6.3100 April 8, 2026 Lecture: Interpreting LQR in the Frequency Domain

April 13, 2026

Outline

Moving freely between state space models (time domain) and transfer functions/frequency responses (frequency domain).

Last lecture: LQR

LQR (for "linear quadratic regulation") refers to an *algorithm* which takes matrices A, B of a *linear* state space input/output model

$$\dot{x}(t) = Ax(t) + Bu(t), \tag{1}$$

as well as matrices Q, R, N defining *quadratic* penalty

$$J(x(\cdot), u(\cdot)) = \int_0^\infty \left[x(t)^T Q x(t) + u(t)^T R u(t) + 2x(t)^T N u(t)\right] dt$$

and produces the matrix K of a *full state linear feedback* ("regulator" in some old-fashioned terminology)

$$u(t) = -Kx(t)$$

which *stabilizes* the state space model (in the sense that all eigenvalues of matrix $A - BK$ have negative real parts) and minimizes the quadratic penalty $J(x(\cdot), u(\cdot))$ computed with a fixed initial condition $x(0) = x_0$. A remarkable property of LQR is that while the value of $J(x(\cdot), u(\cdot))$ does depend on x_0 (in addition to depending on A, B, Q, R, N and K), for fixed A, B, Q, R, N the LQR optimal controller K minimizes J simultaneously for *all* possible x_0.

In a typical use, matrices Q, R, N are chosen in such a way that

$$x(t)^T Q x(t) + u(t)^T R u(t) + 2x(t)^T N u(t) = |Cx(t) + Du(t)|^2$$

is the length squared (i.e., sum of squares of all components) of vector $e(t) = Cx(t) + Du(t)$. Algebraically, this means that

$$Q = C^T C, \quad R = D^T D, \quad N = C^T D$$

for some matrices C, D. This means that LQR minimizes the weighed sum of integrals of squares of zero input responses in the feedback system.

Furthermore, the elements of vector $e(t)$ are frequently just the scaled components of vectors $x(t)$ and $u(t)$. For example, for

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix}, \quad u(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix},$$

one can choose

$$e(t) = \begin{bmatrix} q_1 x_1(t) \\ q_3 x_3(t) \\ r_1 u_1(t) \end{bmatrix},$$

in which case

$$C = \begin{bmatrix} q_1 & 0 & 0 \\ 0 & 0 & q_3 \\ 0 & 0 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ r_1 & 0 \end{bmatrix},$$

and

$$Q = \begin{bmatrix} q_1^2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & q_3^2 \end{bmatrix}, \quad R = \begin{bmatrix} r_1^2 & 0 \\ 0 & 0 \end{bmatrix}, \quad N = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

In Python, you can use the `lqr` method of the `control` module to perform LQR optimization, as in

```
import numpy as np
import control as ct
A = np.array([[0,1], [0,0]])  # A=[[0,1],[0,0]] also accepted, but less convenient
B = np.array([[0], [1]])  # B=[[0], [1]] also OK
Q = np.array([[1,0], [0,0]]) # or Q=[[1,0], [0,0]]
R = 1 # can also do R=np.array([[1]]), but there is no point
K, S, E = ct.lqr(A, B, Q, R)   # assumes N is a zero matrix
Acl = A - B@K   # closed loop "A" matrix: here is where numpy arrays are handy!
```

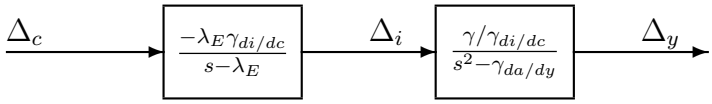
Here S is the symmetric matrix such that, subject to $u(t) = -Kx(t)$,

$$J(x(\cdot), u(\cdot)) = x(0)^T S x(0),$$

and E is the vector of all eigenvalues of $A - BK$.

Example: the Maglev Model

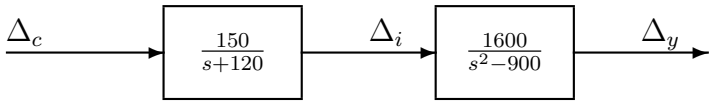
Consider our current model of the Maglev experiment, expressed by the block diagram



Assuming

$$\lambda_E = -120, \quad \gamma = 2000, \quad \gamma_{di/dc} = 1.25, \quad \gamma_{da/dy} = 900,$$

the block diagram becomes



To set this transfer function model for an LQR treatment, several questions need to be answered first:

What is $u(t)$? The only reasonable choice is $u(t) = \Delta_c(t)$.

What is the dimension of $x(t)$? Since the two subsystems have denominators of order 1 and 2, respectively, we expect to need 3=1+2 states

What is the "natural" choice of the components of $x(t)$? Since $\Delta_i(t)$ is practically measured and physically understood as an "inertial" variable, it should be one component of $x(t)$. Similarly, $\Delta_y(t)$ should be another component of $x(t)$. As there are no obvious candidates, the derivative $\dot{\Delta}_y(t)$ would be a good choice, as the "missing" second state corresponding to the subsystem with output $\Delta_y(t)$.

What are A and B? One can choose

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} = \begin{bmatrix} \Delta_i(t) \\ \Delta_y(t) \\ \dot{\Delta}_y(t) \end{bmatrix},$$

in which case

$$\begin{aligned} \dot{x}_1(t) &= -120x_1(t) + 150u(t), \\ \dot{x}_2(t) &= x_3(t), \\ \dot{x}_3(t) &= 1600x_1(t) + 900x_2(t), \end{aligned}$$

i.e.,

$$A = \begin{bmatrix} -120 & 0 & 0 \\ 0 & 0 & 1 \\ 1600 & 900 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 150 \\ 0 \\ 0 \end{bmatrix}.$$

This is by far not the only possibility. For example, with

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} = \begin{bmatrix} \Delta_i(t) \\ \dot{\Delta}_y(t) \\ \Delta_y(t) \end{bmatrix},$$

we have

$$\begin{aligned} \dot{x}_1(t) &= -120x_1(t) + 150u,(t) \\ \dot{x}_2(t) &= 1600x_1(t) + 900x_3(t), \\ \dot{x}_3(t) &= x_2(t), \end{aligned}$$

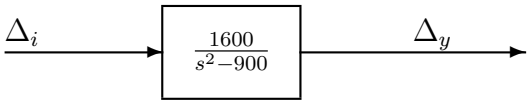
i.e.,

$$A = \begin{bmatrix} -120 & 0 & 0 \\ 1600 & 0 & 900 \\ 0 & 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 150 \\ 0 \\ 0 \end{bmatrix}.$$

Clearly, a single transfer function model corresponds to multiple (actually, infinitely many) state space models, all differing by how the state vector $x(t)$ is defined.

Exposing the States

A bit of inconvenience of using the block with input Δ_i, output Δ_y, and transfer function $\frac{1600}{s^2-900}$ in the last block diagram is that it does not show the $\dot{\Delta}y$ state explicitly. This can be easily fixed by realizing that forming $\frac{1600}{s^2-900}$ uses two pure integrators (transfer function $1/s$ for each), a feedback loop with the coefficient of 900, and an input scaling with coefficient 1600. Indeed, by the "symbolic" interpretation of transfer functions, the differential equation corresponding to the block



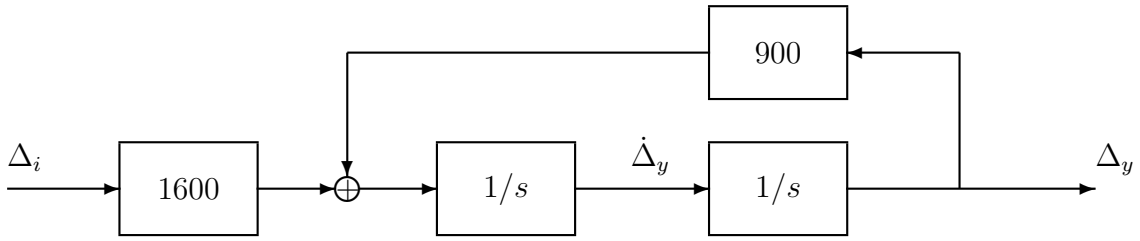
is

$$\ddot{\Delta}_y(t) - 900\Delta_y(t) = 1600\Delta_i(t),$$

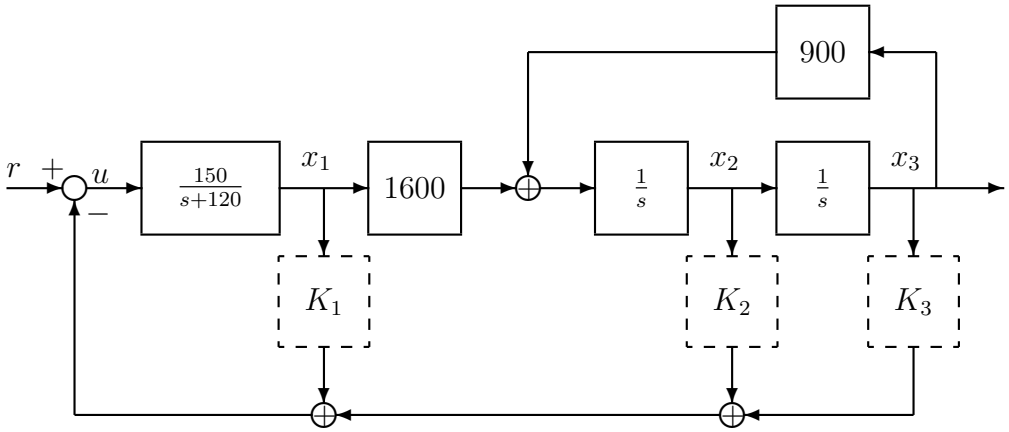
which is equivalent to

$$\begin{aligned} \frac{d}{dt}\Delta_y &= \dot{\Delta}_y(t), \\ \frac{d}{dt}\dot{\Delta}_y &= 1600\Delta_i(t) + 900\Delta_y(t), \end{aligned}$$

which can be depicted by the equivalent block diagram



Ultimately, the block diagram below shows the components of the total state $x(t)$, as well as the feedback connection $u(t) = -Kx(t)$ (to be designed, hence shown with dashed boxes) for the second version of the state space model:



Here the external input r stands for whatever can be added to $-Kx(t)$ to form the command $u(t)$ (for example, $r(t) = K_r y_d(t)$).

Transfer Function of a State Space Model

What is the transfer function corresponding to the state space model

$$\dot{x}(t) = Ax(t) + Bw(t), \quad y(t) = Cx(t) + Dw(t),$$

where $w(t)$ is the input and $y(t)$ s the output?

Replacing the d/dt with s and then removing x from the obtained equation results in

$$\begin{aligned} sx &= Ax + Bw, \quad y = Cx + Dw, \\ (sI - A)x &= Bw, \quad y = Cx + Dw, \\ x &= (sI - A)^{-1}Bw, \quad y = Cx + Dw, \\ y &= [C(sI - A)^{-1}B + D]w, \end{aligned}$$

which shows that the resulting transfer function is

$$H_{w \to y}(s) = C(sI - A)^{-1}B + D.$$

In this derivation, $I = I_n$ is the identity matrix (i.e., the square diagonal matrix with all 1's on the diagonal), of dimensions equal to the dimension n of vector $x(t)$. Hence sI is the diagonal matrix with elements s on the diagonal, and $sx = (sI)x$. Since inverting $sI - A$ involves dividing by its determinant, which is the characteristic polynomial of A, i.e., a polynomial of degree n, you expect $H_{w\to y}(s)$ to be a rational function with denominator of degree n.

Translating Time Domain Costs to Frequency Domain

The last element of the LQR to translate to the language of transfer matrices and frequency responses is the cost, which is originally a sum of integrals of squares of components of zero input responses in the closed loop system. To this end, we compare two possible interpretations of the "closed loop system":

No External Input, but Non-Zero Initial Conditions. The solution $y(t)$ of the state space feedback control model equations

$$\dot{x}(t) = Ax(t) + Bu(t),\ x(0) = B_1, \quad y(t) = Cx(t) + Du(t), \quad u(t) = -Kx(t),$$

where all eigenvalues of the matrix $A - BK$ have negative real part, is given by

$$y(t) = (C - DK)e^{(A-BK)t}B_1.$$

If $y(t)$ is a component of the total cost used in setting up LQR optimization (to produce the stabilizing feedback gain matrix K) then

$$J_y = \int_0^\infty |y(t)|^2 dt$$

is the corresponding contribution of y to the total cost.

External Input replaces Non-Zero Initial Conditions. In the modified model

$$\dot{x}(t) = Ax(t) + Bu(t) + B_1 w(t), \quad y(t) = Cx(t) + Du(t), \quad u(t) = -Kx(t),$$

we "trade off" non-zero initial conditions by inserting a "noise" signal into the state space equations. Note that the vector coefficient B_1 with which the noise is inserted is the same as the initial condition in the first model. In this new setup, consider the transfer function from noise w to output y:

$$H_{w\to y}(s) = (C - DK)(sI - A + BK)^{-1}B_1.$$

Is it possible to express the time domain integral J_y in terms of the transfer function $H_{w\to y}$? The relation is given by the famous *Parceval identity* which states that

$$J_y = \frac{1}{2\pi}\int_{-\infty}^{\infty} |H_{w\to y}(j\omega)|^2 d\omega.$$

While the specific coefficients of the Parceval identity are of little importance to us, the significant message is that by doing LQR optimization we are minimizing the sum of integrals of the absolute value squared of the frequency responses to all cost outputs. In other words, LQR objective has a very clear frequency domain interpretation: "pushing down" all frequency responses to the cost output components.