Full state feedback control

Contents map

developed content units	taxonomy levels
poles placement	u1, e1

prerequisite content units	taxonomy levels
feedback control	u1, e1
state space LTI systems	u1, e1

Main ILO of sub-module "Full state feedback control"

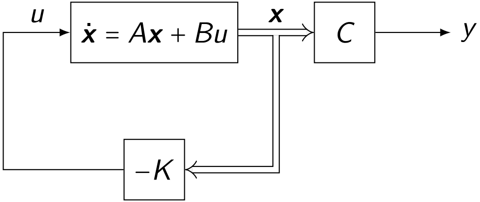
Formulate a state feedback control law $u = -K\boldsymbol{x}$ to modify the closed-loop dynamics of a linear time-invariant system, given matrices A and B in state-space form

Compute the matrix K to place the poles of the closed-loop system at specified locations, using characteristic polynomial matching

Apply the pole placement algorithm to determine the feedback matrix K for a system with A, B in control canonical form, using time-domain specifications

note: the considerations below are the same
for both discrete time and continuous time LTIs

Control-law design for full-state feedback – assumed structure



$$u = -K\mathbf{x} = -\begin{bmatrix} K_1 & \dots & K_n \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$

(estimating $\mathbf{x}$ *from the measurements = later on)*

Finding the control law

$$\begin{cases} \dot{\boldsymbol{x}} &= A\boldsymbol{x} + Bu \\ y &= C\boldsymbol{x} \end{cases} \qquad \text{“ + ”} \qquad u = -K\boldsymbol{x}$$

$$\Downarrow$$

$$\begin{aligned} \dot{\boldsymbol{x}} &= (A - BK)\,\boldsymbol{x} \\ y &= C\boldsymbol{x} \end{aligned}$$

Finding the control law

$$\begin{cases} \dot{\boldsymbol{x}} &= A\boldsymbol{x} + Bu \\ y &= C\boldsymbol{x} \end{cases} \qquad \text{“ + ”} \qquad u = -K\boldsymbol{x}$$

$$\Downarrow$$

$$\begin{aligned} \dot{\boldsymbol{x}} &= (A - BK)\,\boldsymbol{x} \\ y &= C\boldsymbol{x} \end{aligned}$$

Important:

$$BK = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} \begin{bmatrix} K_1 & \dots & K_n \end{bmatrix} = \begin{bmatrix} b_1 K_1 & \cdots & b_1 K_n \\ \vdots & & \vdots \\ b_n K_1 & \cdots & b_n K_n \end{bmatrix}$$

Finding the control law – what are the poles now?

$$\begin{aligned} \dot{\boldsymbol{x}} &= (A - BK)\,\boldsymbol{x} \\ y &= C\boldsymbol{x} \end{aligned} \quad \Rightarrow \quad \det\big(sI - (A - BK)\big) = 0$$

Finding the control law – what are the poles now?

$$\begin{aligned} \dot{\boldsymbol{x}} &= (A - BK)\,\boldsymbol{x} \\ y &= C\boldsymbol{x} \end{aligned} \quad \Rightarrow \quad \det\Big(sI - (A - BK)\Big) = 0$$

choose K *so that the closed-loop poles are where we like*

Finding the control law – what are the poles now?

$$\begin{aligned} \dot{\boldsymbol{x}} &= (A - BK)\,\boldsymbol{x} \\ y &= C\boldsymbol{x} \end{aligned} \qquad \Rightarrow \qquad \det\Big(sI - (A - BK)\Big) = 0$$

choose K *so that the closed-loop poles are where we like*

Poles allocation algorithm

- from time-domain specifications, numerically determine the n desired poles $p_1, \ldots, p_n$

Finding the control law – what are the poles now?

$$\begin{aligned} \dot{\boldsymbol{x}} &= (A - BK)\,\boldsymbol{x} \\ y &= C\boldsymbol{x} \end{aligned} \qquad \Rightarrow \qquad \det\Big(sI - (A - BK)\Big) = 0$$

choose K *so that the closed-loop poles are where we like*

Poles allocation algorithm

- from time-domain specifications, numerically determine the n desired poles $p_1, \ldots, p_n$
- numerically compute the desired denominator of the closed loop TF as $\prod_{i=1}^{n} (s - p_i)$

Finding the control law – what are the poles now?

$$\begin{aligned} \dot{\mathbf{x}} &= (A - BK)\,\mathbf{x} \\ y &= C\mathbf{x} \end{aligned} \qquad \Rightarrow \qquad \det\Big(sI - (A - BK)\Big) = 0$$

choose K so that the closed-loop poles are where we like

Poles allocation algorithm

- from time-domain specifications, numerically determine the n desired poles $p_1, \ldots, p_n$
- numerically compute the desired denominator of the closed loop TF as $\prod_{i=1}^{n} (s - p_i)$
- compute $\det\Big(sI - (A - BK)\Big)$ as a function of $K_1, \ldots, K_n$

Finding the control law – what are the poles now?

$$\begin{aligned} \dot{\boldsymbol{x}} &= (A - BK)\,\boldsymbol{x} \\ y &= C\boldsymbol{x} \end{aligned} \qquad \Rightarrow \qquad \det\Big(sI - (A - BK)\Big) = 0$$

choose K *so that the closed-loop poles are where we like*

Poles allocation algorithm

- from time-domain specifications, numerically determine the n desired poles $p_1, \ldots, p_n$
- numerically compute the desired denominator of the closed loop TF as $\prod_{i=1}^{n} (s - p_i)$
- compute $\det\Big(sI - (A - BK)\Big)$ as a function of $K_1, \ldots, K_n$
- find $K_1, \ldots, K_n$ by equating the two polynomials

Example

Close the loop around the open loop system $\left\{ \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -0.5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u \right.$ so that the closed loop is stable and with raise time not longer than 5 seconds

$$\left(\text{recall:} \quad G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad \rightarrow \quad \text{rise time } t_r = \frac{1.8}{\omega_n} \right)$$

Finding the control law – drawback when A has no special structure

Example:

$$A = \begin{bmatrix} 5 & 1 & 3 & 8 & 3 \\ 7 & 3 & 9 & 6 & 9 \\ 9 & 4 & 4 & 1 & 7 \\ 2 & 2 & 5 & 3 & 6 \\ 1 & 1 & 0 & 1 & 4 \end{bmatrix} \qquad B = \begin{bmatrix} 3 \\ 4 \\ 1 \\ 0 \\ 9 \end{bmatrix}$$

drawback: doing as before is cumbersome

$\downarrow$

is there any alternative way of finding K?

Determinant of a matrix in control canonical form

Let

$$A = \begin{bmatrix} -a_1 & -a_2 & \dots & \dots & -a_n \\ 1 & 0 & \dots & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ & \ddots & \ddots & \ddots & \\ & & 0 & 1 & 0 \end{bmatrix}$$

then

$$\det(sI - A) = s^n + a_1 s^{n-1} + \ldots + a_n$$

Incidentally. . .

$$Y(s) = \frac{b_1 s^{n-1} + \ldots + b_n}{s^n + a_1 s^{n-1} + \ldots + a_n} U(s)$$

$$\mapsto \quad A = \begin{bmatrix} -a_1 & -a_2 & \ldots & \ldots & -a_n \\ 1 & 0 & \ldots & \ldots & 0 \\ 0 & 1 & 0 & \ldots & 0 \\ & \ddots & \ddots & \ddots & \\ & & 0 & 1 & 0 \end{bmatrix}$$

$$\mapsto \quad \det(sI - A) = s^n + a_1 s^{n-1} + \ldots + a_n$$

Finding the control law with (A, B) in control canonical form

$$B = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad \Rightarrow \qquad BK = \begin{bmatrix} K_1 & K_2 & \cdots & K_n \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$

Finding the control law with (A, B) in control canonical form

$$B = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad \Rightarrow \qquad BK = \begin{bmatrix} K_1 & K_2 & \cdots & K_n \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$

so that

$$A - BK = \begin{bmatrix} -a_1 - K_1 & -a_2 - K_2 & \dots & \dots & -a_n - K_n \\ 1 & 0 & \dots & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ & \ddots & \ddots & \ddots & \\ & & 0 & 1 & 0 \end{bmatrix}$$

Finding the control law with (A, B) in control canonical form

$$B = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad \Rightarrow \qquad BK = \begin{bmatrix} K_1 & K_2 & \cdots & K_n \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$

so that

$$A - BK = \begin{bmatrix} -a_1 - K_1 & -a_2 - K_2 & \dots & \dots & -a_n - K_n \\ 1 & 0 & \dots & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ & \ddots & \ddots & \ddots & \\ & & 0 & 1 & 0 \end{bmatrix}$$

and thus the poles of the closed loop system are the roots of

$$\det\Big(sI - (A - BK)\Big) = s^n + (a_1 + K_1)\, s^{n-1} + \dots + (a_n + K_n)$$

Summary (valid also for discrete-time systems!)

$$(A, B) \text{ in control canonical form} + K \text{ generic}$$

$$\Downarrow$$

$$\text{closed loop is } \dot{\boldsymbol{x}} = (A - BK)\boldsymbol{x} = \begin{bmatrix} -a_1 - K_1 & -a_2 - K_2 & \dots & \dots & -a_n - K_n \\ 1 & 0 & \dots & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ & \ddots & \ddots & \ddots & \\ & & 0 & 1 & 0 \end{bmatrix} \boldsymbol{x}$$

Summary (valid also for discrete-time systems!)

(A, B) in control canonical form + K generic

$\Downarrow$

$$\text{closed loop is } \dot{\boldsymbol{x}} = (A - BK)\boldsymbol{x} = \begin{bmatrix} -a_1 - K_1 & -a_2 - K_2 & \dots & \dots & -a_n - K_n \\ 1 & 0 & \dots & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ & \ddots & \ddots & \ddots & \\ & & 0 & 1 & 0 \end{bmatrix} \boldsymbol{x}$$

$\Downarrow$

poles of the closed loop system = roots of

$$\det\Big(sI - (A - BK)\Big) = s^n + (a_1 + K_1)\, s^{n-1} + \ldots + (a_n + K_n)$$

Summary of the algorithm for (A, B) in control canonical form

- from time domain specifications, find the desired poles $p_1, \ldots, p_n$

Summary of the algorithm for (A, B) in control canonical form

- from time domain specifications, find the desired poles $p_1, \ldots, p_n$
- form the desired characteristic polynomial

$$\alpha(s) = \prod_{i=1}^{n} (s - p_i) = s^n + \alpha_1 s^{n-1} + \ldots + \alpha_n$$

Summary of the algorithm for (A, B) in control canonical form

- from time domain specifications, find the desired poles $p_1, \ldots, p_n$
- form the desired characteristic polynomial

$$\alpha(s) = \prod_{i=1}^{n} (s - p_i) = s^n + \alpha_1 s^{n-1} + \ldots + \alpha_n$$

- find K s.t. $\det(sI - A + BK) = \alpha(s)$ by solving

$$\begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix} = \begin{bmatrix} a_1 + K_1 \\ \vdots \\ a_n + K_n \end{bmatrix}$$

Example

Close the loop around the discrete time open loop system $A = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ with sampling period 0.2 seconds so that the closed loop raise time is not longer than 10 seconds

$$\left(\text{recall:} \quad G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad \rightarrow \quad \text{rise time } t_r = \frac{1.8}{\omega_n} \quad \text{but we need to discretize!} \right)$$

Test this out: write a K that makes the discrete time open loop system

$$A = \begin{bmatrix} 1 & 2 \\ 1 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

with sampling period 0.5 seconds have a raise time is not longer than 15 seconds.

Fundamental difference with PIDs

$$\det\left(sI - (A - BK)\right) = s^n + (a_1 + K_1)\, s^{n-1} + \ldots + (a_n + K_n)$$

state-feedback in fully controllable systems allows allocating all the closed loop poles wherever one wants

Caveats

- weak controllability

Caveats

- weak controllability
- the more you move poles, the more you use actuators (risk of saturations!)

Caveats

- weak controllability
- the more you move poles, the more you use actuators (risk of saturations!)
- effect of zeros

Do we actually need to compute the control canonical form?

Do we actually need to compute the control canonical form?

no, there exists the so-called Ackermann's formula

$$K = [0 \; \ldots \; 0 \; 1] \, \mathcal{C}^{-1} \alpha(A)$$

we will though do not cover it - will be in follow up courses!

But how do we select the locations of the poles?

Strategies:

- *in this course,* dominant second-order poles approximations
- *in follow up courses,* very many other ones!

Summarizing

Formulate a state feedback control law $u = -K\boldsymbol{x}$ to modify the closed-loop dynamics of a linear time-invariant system, given matrices A and B in state-space form

Compute the matrix K to place the poles of the closed-loop system at specified locations, using characteristic polynomial matching

Apply the pole placement algorithm to determine the feedback matrix K for a system with A, B in control canonical form, using time-domain specifications

Most important python code for this sub-module

control (Python Control Systems Library)

main functions:

- `acker` (Ackermann's method)
- `place` (robust pole placement)

Question 1

What is the primary advantage of state feedback control with pole placement compared to PID control?

Potential answers:

I: PID control is always more stable than state feedback.
II: State feedback allows arbitrary placement of all closed-loop poles when the system is fully controllable.
III: State feedback does not require knowledge of the system's state variables.
IV: PID control can achieve faster response times than state feedback.
V: I do not know

Question 2

Why is the control canonical form particularly useful for pole placement problems?

Potential answers:

I: It makes the system matrix A diagonal.
II: It eliminates all zeros from the transfer function.
III: The coefficients of the characteristic polynomial appear directly in the first row of A.
IV: It guarantees that the system will be observable.
V: I do not know

Question 3

What is a major practical limitation of aggressive pole placement through state feedback?

Potential answers:

I: It makes the system uncontrollable.
II: It may require large control inputs that could lead to actuator saturation.
III: It always makes the system unstable.
IV: It prevents the use of output feedback.
V: I do not know

Question 4

When designing state feedback control, why might we choose poles with dominant second-order characteristics?

Potential answers:

I: Because higher-order systems cannot be controlled effectively.
II: Because it eliminates all zeros from the system.
III: Because it allows us to approximate the response using familiar second-order performance measures.
IV: Because it guarantees minimum-phase behavior.
V: I do not know

Recap of sub-module "Full state feedback control"

- full state feedback enables placing the poles wherever one wants
- with respect to PID it has more flexibility
- this comes with the cost of having a sufficiently accurate model (and that the model can be written in control canonical form, something that is not always guaranteed!)

?