16-299 Lecture 6: State Estimation, Observers, and Kalman Filters

This lecture focuses on discrete time models and design techniques.

What's wrong with full state feedback?

As we discussed when talking about full state feedback, there are some issues we need to address.

1. We don't measure all the states. State estimation, and in particular the Kalman Filter, estimate the states that are not measured. Another approach is to only use the states that are measured for feedback control. This is known as output feedback.

2. Our models are inaccurate and full state feedback doesn't work as expected. Robust control design addresses this problem. However, the basic solution is "Don't be greedy." Don't ask for a more aggressive controller than you really need.

3. There is sensor noise. State estimation, and in particular the Kalman Filter, tries to filter out sensor noise.

Modeling partial state feedback

In order to do model-based design, we need to model the situation of only partial state feedback. We use the same model of the dynamics:

$$\mathbf{x}_{\text{next}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{1}$$

and we add a model of the measurements ("observables"):

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u} \tag{2}$$

$\mathbf{C}$ is a matrix indicating how the measurements $\mathbf{y}$ depend on the states $\mathbf{x}$, and $\mathbf{D}$ is a matrix indicating how the measurements depend on actuator commands directly. For most systems $\mathbf{D}$ is $\mathbf{0}$, so we will assume that unless indicated otherwise.

An observer estimates the missing states

A controller is used to control a system. An observer is used to observe a system, producing estimates of the missing states. The estimated state $\hat{\mathbf{x}}$ is known in AI as a belief state. Feedback is used to track the true state. At each time step we can measure the error in the measurement space $(\mathbf{C}\hat{\mathbf{x}} - \mathbf{y})$, and use that to correct the belief state. In addition, the belief state should have the same dynamics as the true state:

$$\hat{\mathbf{x}}_{\text{next}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} - \mathbf{K}_o(\mathbf{C}\hat{\mathbf{x}} - \mathbf{y}) \tag{3}$$

The above equation is the generic form for observers. Different observers are generated by designing the observer feedback gains $\mathbf{K}_o$ in different ways.

What are the dynamics of the state estimation error $\mathbf{e} = \hat{\mathbf{x}} - \mathbf{x}$? Subtracting the dynamics of the true state, $\mathbf{x}_{\text{next}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$ from the above equation, we get:

$$\hat{\mathbf{x}}_{\text{next}} - \mathbf{x}_{\text{next}} = \mathbf{A}\hat{\mathbf{x}} - \mathbf{A}\mathbf{x} - \mathbf{K}_o\mathbf{C}(\hat{\mathbf{x}} - \mathbf{x}) \tag{4}$$

so

$$\mathbf{e}_{\text{next}} = (\mathbf{A} - \mathbf{K}_o\mathbf{C})\mathbf{e} \tag{5}$$

Designing the observer involves choosing the observer gain $\mathbf{K}_o$ so that the dynamics of the error are desirable. This turns out to be an identical problem as choosing the controller gain so that the dynamics of the state are desirable, and the same control design techniques as eigenvalue/pole placement and LQR are useful, as we shall see.

Observability

A discrete time system is observable if we can reconstruct the initial state $\mathbf{x}_0$ from the measurements $(\mathbf{y}_0, \mathbf{y}_1, \cdots, \mathbf{y}_n)$ for some n, assuming all the commands $\mathbf{u}_i$ are zero. We know that $\mathbf{y}_0 = \mathbf{C}\mathbf{x}_0$, $\mathbf{y}_1 = \mathbf{C}\mathbf{A}\mathbf{x}_0$, $\mathbf{y}_2 = \mathbf{C}\mathbf{A}\mathbf{A}\mathbf{x}_0$, $\cdots$, $\mathbf{y}_n = \mathbf{C}\mathbf{A}^n\mathbf{x}_0$. Stacking this up as a big set of equations:

$$\begin{pmatrix} \mathbf{y}_0 \\ \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_n \end{pmatrix} = \begin{pmatrix} \mathbf{C} \\ \mathbf{CA} \\ \mathbf{CA}^2 \\ \vdots \\ \mathbf{CA}^n \end{pmatrix} \mathbf{x}_0 = \mathcal{O}\mathbf{x}_0 \tag{6}$$

These equations can be solved if the matrix $\mathcal{O}$ has rank N, the dimensionality of $\mathbf{x}$. In this case the system is observable. Convince yourself that only a maximum of N observations $\mathbf{y}_i$ are needed for this test, since more observations won't increase the rank of $\mathcal{O}$ if it is not full rank after N observations.

Observability and Controllability are duals

The matrix to test for controllability (from a previous lecture) is

$$\mathcal{C} = [\mathbf{A}^{N-1}\mathbf{B} \ \cdots \ \mathbf{AB} \ \mathbf{B}] \tag{7}$$

If we reverse the order of $\mathcal{O}$ (reversing time), and take the transpose of that, we end up with a matrix with a similar form:

$$\mathrm{ReverseTime}(\mathcal{O})^{\mathrm{T}} = [(\mathbf{A}^{\mathrm{T}})^{N-1}\mathbf{C}^{\mathrm{T}} \ \cdots \ \mathbf{A}^{\mathrm{T}}\mathbf{C}^{\mathrm{T}} \ \mathbf{C}^{\mathrm{T}}] \tag{8}$$

Neither operation affects matrix rank and thus observability. In going from $\mathcal{C}$ to $\mathcal{O}$, $\mathbf{A}$ is replaced by $\mathbf{A}^{\mathrm{T}}$, and $\mathbf{B}$ is replaced by $\mathbf{C}^{\mathrm{T}}$. The pair $(\mathbf{A}, \mathbf{C})$ is observable if the pair $(\mathbf{A}^{\mathrm{T}}, \mathbf{C}^{\mathrm{T}})$ is controllable. This enables the dynamics of an observer to be designed using tools for designing controllers like eigenvalue/pole placement and LQR by swapping $\mathbf{A}^{\mathrm{T}}$ for $\mathbf{A}$ and $\mathbf{C}^{\mathrm{T}}$ for $\mathbf{B}$.

The Kalman Filter

The Kalman Filter designs observer gains $\mathbf{K}_o$ based on a probabilistic model of sensor noise and process noise (deviations from the dynamics due to perturbations and/or modeling error). Minimizing the variance of the state estimation error (a form of optimization) drives the design. Probabilistic approaches to anything are known as Bayesian approaches.

We will use some facts about random variables. George Kantor's notes on Kalman Filtering `http://www.cs.cmu.edu/~cga/controls-intro/kantor/16_299_Kalman_Filter.pdf` have a nice review of Gaussian Random variables.

Fact 1: A Gaussian random vector is fully characterized by its mean (first moment) and variance (2nd moment). A compact notation is $\mathbf{x} \sim \mathrm{N}(\mathrm{mean}, \mathrm{variance})$.

Fact 2: For any random vector $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \Sigma)$, $\mathbf{Ax} \sim \mathrm{N}(\mathbf{Am}, \mathbf{A}\Sigma\mathbf{A}^{\mathrm{T}})$

Fact 3: If any two independent random vectors ($\mathbf{x}_1 \sim \mathrm{N}(\mathbf{m}_1, \Sigma_1)$ and $\mathbf{x}_2 \sim \mathrm{N}(\mathbf{m}_2, \Sigma_2)$), are added, the result is $\mathrm{N}(\mathbf{m}_1 + \mathbf{m}_2, \Sigma_1 + \Sigma_2)$

Fact 4: If you are given two predictions or belief states about a random variable $\mathbf{x}$, and the accuracy of these predications is $\hat{\mathbf{x}}_1 \sim \mathrm{N}(\mathbf{m}_1, \Sigma_1)$ (the belief of expert 1) and $\hat{\mathbf{x}}_2 \sim \mathrm{N}(\mathbf{m}_2, \Sigma_2)$ (the belief of expert 2), your best linear unbiased estimate (BLUE) of $\mathbf{x}$ is $\mathbf{W}_1\mathbf{m}_1 + \mathbf{W}_2\mathbf{m}_2$, with $\mathbf{W}_1 = \Sigma_2(\Sigma_1 + \Sigma_2)^{-1}$ and $\mathbf{W}_2 = \Sigma_1(\Sigma_1 + \Sigma_2)^{-1}$. The variance of this estimate is:

$$\Sigma_2(\Sigma_1 + \Sigma_2)^{-1}\Sigma_1\Sigma_2(\Sigma_1 + \Sigma_2)^{-1} + \Sigma_1(\Sigma_1 + \Sigma_2)^{-1}\Sigma_2(\Sigma_1 + \Sigma_2)^{-1}\Sigma_1 \tag{9}$$

What a mess! However, all the above matrices are symmetric, so we can reorder them and get

$$\Sigma_1\Sigma_2(\Sigma_1 + \Sigma_2)^{-1} \tag{10}$$

A useful way to express the same thing (since $\mathbf{W}_1 = (1 - \mathbf{W}_2)$) that we will use in the derivation of the Kalman Filter is:

$$\mathbf{m} = \mathbf{m}_1 + \mathbf{W}_2(\mathbf{m}_2 - \mathbf{m}_1) \tag{11}$$

and

$$\Sigma = \Sigma_1 - \Sigma_1(\Sigma_1 + \Sigma_2)^{-1}\Sigma_1 \tag{12}$$

Kalman Filter Derivation: The Prediction Step

The Kalman Filter alternates between predicting the probability distribution of the belief state on the next step (the prediction step), and incorporating an observation (the update step). After a prediction step we have a belief state $\hat{\mathbf{x}} \sim \mathrm{N}(\mathbf{m}^-, \Sigma^-)$ and after a update step we have a belief state $\hat{\mathbf{x}} \sim \mathrm{N}(\mathbf{m}^+, \Sigma^+)$. The superscripts - and + keep track of whether we have incorporated the current measurement or not.

For a nonlinear discrete time system $\mathbf{F}()$, the belief state mean is propagated forward in time just using the nonlinear dynamics:

$$\mathbf{m}^-_{\mathrm{next}} = \mathbf{F}(\mathbf{m}^+, \mathbf{u}) \tag{13}$$

The variance Σ is propagated by linearizing $\mathbf{F}()$ about $\mathbf{m}$:

$$\Sigma^-_{\mathrm{next}} = \mathbf{A}\Sigma^+\mathbf{A}^{\mathrm{T}} + \Sigma_p \tag{14}$$

Σ_{p} is the variance of the process noise. The Gaussian process noise is a perturbation, or a way to model modeling error. Note that $\mathbf{u}$ plays no role in uncertainty propagation, since the commands are known perfectly and the local dynamics are linear.

Kalman Filter Derivation: The Update Step

Let's model sensor noise as additive Gaussian noise with $\mathbf{w} \sim \mathrm{N}(\mathbf{0}, \Sigma_o)$:

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{w} \tag{15}$$

$\widehat{\mathbf{C}\mathbf{x}}$ is a prediction of a measurement. One way to predict it is to use $\mathbf{C}\hat{\mathbf{x}} = \mathbf{C}\mathbf{m}^-$, which has variance $\mathbf{C}\Sigma^-\mathbf{C}^\mathrm{T}$.

Another way to predict it is to use the actual measurement $\mathbf{y}$, which has variance Σ_o.

Now we use Gaussian Fact 4 to combine these predictions. The optimal estimate is:

$$\widehat{\mathbf{C}\mathbf{x}} = \mathbf{W}_p\mathbf{C}\mathbf{m}^- + \mathbf{W}_m\mathbf{y} \tag{16}$$

where the weight on the prediction is

$$\mathbf{W}_p = \Sigma_o(\Sigma_o + \mathbf{C}\Sigma^-\mathbf{C}^\mathrm{T})^{-1} \tag{17}$$

and the weight on the measurement is

$$\mathbf{W}_m = \mathbf{C}\Sigma^-\mathbf{C}^\mathrm{T}(\Sigma_o + \mathbf{C}\Sigma^-\mathbf{C}^\mathrm{T})^{-1} \tag{18}$$

We will use the definition $\mathbf{S} = \Sigma_o + \mathbf{C}\Sigma^-\mathbf{C}^\mathrm{T}$, and the fact that symmetric matrices commute in matrix multiplication to simplify what follows

So the optimal estimate for $\widehat{\mathbf{C}\mathbf{x}}$ is:

$$\begin{aligned} \mathrm{mean}(\widehat{\mathbf{C}\mathbf{x}}) &= (1 - \mathbf{W}_m)\mathbf{C}\mathbf{m}^- + \mathbf{W}_m\mathbf{y} \\ &= \mathbf{C}\mathbf{m}^- - \mathbf{W}_m(\mathbf{C}\mathbf{m}^- - \mathbf{y}) \\ &= \mathbf{C}(\mathbf{m}^- - \Sigma^-\mathbf{C}^\mathrm{T}\mathbf{S}^{-1}(\mathbf{C}\mathbf{m}^- - \mathbf{y})) \end{aligned} \tag{19}$$

Since $\mathbf{C}$ is a constant and the $\mathrm{mean}()$ operation is linear,

$$\begin{aligned} \mathrm{mean}(\hat{\mathbf{x}}) = \mathbf{m}^+ &= \mathbf{m}^- - \Sigma^-\mathbf{C}^\mathrm{T}\mathbf{S}^{-1}(\mathbf{C}\mathbf{m}^- - \mathbf{y}) \\ &= \mathbf{m}^- - \mathbf{K}^*(\mathbf{C}\mathbf{m}^- - \mathbf{y}) \end{aligned} \tag{20}$$

so the optimal Kalman filter gain is $\mathbf{K}^* = \Sigma^-\mathbf{C}^\mathrm{T}\mathbf{S}^{-1}$.

We also need to propagate the variance

$$\mathrm{Var}(\widehat{\mathbf{C}\mathbf{x}}) = \mathbf{C}\Sigma^-\mathbf{C}^\mathrm{T} - \mathbf{C}\Sigma^-\mathbf{C}^\mathrm{T}\mathbf{S}^{-1}\mathbf{C}\Sigma^-\mathbf{C}^\mathrm{T} \tag{21}$$

Peeling off the left $\mathbf{C}$ and right $\mathbf{C}^{\mathrm{T}}$:

$$\mathrm{Var}(\hat{\mathbf{x}}) = \Sigma^{-} - \Sigma^{-}\mathbf{C}^{\mathrm{T}}\mathbf{S}^{-1}\mathbf{C}\Sigma^{-} \tag{22}$$

and substituting in Σ^{+} and $\mathbf{K}^{*}$ gives the update equation for Σ:

$$\Sigma^{+} = \Sigma^{-} - \mathbf{K}^{*}\mathbf{C}\Sigma^{-} \tag{23}$$

We can see that the reduction in variance of the belief state due to the Kalman Filter is $\mathbf{K}^{*}\mathbf{C}\Sigma^{-}$. Interestingly, it is proportional to the variance of the belief state before the update Σ^{-}. It makes sense that when there is no uncertainty before the update, the update can't reduce it further.

Controlling and observing at the same time

Separation principle
differential flatness page 8-25
gain scheduling
MPC
internal model principle 8-25
integral windup
policy optimization for output feedback