rank 3, the system is controllable.

c) Using state feedback $u(t) = K\vec{x}(t) = \begin{bmatrix} k_0 & k_1 & k_2 \end{bmatrix} \vec{x}(t)$ design a controller to place the closed-loop system's eigenvalues at $0, 1/2, -1/2$.

Solution

The characteristic polynomial for the feedback matrix $A + BK$ will be:

$$\lambda^3 - (-6 + k_2)\lambda^2 - (-9 + k_1)\lambda - k_0 = 0$$

In order to place the eigenvalues at $0, \frac{1}{2}, -\frac{1}{2}$ we desire the characteristic polynomial:

$$\lambda^3 - \frac{1}{4}\lambda = 0$$

Therefore by matching coefficients, we see that

$$6 - k_2 = 0$$
$$9 - k_1 = -\frac{1}{4}$$
$$k_0 = 0$$

So it follows that

$$k_0 = 0, k_1 = \frac{37}{4}, k_2 = 6$$

d) Upon implementing this controller in the lab, we run into issues since our hardware can only produce K values in between -5 and 5. **Is it still possible to design a controller that can stabilize the original system given in (1)?**

Solution

If all of our eigenvalues were on the unit circle at $\lambda = -1$, then the characteristic polynomial would be

$$(\lambda + 1)^3 = \lambda^3 + 3\lambda^2 + 3\lambda + 1 = 0$$

If we picked eigenvalues inside the unit circle, the coefficients to the characteristic polynomial would be smaller.

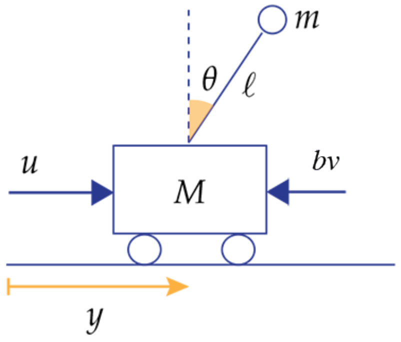
This tells us that the coefficients of the characteristic polynomial must be less than 3. However, if we try to set the λ coefficient to be less than 3, that would require $k_1 > 6$ which is not possible with our controller. Therefore, we cannot stabilize this system.

2 Inverted Pendulum on a Rolling Cart

Note: In this problem we use the state-feedback policy $\vec{u} = -K\vec{x}$ to stabilize the eigenvalues of $A - BK$ instead of $A + BK$ in order to use external libraries to solve for K. This should not change any of the empirical results, but the values of K will be negated if you use $\vec{u} = K\vec{x}$ to pick the eigenvalues.

Consider the inverted pendulum depicted below, which is placed on a rolling cart and whose equations of motion are given by:

$$\begin{aligned}\ddot{y} &= \frac{1}{M + m\sin^2\theta}\left(u - m\dot{\theta}^2\ell\sin\theta + mg\sin\theta\cos\theta - b\dot{y}\right)\\ \ddot{\theta} &= \frac{1}{\ell(M + m\sin^2\theta)}\left(u\cos\theta + m\dot{\theta}^2\ell\sin\theta\cos\theta + (M+m)g\sin\theta - b\dot{y}\cos\theta\right).\end{aligned}$$



The dots above variables are shorthand for derivatives – that is ($\dot{x} = \frac{d}{dt}x$ and $\ddot{x} = \frac{d^2}{dt^2}x$).

a) Write out the state model using the variables $x_1 = y, x_2 = \dot{y}, x_3 = \theta, x_4 = \dot{\theta}$.

Solution

Taking the derivative of each state, we see that

$$\begin{aligned}\dot{x_1} &= \dot{y} = x_2 && \triangleq f_1(x_1, x_2, x_3, x_4, u)\\ \dot{x_2} &= \ddot{y} = \frac{u - m\dot{\theta}^2\ell\sin\theta + mg\sin\theta\cos\theta - b\dot{y}}{M + m\sin^2\theta}\\ &= \frac{u - mx_4^2\ell\sin x_3 + mg\sin x_3\cos x_3 - bx_2}{M + m\sin^2 x_3} && \triangleq f_2(x_1, x_2, x_3, x_4, u)\\ \dot{x_3} &= \dot{\theta} = x_4 && \triangleq f_3(x_1, x_2, x_3, x_4, u)\\ \dot{x_4} &= \ddot{\theta} = \frac{u\cos\theta + m\dot{\theta}^2\ell\sin\theta\cos\theta + (M+m)g\sin\theta - b\dot{y}\cos\theta}{\ell(M + m\sin^2\theta)}\\ &= \frac{u\cos x_3 + m\ell x_4^2\sin x_3\cos x_3 + (M+m)g\sin x_3 - bx_2\cos x_3}{\ell(M + m\sin^2 x_3)} && \triangleq f_4(x_1, x_2, x_3, x_4, u)\end{aligned}$$

b) Linearize this model at the equilibrium $x_1 = 0$, $x_2 = 0$, $x_3 = 0$, $x_4 = 0$ and $u = 0$, and indicate the resulting A and B matrices.

Hint: When taking partial derivatives, all other variables will be constant. For example, if we need to evaluate $\frac{\partial f}{\partial x_1}$, we can plug in $x_2 = x_3 = x_4 = u = 0$ before taking any derivatives.

Solution

We can keep in mind that $x_1^* = x_2^* = x_3^* = x_4^* = u^* = 0$ to make the derivative much easier.

$$\begin{array}{llll}
\frac{\partial f_1}{\partial x_1}(0,0,0,0,0) = 0 & \frac{\partial f_1}{\partial x_2}(0,0,0,0,0) = 1 & \frac{\partial f_1}{\partial x_3}(0,0,0,0,0) = 0 & \frac{\partial f_1}{\partial x_4}(0,0,0,0,0) = 0 \\
\frac{\partial f_2}{\partial x_1}(0,0,0,0,0) = 0 & \frac{\partial f_2}{\partial x_2}(0,0,0,0,0) = -\frac{b}{M} & \frac{\partial f_2}{\partial x_3}(0,0,0,0,0) = \frac{m}{M}g & \frac{\partial f_2}{\partial x_4}(0,0,0,0,0) = 0 \\
\frac{\partial f_3}{\partial x_1}(0,0,0,0,0) = 0 & \frac{\partial f_3}{\partial x_2}(0,0,0,0,0) = 0 & \frac{\partial f_3}{\partial x_3}(0,0,0,0,0) = 0 & \frac{\partial f_3}{\partial x_4}(0,0,0,0,0) = 1 \\
\frac{\partial f_4}{\partial x_1}(0,0,0,0,0) = 0 & \frac{\partial f_4}{\partial x_2}(0,0,0,0,0) = -\frac{b}{M\ell} & \frac{\partial f_4}{\partial x_3}(0,0,0,0,0) = \frac{M+m}{M\ell}g & \frac{\partial f_4}{\partial x_4}(0,0,0,0,0) = 0
\end{array}$$

And,

$$\frac{\partial f_1}{\partial u}(0,0,0,0,0) = 0 \quad \frac{\partial f_2}{\partial u}(0,0,0,0,0) = \frac{1}{M} \quad \frac{\partial f_3}{\partial u}(0,0,0,0,0) = 0 \quad \frac{\partial f_4}{\partial u}(0,0,0,0,0) = \frac{1}{M\ell}$$

Since $x^* = \vec{0}$ and $u^* = 0$, the linearized state variables $x_\ell = x$ and $u_\ell = u$.

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{b}{M} & \frac{m}{M}g & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{b}{M\ell} & \frac{M+m}{M\ell}g & 0 \end{bmatrix}}_{A} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{M\ell} \end{bmatrix}}_{B} u$$

c) Let $m = 1\,\text{kg}, M = 5\,\text{kg}, L = 2\,\text{m}, g = 10\,\frac{\text{m}}{\text{s}^2}, b = 1\,\frac{\text{kg}}{\text{s}}$. Show that the linearized model is controllable but unstable. Then plot the response of the system in the Jupyter notebook and explain how the cart-pendulum system behaves if we push the cart with a small input force.

Solution

To show the system is controllable, we can compute the controllability matrix.

$$C = \begin{bmatrix} B & AB & A^2 & A^3B \end{bmatrix} = \begin{bmatrix} 0 & 0.2 & -0.04 & 0.208 \\ 0.2 & -0.04 & 0.208 & -0.0816 \\ 0 & 0.1 & -0.02 & 0.604 \\ 0.1 & -0.02 & 0.604 & -0.1408 \end{bmatrix}$$

```
p = np.linalg.matrix_power
C = np.column_stack((B, A.dot(B), p(A,2).dot(B), p(A,3).dot(B)))
'The rank of the controllabilty matrix is 4'
```

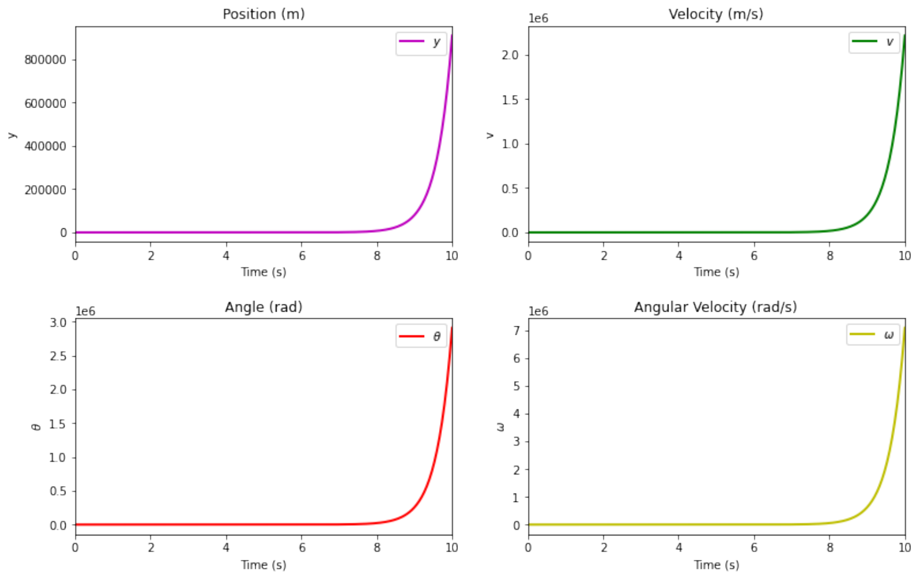
This matrix has rank 4 so the system is indeed controllable.

To determine stability, we can compute the eigenvalues of A

```
eigs = np.linalg.eig(A)[0]
'The eigenvalues of the state matrix are [0. -2.47 -0.167 2.43]'
```

The system is unstable since $\Re\mathfrak{e}[2.43] \geq 0$.

Looking at the plots of the response $\vec{x}$ for $t \in [0, 10]$, all of the states increase without bound.



From a physical perspective, the states won't actually increase without bound, but if we slightly perturb the pendulum in its upright position, then it will quickly fall down. Remember that we are plotting the response of the linearized system. Since the linear model is unstable, the linear approximation will no longer be valid for values of $\vec{x}$ far away from the equilibrium $\vec{x}^*$.

d) Since the system is controllable, we can design a state-feedback controller $u = -K\vec{x}$ to assign the eigenvalues of $A_{cl} = A - BK$. Find the values of K using the Jupyter Notebook or by hand to place the closed-loop eigenvalues at $\begin{bmatrix}-2 & -1.4 & -1.3 & -1.2\end{bmatrix}$.

Solution

We can use the `place_poles` function from `scipy.signal` to set the eigenvalues at $[-2, -1.4, -1.3, -1.2]$.

```
eigs = [-2, -1.4, -1.2, -1.3]
K = place_poles(A, B, eigs).gain_matrix
```

Therefore, we can conclude that the values of K are

$$K = \begin{bmatrix}-4.37 & -13.3 & 197.4 & 83.6\end{bmatrix}$$

Self-Grade Note: If you set the eigenvalues at $[-2, -1.3, -1.2, -1.1]$, the K values will be

$$K = \begin{bmatrix}-3.43 & -11.3 & 181.96 & 76.7\end{bmatrix}$$

e) Now that we have stabilized our system, let's analyze how we can move our system to a target state $\vec{r}$. To do this, we define $\vec{e} = \vec{r} - \vec{x}$ as the error between the state $\vec{x}$ and the target $\vec{r}$.

Then we will use the control policy $u = K\vec{e} = K(\vec{r} - \vec{x})$. **Show that if $\vec{r} \in \textbf{Nul}(A)$, we can write out the following closed-loop state-space model for $\vec{e}$.**

$$\frac{d}{dt}\vec{e} = (A - BK)\vec{e} \tag{2}$$

Explain why the state $\vec{x}$ converges to the reference state $\vec{r}$ as $t \longrightarrow \infty$.

Solution

Let's start by taking the derivative of $\vec{e} = \vec{r} - \vec{x}$.

$$\frac{d}{dt}\vec{e} = -\frac{d}{dt}\vec{x} = -(A\vec{x} + B\vec{u})$$

With the input $u = K(\vec{r} - \vec{x})$, the dynamics of the system can be written as

$$\begin{aligned}\frac{d}{dt}\vec{e} &= -A\vec{x} - BK(\vec{r} - \vec{x}) \\ &= -BK\vec{r} - (A - BK)\vec{x}\end{aligned}$$

If $\vec{r} \in \text{Nul}(A)$, then $A\vec{r} = \vec{0}$, meaning we can add $A\vec{r}$ to get

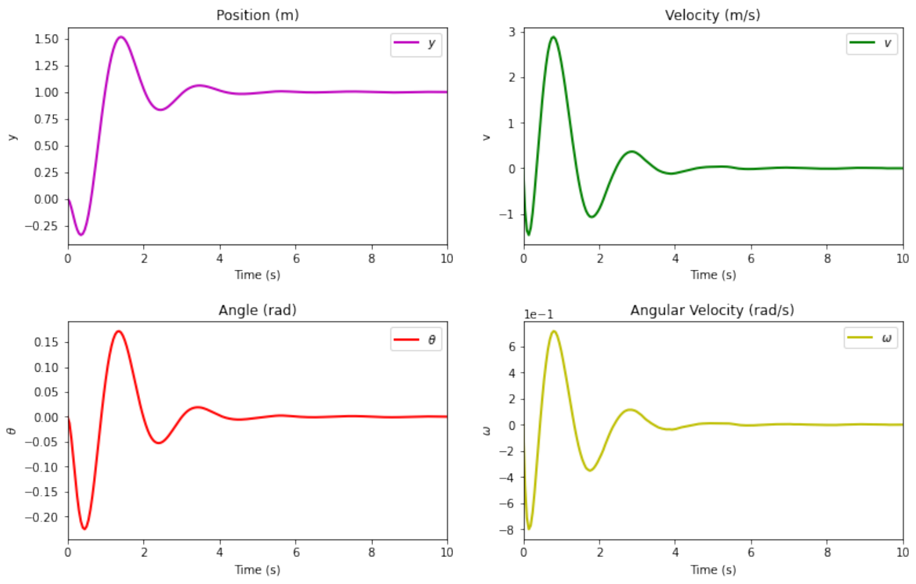
$$\begin{aligned}\frac{d}{dt}\vec{e} &= A\vec{r} - BK\vec{r} - (A - BK)\vec{x} \\ &= (A - BK)(\vec{r} - \vec{x}) \\ &= (A - BK)\vec{e}\end{aligned}$$

Since we have stabilized our original system so that the eigenvalues of $A - BK$ have real part less than zero, this new error system will be also stable meaning $\vec{e}(t)$ converges to $\vec{0}$. This is equivalent to saying $\vec{x}$ will converge to $\vec{r}$.

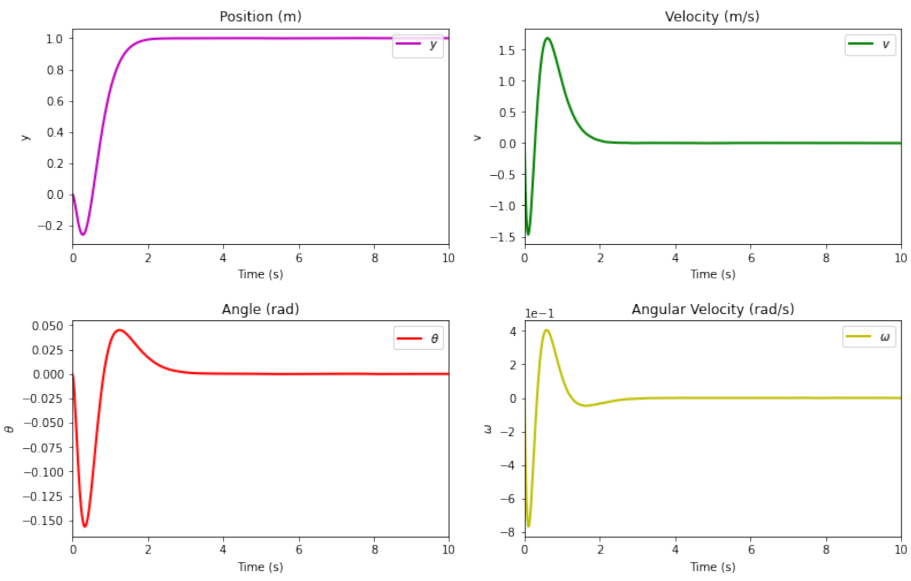
f) Pick different eigenvalues for the closed-loop matrix and run the simulation of the cart-pendulum system. Then explain the effect that the eigenvalues have on the system.

Solution

If our system has complex eigenvalues, we will observe oscillations in the response.

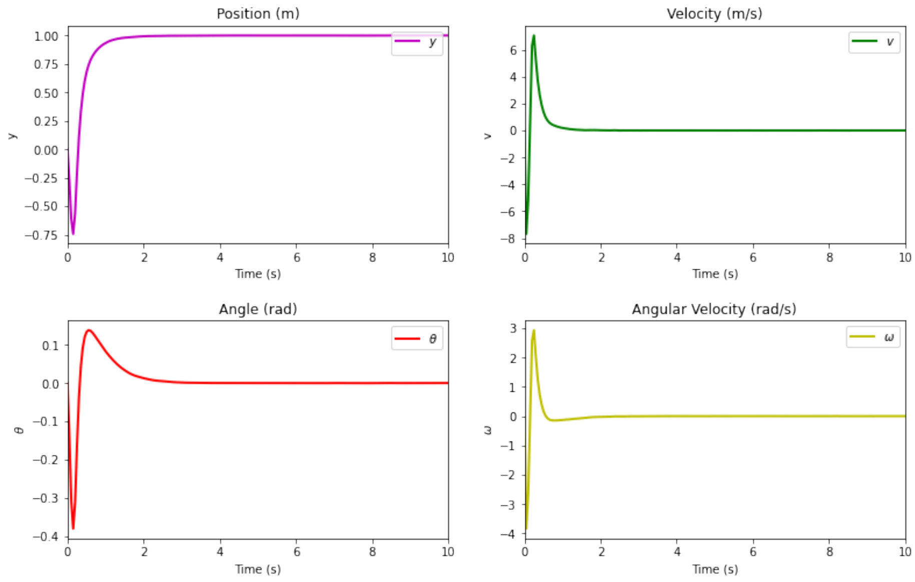


On the other hand, if all of the eigenvalues are purely real, then the trajectory will be much smoother.



As we place the eigenvalues further into the left-half plane (large and negative), the system converges to the reference much quicker. However, notice that in order to converge quicker, the cart-pendulum needs to move to the left before moving right to the target $y = 1$.

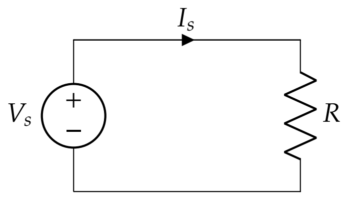
If the values of K are too large, then the cart-pendulum will have to move further left and have very "jerky" movement. In a physical context, it will also cost much more energy to amplify $\vec{x}$. For example, when eigs = [-20, -10, -5, -2], values of K will be around 8000 which will not be feasible to implement in practice.



3 System Identification

Taejin is building his SIXT33N car in the lab and needs your help on identifying some unknown parameters. Each part will require you to set up and solve a Least-Squares problem in the provided Jupyter Notebook.

a) While building the front-end circuit, he notices that one of his resistors has no color bands. Therefore, he decides to apply voltages V_s and measure the current I_s across the resistor.



Using the measurement data provided in the Jupyter Notebook, **set up and solve a least-squares problem to estimate the resistance** R.

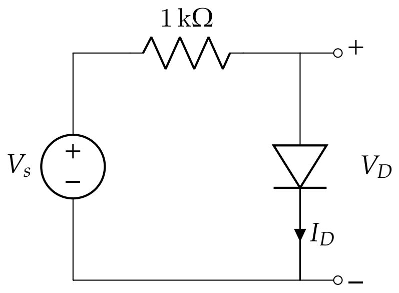
Solution

Ohm's law states that $V = I \cdot R$. Given the experimental data (V, I), we can set up a Least-Squares problem of the form

$$\begin{bmatrix} I_1 \\ \vdots \\ I_n \end{bmatrix} = \begin{bmatrix} V_1 \\ \vdots \\ V_n \end{bmatrix} \cdot w \tag{3}$$

where $w = \frac{1}{R}$. Solving the Least-Squares problem, $w = 1.15 \cdot 10^{-3}$ so $R = 871\,\Omega$.

b) Taejin continues to build the front-end circuit, but he comes across a circuit device he's never seen before. After consulting Ramsey, the circuits expert, he learns that $I_D = I_s e^{V_D/V_{TH}}$.



Here I_s is the saturation current and V_{TH} is the thermal voltage both of which depend on the device physics. To obtain experimental data, Taejin applies voltages V_s and measures voltages and currents V_D and I_D across the mystery device. **Set up and solve a least-squares problem in the Jupyter Notebook to estimate I_s and V_{TH}.**

Solution

We are given the observations (V_D, I_D) but the relationship between I_D, V_D, and V_{TH} is nonlinear. Therefore, we need to transform our observations to obtain a linear model.

$$I_D = I_s e^{V_D/V_{TH}}$$
$$\ln(I_D) = \ln(I_s) + \frac{1}{V_{TH}} \cdot V_D$$

Given our datapoints, we can define variables $y = \ln(I_D)$, $x = V_D$ and set up of a linear model

$$y = mx + b \tag{4}$$

where $m = \frac{1}{V_{TH}}$ and $b = \ln(I_s)$ are unknown paramters we need to solve for. The feature matrix, weights, and observations can be defined as

$$X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \quad \vec{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \quad \vec{w} = \begin{bmatrix} m \\ b \end{bmatrix} \tag{5}$$

meaning the Least-Squares problem and solution will be of the form

$$\min_{\vec{w}} \left\| X\vec{w} - \vec{y} \right\| \implies \hat{\vec{w}} = (X^T X)^{-1} X^T \vec{y} \tag{6}$$

After solving the Least-Squares problem, we see that $\vec{w} = \begin{bmatrix} 33 \\ -20 \end{bmatrix}$ meaning

$$V_{TH} = \frac{1}{33} = 30\,\mathrm{mV} \qquad I_s = e^{-20} = 1.95\,\mathrm{nA}$$

c) Now Taejin moves onto building the car. His lab partner, Nick, tells him that the position of the car can be formulated as a discrete-time system of the form

$$\begin{bmatrix} d_L(t+1) \\ d_R(t+1) \end{bmatrix} = A \begin{bmatrix} d_L(t) \\ d_R(t) \end{bmatrix} + B \begin{bmatrix} u_L(t) \\ u_R(t) \end{bmatrix} \tag{7}$$

The car's wheels start at $d_L(0) = d_R(0) = 0$. They apply the sequence of inputs

$$u_L(0), u_R(0), u_L(1), u_R(1), \dots, u_L(98), u_R(98), u_L(99), u_R(99),$$

and measure the car's position

$$d_L(1), d_R(1), d_L(2), d_R(2), \dots, d_L(99), d_R(99), d_L(100), d_R(100),$$

at each time step. Using the measurement data provided in the notebook, **set up and solve a least-squares problem to find what the A and B matrices are.**

Solution

The matrices A and B are the unknowns in this question. We'll label each of the inidividual entries of A and B as

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

Since $\vec{u} \in \mathbb{R}^2$, B will be a 2×2 matrix.

Now let's try writing out the first couple of states in terms of the previous state with measurement error ϵ

$$\begin{aligned} d_L(1) &= a_{11}d_L(0) + a_{12}d_R(0) + b_{11}u_L(0) + b_{12}u_R(0) + \epsilon_L(0) \\ d_R(1) &= a_{21}d_L(0) + a_{22}d_R(0) + b_{21}u_L(0) + b_{22}u_R(0) + \epsilon_R(0) \\ d_L(2) &= a_{11}d_L(1) + a_{12}d_R(1) + b_{11}u_L(1) + b_{12}u_R(1) + \epsilon_L(1) \\ d_R(2) &= a_{21}d_L(1) + a_{22}d_R(1) + b_{21}u_L(1) + b_{22}u_R(1) + \epsilon_R(1) \end{aligned}$$

Each set time-step will give us two equations that can be put into the following matrix form:

$$\begin{bmatrix} d_L(0) & d_R(0) & 0 & 0 & u_L(0) & u_R(0) & 0 & 0 \\ 0 & 0 & d_L(0) & d_R(0) & 0 & 0 & u_L(0) & u_R(0) \\ d_L(1) & d_R(1) & 0 & 0 & u_L(1) & u_R(1) & 0 & 0 \\ 0 & 0 & d_L(1) & d_R(1) & 0 & 0 & u_L(1) & u_R(1) \end{bmatrix} \begin{bmatrix} a_{11} \\ a_{12} \\ a_{21} \\ a_{22} \\ b_{11} \\ b_{12} \\ b_{21} \\ b_{22} \end{bmatrix} = \begin{bmatrix} d_L(1) \\ d_R(1) \\ d_L(2) \\ d_R(2) \end{bmatrix} + \vec{\epsilon}$$

There are two ways to solve this problem, the second method is easier to implement in practice. **Method 1:** We can define the data matrix D, observations $\vec{y}$, and the weights $\vec{w}$ as

$$D = \begin{bmatrix} d_L(0) & d_R(0) & 0 & 0 & u_L(0) & u_R(0) & 0 & 0 \\ 0 & 0 & d_L(0) & d_R(0) & 0 & 0 & u_L(0) & u_R(0) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ d_L(99) & d_R(99) & 0 & 0 & u_L(99) & u_R(99) & 0 & 0 \\ 0 & 0 & d_L(99) & d_R(99) & 0 & 0 & u_L(99) & u_R(99) \end{bmatrix} \quad \vec{w} = \begin{bmatrix} a_{11} \\ a_{12} \\ a_{21} \\ a_{22} \\ b_{11} \\ b_{12} \\ b_{21} \\ b_{22} \end{bmatrix} \quad \vec{y} = \begin{bmatrix} d_L(1) \\ d_R(1) \\ \vdots \\ d_L(99) \\ d_R(99) \end{bmatrix}$$

Therefore, we can set up and solve the Least-Squares problem as follows

$$\min_{\vec{w}} \left\| D\vec{w} - \vec{y}_L \right\|^2 \implies \hat{\vec{w}} = (D^T D)^{-1} D^T \vec{y}$$

Method 2: Alternatively, we can notice the redundant entries in our data-matrix and instead create a single data-matrix with two sets of observations and weights

$$D = \begin{bmatrix} d_L(0) & d_R(0) & u_L(0) & u_R(0) \\ d_L(1) & d_R(1) & u_L(1) & u_R(1) \\ \vdots & \vdots & \vdots & \vdots \\ d_L(99) & d_R(99) & u_L(99) & u_R(99) \end{bmatrix}$$

$$\vec{w}_L = \begin{bmatrix} a_{11} \\ a_{12} \\ b_{11} \\ b_{12} \end{bmatrix} \quad \vec{y}_L = \begin{bmatrix} d_L(1) \\ d_L(2) \\ \vdots \\ d_L(100) \end{bmatrix} \quad \vec{w}_R = \begin{bmatrix} a_{21} \\ a_{22} \\ b_{21} \\ b_{22} \end{bmatrix} \quad \vec{y}_R = \begin{bmatrix} d_R(1) \\ d_R(2) \\ \vdots \\ d_R(100) \end{bmatrix}$$

Therefore, we can setup two independent Least-Squares problems with weights $\vec{w}_L$ and $\vec{w}_R$.

$$\min_{\vec{w}_L} \left\| D\vec{w}_L - \vec{y}_L \right\|^2$$
$$\min_{\vec{w}_R} \left\| D\vec{w}_R - \vec{y}_R \right\|^2$$

Writing out the matrix in block form, notice that

$$\begin{bmatrix} \vec{y}_L & \vec{y}_R \end{bmatrix} = D \begin{bmatrix} \vec{w}_L & \vec{w}_R \end{bmatrix} + \begin{bmatrix} \epsilon_L & \epsilon_R \end{bmatrix}$$

The Least-Sqaures solution can be written as

$$\begin{bmatrix} \vec{w}_L & \vec{w}_R \end{bmatrix} = (D^T D)^{-1} D^T \begin{bmatrix} \vec{y}_L & \vec{y}_R \end{bmatrix}$$

In the Jupyter Notebook, we can implement this as follows

```
X, y = dists[:-1,], dists[1:,]
u = inputs[:-1,]
D = np.column_stack((X, u))

w = np.linalg.lstsq(D, y, rcond=None)[0]
A = w[:2,].T
B = w[2:,].T
```

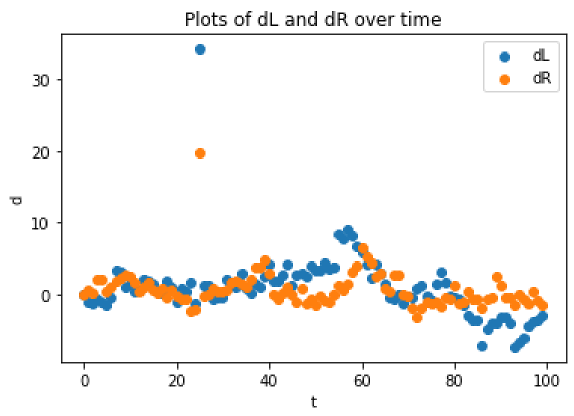
Equating the coefficients of $\vec{w}$ to the A and B matrices, we see that

$$A = \begin{bmatrix} 0.69 & -0.67 \\ -0.06 & 0.38 \end{bmatrix} \quad B = \begin{bmatrix} 1.27 & 0.11 \\ 0.26 & 0.47 \end{bmatrix}$$

d) Upon solving for A and B and providing new inputs to the system, they notice that the car isn't responding to the inputs properly. Take a look at the data-points and explain why this might be the case. **How can Taejin and Nick fix this issue to make their car run smoothly?**

Solution

Let's start by plotting that data-points d_L and d_R over time

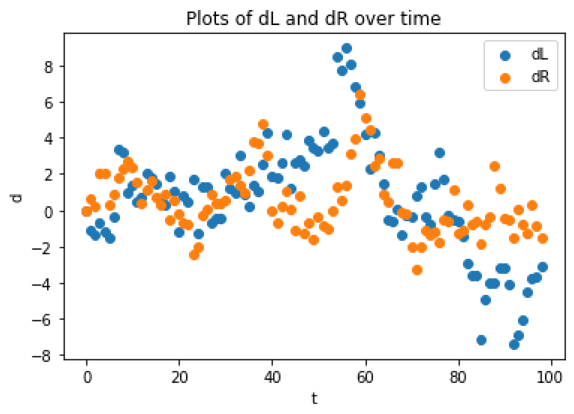


We can notice that there is a single outlier point that is corrupting our Least-Squares solution. There are other techniques to detect and remove outliers, but we won't go into these techniques and remove it manually.

```
idx = np.argmax(X[:,0])

X = np.delete(X.copy(), idx, 0)
y = np.delete(y.copy(), idx-1, 0)
u = np.delete(u.copy(), idx-1, 0)
D = np.column_stack((X, u))
```

and re-plot the data-points d_L and d_R



Recomputing the Least-Squares solution, the A and B matrices will be

$$A = \begin{bmatrix} 0.90 & -0.05 \\ 0.06 & 0.75 \end{bmatrix} \quad B = \begin{bmatrix} 1.09 & 0.15 \\ -0.02 & 0.38 \end{bmatrix}$$

We can observe that the A and B matrices change substantially after removing the outlier. This is because Least-Squares averages the squared difference of all data-points which is sensitive to outliers.

4 Open-Loop Control of SIXT33N

Last time, we learned that the ideal input PWM for running a motor at a target velocity v^* is:

$$u(t) = \frac{v^* + \beta}{\theta}$$

In this problem, we will extend our analysis from one motor to a two-motor car system and evaluate how well our open-loop control scheme does.

$$v_L(t) = d_L(t+1) - d_L(t) = \theta_L u_L(t) - \beta_L$$
$$v_R(t) = d_R(t+1) - d_R(t) = \theta_R u_R(t) - \beta_R$$

a) In reality, we need to "kickstart" electric motors with a pulse in order for them to work. That is, we can't go straight from 0 to our desired input signal for $u(t)$, since the motor needs to overcome its initial inertia in order to operate in accordance with our model.

Let us model the pulse as having a width (in timesteps) of t_p. In order to model this phenomenon, we can say that $u(t) = 255$ for $t \in [0, t_p - 1]$[1]. In addition, the car initially (at $t = 0$) hasn't moved, so we can also say $d(0) = 0$.

Firstly, let us examine what happens to d_L and d_R at $t = t_p$, that is, right after the kickstart pulse has passed. **Find $d_L(t_p)$ and $d_R(t_p)$.** (*Hint:* If it helps, try finding $d_L(1)$ and $d_R(1)$ first and then generalizing your result to the t_p case.)

Note: It is very important that you distinguish θ_L and θ_R as the motors we have are liable to vary in their parameters, just as how real resistors vary from their ideal resistance.

Solution

Applying the model directly, we get:

$$\begin{aligned}
d(1) &= d(0) + (255\theta - \beta) \\
d(2) &= d(1) + (255\theta - \beta) = d(0) + (255\theta - \beta) + (255\theta - \beta) = d(0) + 2(255\theta - \beta) \\
d(3) &= d(2) + (255\theta - \beta) = d(0) + 2(255\theta - \beta) + (255\theta - \beta) = d(0) + 3(255\theta - \beta) \\
d(t_p) &= d(0) + t_p(255\theta - \beta) \text{ (by analogy)} \\
d(t_p) &= t_p(255\theta - \beta) \text{ (substitute } d(0))
\end{aligned}$$

Thus we get:

$$d_L(t_p) = t_p(255\theta_L - \beta_L)$$
$$d_R(t_p) = t_p(255\theta_R - \beta_R)$$

b) Let us define $\delta(t) = d_L(t) - d_R(t)$ as the difference in positions between the two wheels. If both wheels of the car are going at the same velocity, then this difference δ should remain constant since no wheel will advance by more ticks than the other. As a result, this will be useful in our analysis and in designing our control schemes.

Find $\delta(t_p)$. For both an ideal car ($\theta_L = \theta_R$ and $\beta_L = \beta_R$) where both motors are perfectly ideal and a non-ideal car ($\theta_L \neq \theta_R$ and $\beta_L \neq \beta_R$), did the car turn when applying the initial pulse?

Note: Since $d(0) = d_L(0) = d_R(0) = 0$, $\delta(0) = 0$.

[1] $x \in [a, b]$ means that x goes from a to b inclusive.

Solution

$$\begin{aligned}
\delta(t_p) &= d_L(t_p) - d_R(t_p) \\
\delta(t_p) &= t_p(255\theta_L - \beta_L) - t_p(255\theta_R - \beta_R) \\
\delta(t_p) &= t_p((255\theta_L - \beta_L) - (255\theta_R - \beta_R)) \\
\delta(t_p) &= t_p(255(\theta_L - \theta_R) - (\beta_L - \beta_R))
\end{aligned}$$

For an ideal car, both the $\theta_L - \theta_R$ and $\beta_L - \beta_R$ terms go to zero, so the pulse made the car go perfectly straight. However, in the non-ideal car, we aren't so lucky, since the car did turn somewhat during the initial pulse.

c) We can still declare victory though, even if the car turns a little bit during the initial pulse (t_p will be very short in lab), so long as the car continues to go straight afterwards when we apply our control scheme; that is, as long as $\delta(t \to \infty)$ converges to a constant value (as opposed to going to $\pm\infty$ or oscillating).

Let's try applying the open-loop control scheme we learned last week to each of the motors independently, and see if our car still goes straight.

$$\begin{aligned}
u_L(t) &= \frac{v^* + \beta_L}{\theta_L} \\
u_R(t) &= \frac{v^* + \beta_R}{\theta_R}
\end{aligned}$$

Let $\delta(t_p) = \delta_0$. Find $\delta(t)$ for $t \geq t_p$ in terms of δ_0. (*Hint:* As in part (a), if it helps you, try finding $\delta(t_p + 1)$, $\delta(t_p + 2)$, etc., and generalizing your result to the $\delta(t)$ case.)

Does $\delta(t \to \infty)$ deviate from δ_0? Why or why not?

Solution

$$\begin{aligned}
\delta(t_p + 1) &= d_L(t_p + 1) - d_R(t_p + 1) \\
&= d_L(t_p) + \theta_L u(t) - \beta_L - (d_R(t_p) + \theta_R u(t) - \beta_R) \\
&= d_L(t_p) + \theta_L u(t) - \beta_L - d_R(t_p) - \theta_R u(t) + \beta_R \\
&= \Big(d_L(t_p) - d_R(t_p)\Big) + \big(\theta_L u(t) - \beta_L\big) - \big(\theta_R u(t) - \beta_R\big) \\
&= \Big(d_L(t_p) - d_R(t_p)\Big) + v^* - v^* \\
&= \Big(d_L(t_p) - d_R(t_p)\Big) \\
&= d(t_p) \\
&= \delta_0 \\
\delta(t) &= \delta_0 \quad \text{(by generalization: every step does not change } \delta\text{)}
\end{aligned}$$

Since we are able to apply just the right amount of input PWM to keep a constant velocity on both wheels, neither wheel gets ahead of the other, so $\delta(t)$ does not change, meaning that the car does not turn.

d) Unfortunately, in real life, it is hard to capture the precise parameters of the car motors like θ and β, and even if we did manage to capture them, they could vary as a function of temperature, time, wheel conditions, battery voltage, etc. In order to model this effect of **model mismatch**, we consider model mismatch terms (such as $\Delta\theta_L$), which reflects the discrepancy between the model parameters and actual parameters.

$$v_L(t) = d_L(t+1) - d_L(t) = (\theta_L + \Delta\theta_L)u_L(t) - (\beta_L + \Delta\beta_L)$$
$$v_R(t) = d_R(t+1) - d_R(t) = (\theta_R + \Delta\theta_R)u_R(t) - (\beta_R + \Delta\beta_R)$$

Let us try applying the open-loop control scheme again to this new system. Note that **no model mismatch terms appear below** – this is intentional since our control scheme is derived from the model parameters for θ and β, not from the actual $\theta + \Delta\theta$, etc.[2]

$$u_L(t) = \frac{v^* + \beta_L}{\theta_L}$$
$$u_R(t) = \frac{v^* + \beta_R}{\theta_R}$$

As before, let $\delta(t_p) = \delta_0$. Find $\delta(t)$ for $t \geq t_p$ in terms of δ_0.

Does $\delta(t \to \infty)$ change from δ_0? Why or why not, and how is it different from the previous case of no model mismatch?

Solution

$$\begin{aligned}
\delta(t_p+1) &= d_L(t_p+1) - d_R(t_p+1)\\
&= \Big(d_L(t_p) + (\theta_L + \Delta\theta_L)u_L(t) - (\beta_L + \Delta\beta_L)\Big) - \Big(d_R(t_p) + (\theta_R + \Delta\theta_R)u_R(t) - (\beta_R + \Delta\beta_R)\Big)\\
&= \Big(d_L(t_p) - d_R(t_p)\Big) + \big((\theta_L + \Delta\theta_L)u_L(t) - (\beta_L + \Delta\beta_L)\big) - \big((\theta_R + \Delta\theta_R)u_R(t) - (\beta_R + \Delta\beta_R)\big)\\
&= \delta(t_p) + \big((\theta_L + \Delta\theta_L)u_L(t) - (\beta_L + \Delta\beta_L)\big) - \big((\theta_R + \Delta\theta_R)u_R(t) - (\beta_R + \Delta\beta_R)\big)\\
&= \delta_0 + \big(\theta_L u_L(t) - \beta_L + \Delta\theta_L u_L(t) - \Delta\beta_L\big) - \big(\theta_R u_R(t) - \beta_R + \Delta\theta_R u_R(t) - \Delta\beta_R\big)\\
&= \delta_0 + \big(v^* + \Delta\theta_L u_L(t) - \Delta\beta_L\big) - \big(v^* + \Delta\theta_R u_R(t) - \Delta\beta_R\big)\\
&= \delta_0 + v^* - v^* + \big(\Delta\theta_L u_L(t) - \Delta\beta_L\big) - \big(\Delta\theta_R u_R(t) - \Delta\beta_R\big)\\
&= \delta_0 + \big(\Delta\theta_L u_L(t) - \Delta\beta_L\big) - \big(\Delta\theta_R u_R(t) - \Delta\beta_R\big)\\
&= \delta_0 + \left(\frac{\Delta\theta_L}{\theta_L}(v^* + \beta_L) - \Delta\beta_L\right) - \left(\frac{\Delta\theta_R}{\theta_R}(v^* + \beta_R) - \Delta\beta_R\right)\\
\delta(t) &= \delta_0 + (t - t_p)\left(\left(\frac{\Delta\theta_L}{\theta_L}(v^* + \beta_L) - \Delta\beta_L\right) - \left(\frac{\Delta\theta_R}{\theta_R}(v^* + \beta_R) - \Delta\beta_R\right)\right) \text{ (generalizing)}
\end{aligned}$$

If there is no model mismatch (i.e., all mismatch terms are zero), then we are back to the same case as last time (all those terms drop out, and δ does not change).

If there is model mismatch, however, we are not so lucky.

[2]Why not just do a better job of capturing the parameters, one may ask? Well, as noted above, the mismatch can vary as a function of an assortment of factors including temperature, time, wheel conditions, battery voltage, and it is not realistic to try to capture the parameters under every possible environment, so it is up to the control designer to ensure that the system can tolerate a reasonable amount of mismatch.

As $t \longrightarrow \infty$, the term $\left(\left(\frac{\Delta\theta_L}{\theta_L}(v^* + \beta_L) - \Delta\beta_L\right) - \left(\frac{\Delta\theta_R}{\theta_R}(v^* + \beta_R) - \Delta\beta_R\right)\right)$ (which is highly unlikely to be zero) causes δ to either steadily increase or decrease, meaning that the car turns steadily more and more.

You may have noticed that open-loop control is insufficient in light of non-idealities and mismatches. Next time, we will analyze a more powerful form of control (closed-loop control) which should be more robust against these kinds of problems.

5 Homework Process and Study Group

Citing sources and collaborators are an important part of life, including being a student!
We also want to understand what resources you find helpful and how much time homework is taking, so we can change things in the future if possible.

a) **What sources (if any) did you use as you worked through the homework?**

b) **If you worked with someone on this homework, who did you work with?**
List names and student ID's. (In case of homework party, you can also just describe the group.)

c) **Roughly how many total hours did you work on this homework?**

d) **Do you have any feedback on this homework assignment?**