EE16B - Fall'16 - Lecture 7B Notes[1]

Murat Arcak

13 October 2016

Outputs, Observability, and Observers

In applications we may not have measurements of all state variables, but only an "output" vector

$$\vec{y}(t) = C\vec{x}(t).$$

If we have n states and p outputs, then C is a $p \times n$ matrix. For example, if we measure only the ith state variable, $y(t) = x_i(t)$, then C has a single row that consists of the ith unit vector.

Thus we augment our state model as

$$\begin{aligned} \vec{x}(t+1) &= A\vec{x}(t) + B\vec{u}(t) \\ \vec{y}(t) &= C\vec{x}(t). \end{aligned} \tag{1}$$

Then an important question is, *if we only monitor the output $\vec{y}(t)$ can we infer the full state $\vec{x}(t)$ with the help of this model?* If the answer is yes, we say that the system is *observable*.

Observability is equivalent to the ability to determine the initial state $\vec{x}(0)$ from a set of measurements $\vec{y}(0), \vec{y}(1), \ldots, \vec{y}(t)$. This is because, if we can determine $\vec{x}(0)$, then we can use the explicit solution of the state equation studied in previous lectures to find $\vec{x}(t)$.

To see how we may determine $\vec{x}(0)$ from $\vec{y}(0), \vec{y}(1), \ldots, \vec{y}(t)$, assume for now $u(t) = 0$ for all t and note that

$$\begin{aligned} \vec{y}(0) &= C\vec{x}(0) \\ \vec{y}(1) &= C\vec{x}(1) = CA\vec{x}(0) \\ &\vdots \\ \vec{y}(t) &= C\vec{x}(t) = CA^t\vec{x}(0) \end{aligned} \tag{2}$$

or, equivalently,

$$\begin{bmatrix} \vec{y}(0) \\ \vec{y}(1) \\ \vdots \\ \vec{y}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} C \\ CA \\ \vdots \\ CA^t \end{bmatrix}}_{\triangleq O_t} \vec{x}(0). \tag{3}$$

To *uniquely* determine $\vec{x}(0)$, which has n entries, we need the matrix O_t to have n linearly independent rows so that its null space is $\{0\}$.

It follows from an argument similar to the one we made for controllability that, if O_t doesn't have n independent rows at $t = n-1$, then it never will[2] for $t \geq n$.

[2] Try to prove this claim by adapting our argument for the columns of R_t in controllability to the rows of O_t.

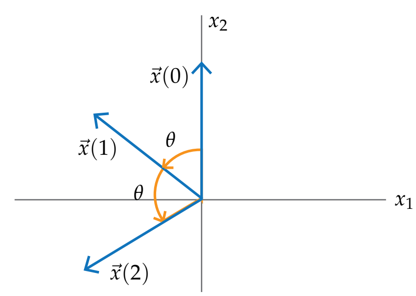
Thus, for observability, we need O_{n-1} to have n linearly independent rows, that is rank $= n$:

$$\text{Observability} \quad \Leftrightarrow \quad \text{rank} \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} = n.$$

Although we assumed $u(t) = 0$ for all t above, adding inputs does not change this observability condition. The only change in this case is that the right hand side of (3) must be augmented with another term that depends on the history of the input $\vec{u}(0), \cdots, \vec{u}(t-1)$. But this term is known, since we know the control inputs we have applied, and can be subtracted from both sides of the equation. The rest of the arguments above are therefore unchanged.

Example: Consider the second order system

$$\vec{x}(t+1) = \underbrace{\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}}_{A} \vec{x}(t) \tag{4}$$



where the matrix A rotates the state vector by an angle of θ at each time instant, as depicted on the right. With the output

$$y(t) = x_1(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \vec{x}(t) \tag{5}$$

we get

$$\begin{bmatrix} C \\ CA \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \cos\theta & -\sin\theta \end{bmatrix}$$

which has rank $= n = 2$ when $\sin\theta \neq 0$, and rank $= 1$ if $\sin\theta = 0$. Thus, we lose observability for $\theta = n\pi$, $n = 1, 2$.

As an illustration, suppose $\theta = \pi$ and $y(t) = x_1(t) = 0$ for each t. Then we can infer that $\vec{x}(0)$ points in the vertical direction, and $\vec{x}(t)$ oscillates back and forth between the positive and negative vertical axes. However, we have no information about the magnitude of this oscillation and, thus, can't determine $\vec{x}(0)$.

Observer Design

An *observer* is an algorithm that estimates the full state $\vec{x}(t)$ from measurements of $\vec{y}(t)$. Observers complement state feedback controllers by providing estimates for states that are not available for measurement.

A straightforward algorithm is to use the past n measurements at each time t to find $\vec{x}(t-n)$, and to obtain $\vec{x}(t)$ from $\vec{x}(t-n)$ using the solution of the system equations. As an illustration, for a single-output system with $u(t) = 0$ for all t, we obtain

$$\begin{bmatrix} y(t-n) \\ y(t-n+1) \\ \vdots \\ y(t-1) \end{bmatrix} = \underbrace{\begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}}_{=O} \vec{x}(t-n) \tag{6}$$

similarly to (2). Since O is invertible by observability, we get

$$\vec{x}(t-n) = O^{-1} \begin{bmatrix} y(t-n) \\ \vdots \\ y(t-1) \end{bmatrix}$$

from which we can calculate the present state using

$$\vec{x}(t) = A^n \vec{x}(t-n) = A^n O^{-1} \begin{bmatrix} y(t-n) \\ \vdots \\ y(t-1) \end{bmatrix}. \tag{7}$$

<u>Example:</u> Consider again the system (4)-(5). If we take $\theta = \pi/2$, then

$$A = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \quad \Rightarrow \quad A^2 = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$$

and

$$O = \begin{bmatrix} C \\ CA \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad \Rightarrow \quad O^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

Then (7) becomes

$$\vec{x}(t) = A^2 O^{-1} \begin{bmatrix} y(t-2) \\ y(t-1) \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y(t-2) \\ y(t-1) \end{bmatrix} = \begin{bmatrix} -y(t-2) \\ y(t-1) \end{bmatrix}$$

which reconstructs the present state from the past two output values.

□

A shortcoming of the algorithm above is that it injects the measurements $y(t-n), \ldots, y(t-1)$ into the state estimates abruptly, which is not desirable when these measurements are noisy and inaccurate.

We now introduce an observer that allows us to inject the outputs into the state estimates more judiciously. The estimates are denoted by $\hat{x}(t)$ and are updated at each time according to

$$\hat{x}(t+1) = A\hat{x}(t) + L(C\hat{x}(t) - \vec{y}(t)) + B\vec{u}(t) \qquad (8)$$

where L is a $n \times p$ matrix to be designed.

Unlike (1) which describes a physical system, (8) is a computational algorithm that produces state estimates as shown in the block diagram below. In the next lecture we will discuss how L should be designed to guarantee convergence of $\hat{x}(t)$ to the correct state $x(t)$.