16-299 Lecture 3: The State Space Approach

Continuous Time State Space Models

Our goal is to write the dynamics of linear systems using matrix/vector equations, so we can later use the tools of linear algebra. Let's see how to do this with a rock: $force = mass * acceleration$. We start with the 2nd order differential equation in continuous time:

$$\ddot{\theta} = force/m \tag{1}$$

where θ is the position of a mass m.

To put this in continuous time state space form we need to define a state vector:

$$\mathbf{x} = \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix} \tag{2}$$

and a command/action vector

$$\mathbf{u} = \begin{pmatrix} force \end{pmatrix} \tag{3}$$

We can then write a matrix equation:

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{\theta} \\ \ddot{\theta} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix} + \begin{pmatrix} 0 \\ 1/m \end{pmatrix} \begin{pmatrix} force \end{pmatrix} = \mathbf{A}_c\mathbf{x} + \mathbf{B}_c\mathbf{u} \tag{4}$$

with

$$\mathbf{A}_c = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \tag{5}$$

and

$$\mathbf{B}_c = \begin{pmatrix} 0 \\ 1/m \end{pmatrix} \tag{6}$$

We are going to use the matrices $\mathbf{A}_c$ and $\mathbf{B}_c$ to define the dynamics of the system we are interested in when we use Matlab, for example.

Now let's do the spring/mass/damper system:

$$\ddot{\theta} = \frac{force}{m} - \frac{b\dot{\theta}}{m} - \frac{k\theta}{m} \tag{7}$$

where θ is the position of a mass m.

Using the same state and action vectors we can write the matrix equation:

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{\theta} \\ \ddot{\theta} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -k/m & -b/m \end{pmatrix} \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix} + \begin{pmatrix} 0 \\ 1/m \end{pmatrix} (force) = \mathbf{A}_c\mathbf{x} + \mathbf{B}_c\mathbf{u} \quad (8)$$

with

$$\mathbf{A}_c = \begin{pmatrix} 0 & 1 \\ -k/m & -b/m \end{pmatrix} \quad (9)$$

and

$$\mathbf{B}_c = \begin{pmatrix} 0 \\ 1/m \end{pmatrix} \quad (10)$$

Another example: Actuator Dynamics

Many actuators don't produce a force directly. Instead, the command generates a rate of change of force $\mathbf{u} = \dot{\mathbf{f}}$. Electric motors have inductance, and the voltage out of a power amplifier generates a rate of change of the current. Hydraulic and pneumatic systems have valves, and the opening of a valve is associated with the rate of change of pressure. For such a system, the equations for the controlled spring-mass-damper system are:

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{\theta} \\ \ddot{\theta} \\ \dot{\mathrm{f}} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ -k/m & -b/m & 1/m \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \theta \\ \dot{\theta} \\ \mathrm{f} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \begin{pmatrix} u \end{pmatrix} = \mathbf{A}_c\mathbf{x} + \mathbf{B}_c\mathbf{u} \quad (11)$$

with

$$\mathbf{A}_c = \begin{pmatrix} 0 & 1 & 0 \\ -k/m & -b/m & 1/m \\ 0 & 0 & 0 \end{pmatrix} \quad (12)$$

and

$$\mathbf{B}_c = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad (13)$$

Discrete Time State Space Models

Before computers were used for control, the controllers were made out of electric circuits, and were essentially analog computers (which operate in continuous time). As computers were used for control, discrete time representations became useful, because 1) computers sample sensors to generate control commands, and 2) system identification (making models) uses samples. We want to convert the continuous time state space model into a discrete time state space model. We start with:

$$\dot{\mathbf{x}} = \mathbf{A}_c\mathbf{x} + \mathbf{B}_c\mathbf{u} \tag{14}$$

We want:

$$\mathbf{x}_{i+1} = \mathbf{A}_d\mathbf{x}_i + \mathbf{B}_d\mathbf{u}_i \tag{15}$$

where the i subscripts indicate the ith sample. We know the approximation

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \Delta\dot{\mathbf{x}}_i \tag{16}$$

where Δ is a small time step. So:

$$\dot{\mathbf{x}} = \frac{(\mathbf{A}_d - \mathbf{1})}{\Delta}\mathbf{x}_i + \frac{\mathbf{B}_d}{\Delta}\mathbf{u}_i \tag{17}$$

So:

$$\mathbf{A}_d = \Delta\mathbf{A}_c + \mathbf{1} \tag{18}$$

and

$$\mathbf{B}_d = \Delta\mathbf{B}_c \tag{19}$$

Does this make sense as the time step Δ goes to zero?

A rock in discrete time

We use the same state and action as in continuous time.

$$\mathbf{x} = \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix} \tag{20}$$

and a command/action vector

$$\mathbf{u} = \begin{pmatrix} force \end{pmatrix} \tag{21}$$

We can then write a matrix equation:

We can then write a matrix equation:

$$\mathbf{x}_{\text{next}} = \begin{pmatrix} \theta_{\text{next}} \\ \dot{\theta}_{\text{next}} \end{pmatrix} = \begin{pmatrix} 1 & \Delta \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix} + \begin{pmatrix} 0 \\ \Delta/m \end{pmatrix} (force) = \mathbf{A}_d \mathbf{x} + \mathbf{B}_d \mathbf{u} \quad (22)$$

with

$$\mathbf{A}_d = \begin{pmatrix} 1 & \Delta \\ 0 & 1 \end{pmatrix} \quad (23)$$

and

$$\mathbf{B}_d = \begin{pmatrix} 0 \\ \Delta/m \end{pmatrix} \quad (24)$$

Now let's do the spring/mass/damper system:

$$\ddot{\theta} = \frac{force}{m} - \frac{b\dot{\theta}}{m} - \frac{k\theta}{m} \tag{25}$$

where θ is the position of a mass m.

Using the same state and action vectors we can write the matrix equation:

$$\mathbf{x}_{\text{next}} = \begin{pmatrix} \theta_{\text{next}} \\ \dot{\theta}_{\text{next}} \end{pmatrix} = \begin{pmatrix} 0 & \Delta \\ -k\Delta/m & -b\Delta/m \end{pmatrix} \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix} + \begin{pmatrix} 0 \\ \Delta/m \end{pmatrix} (force) = \mathbf{A}_d\mathbf{x} + \mathbf{B}_d\mathbf{u} \tag{26}$$

with

$$\mathbf{A}_d = \begin{pmatrix} 1 & \Delta \\ -k\Delta/m & 1 - b\Delta/m \end{pmatrix} \tag{27}$$

and

$$\mathbf{B}_d = \begin{pmatrix} 0 \\ \Delta/m \end{pmatrix} \tag{28}$$

Another example: Actuator Dynamics

Add $\mathbf{u} = \dot{\mathbf{f}}$ to the previous spring-mass-damper system.

$$\mathbf{x}_{\text{next}} = \begin{pmatrix} \theta_{\text{next}} \\ \dot{\theta}_{\text{next}} \\ \text{f}_{\text{next}} \end{pmatrix} = \begin{pmatrix} 1 & \Delta & 0 \\ -k\Delta/m & 1 - b\Delta/m & \Delta/m \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \theta \\ \dot{\theta} \\ \text{f} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \Delta \end{pmatrix} \begin{pmatrix} u \end{pmatrix} = \mathbf{A}_d\mathbf{x} + \mathbf{B}_d\mathbf{u} \tag{29}$$

with

$$\mathbf{A}_d = \begin{pmatrix} 0 & \Delta & 0 \\ -k\Delta/m & -b\Delta/m & \Delta/m \\ 0 & 0 & 0 \end{pmatrix} \tag{30}$$

and

$$\mathbf{B}_d = \begin{pmatrix} 0 \\ 0 \\ \Delta \end{pmatrix} \tag{31}$$

What Can We Do With State Space Models?

An example of the usefulness and generality of state space models is checking whether a system is stable. The simplest case is to consider a discrete time system with no inputs:

$$\mathbf{x}_{i+1} = \mathbf{A}_d \mathbf{x}_i \tag{32}$$

Note that

$$\mathbf{x}_{i+2} = \mathbf{A}_d \mathbf{A}_d \mathbf{x}_i \tag{33}$$

and

$$\mathbf{x}_{i+n} = \mathbf{A}_d^n \mathbf{x}_i \tag{34}$$

We want to know if $\mathbf{x}$ shrinks to zero or blows up as n goes to infinity.

This question can be answered using techniques from linear algebra. The question of whether $\mathbf{A}_d^n$ shrinks the state to zero or causes it to blow up boils down to whether $\mathbf{A}_d$ increases or decreases the length or "magnitude" of the state vector $\mathbf{x}$. The matrix $\mathbf{A}_d$ can be decomposed using the singular value decomposition (SVD):

$$\mathbf{A}_d = \mathbf{U}\mathbf{D}\mathbf{V}^{\mathrm{T}} \tag{35}$$

where $\mathbf{U}$ and $\mathbf{V}$ are rotations (they do not change the length of a vector) and $\mathbf{D}$ is a diagonal matrix:

$$\mathbf{D} = \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{pmatrix} \tag{36}$$

$\mathbf{A}_d$ shrinks the state vector if the magnitudes (absolute values) of all the singular values λ_i are less than one, and grows the state vector if any $|\lambda_i|$ are bigger than 1. This is a stability test for discrete time linear systems. $max(\lambda_i)$ tells us how fast errors are reduced.

Adding feedback

Suppose we had a linear control law/policy

$$u = -\mathbf{K}\mathbf{x} \tag{37}$$

where for the mass/spring/damper system

$$\mathbf{K} = \begin{pmatrix} k & b \end{pmatrix} \tag{38}$$

Inserting that policy into

$$\mathbf{x}_{i+1} = \mathbf{A}_d\mathbf{x}_i + \mathbf{B}_d\mathbf{u}_i \tag{39}$$

Gives

$$\mathbf{x}_{i+1} = (\mathbf{A}_d - \mathbf{B}_d\mathbf{K})\mathbf{x}_i = \mathbf{A}_{fb}\mathbf{x}_i \tag{40}$$

We now have a new $\mathbf{A}$ matrix, $\mathbf{A}_{fb}$, and can analyze the singular values of that for stability and also how fast errors are reduced.

Controllability

An interesting question is whether we can drive the system anywhere, with a sequence of commands $\mathbf{u}_0, \mathbf{u}_1, ...\mathbf{u}_{n-1}$. The state space equations allow us to write this out in a reasonable way. For n = 2:

$$\mathbf{x}_2 = \mathbf{A}_d\mathbf{x}_1 + \mathbf{B}_d\mathbf{u}_1 = \mathbf{A}_d(\mathbf{A}_d\mathbf{x}_0 + \mathbf{B}_d\mathbf{u}_0) + \mathbf{B}_d\mathbf{u}_1 \tag{41}$$

Let's assume $\mathbf{x}_0 = \mathbf{0}$, since if we can reach anywhere from $\mathbf{x}_0 = \mathbf{0}$, we can do it from anywhere. Now the equation is:

$$\mathbf{x}_2 = \mathbf{A}_d\mathbf{B}_d\mathbf{u}_0 + \mathbf{B}_d\mathbf{u}_1 \tag{42}$$

For n = 3:

$$\begin{aligned} \mathbf{x}_3 &= \mathbf{A}_d\mathbf{x}_2 + \mathbf{B}_d\mathbf{u}_2 \\ &= \mathbf{A}_d(\mathbf{A}_d\mathbf{B}_d\mathbf{u}_0 + \mathbf{B}_d\mathbf{u}_1) + \mathbf{B}_d\mathbf{u}_2 \\ &= \mathbf{A}_d^2\mathbf{B}_d\mathbf{u}_0 + \mathbf{A}_d\mathbf{B}_d\mathbf{u}_1 + \mathbf{B}_d\mathbf{u}_2 \end{aligned} \tag{43}$$

For n steps:

$$\mathbf{x}_n = \sum_{k=0}^{k=n-1} \mathbf{A}_d^k\mathbf{B}_d\mathbf{u}_{n-k-1} \tag{44}$$

We can write this another way:

$$\begin{aligned} \mathbf{x}_n &= [\mathbf{A}_d^{n-1}\mathbf{B}_d + \ldots + \mathbf{A}_d\mathbf{B}_d + \mathbf{B}_d][\mathbf{u}_0^{\mathrm{T}} + \cdots + \mathbf{u}_{n-2}^{\mathrm{T}} + \mathbf{u}_{n-1}^{\mathrm{T}}]^{\mathrm{T}} \\ &= \mathcal{C} \end{aligned} \tag{45}$$

If $\mathcal{C}$ is full rank (no singular values are zero), you can reach anywhere in n steps. This is called controllability (the system is "controllable") if n is less than or equal to the dimensionality of the state vector when $\mathcal{C}$ attains full rank.

Convince yourself that $n > N$ where N is the dimensionality of the state vector won't give you more control, or make an uncontrollable system controllable.

A system is "stabilizable" if the directions you can't control in go to zero because they are stable directions.

NEED SOME MATLAB EXAMPLES.