CHEM-E7225 Advanced Process Control D — Winter 2026

Lecture 09 – Model Predictive Control: State estimation

Otacilio "Minho" Neto (otacilio.bneto@pm.me)

In this lecture, we consider the task of estimating the state of a partially-observed, noisy, system using measurement data. The focus is on designing a device, known as *state estimator*, capable of reconstructing the state in real time.

1 State estimation / noise filtering

Recall the (partially-observed) linear system subject to additive white Gaussian noise (AWGN) from last lecture,

$$x[n+1] = Ax[n] + w[n], \qquad w[n] \sim \mathcal{N}(0, S_w^2); \tag{1a}$$

$$y[n] = Cx[n] + v[n], \qquad v[n] \sim \mathcal{N}(0, S_v^2), \tag{1b}$$

where we assume no process/measurement drift. Under this stochastic setting, the state $x[\cdot]$ cannot be accessed anymore–it is said to be a *latent* quantity. Nevertheless, we still want to use state-space models for control purposes. Thus, we need some mechanism to *reconstruct the state* based on what we have available: the measurement data $y[\cdot]$.

In this course, the problem is approached via a *state estimator* (Figure 1). The estimator is a device that processes measurement data $y[\cdot]$ to filter out the noise ($w[\cdot]$, $v[\cdot]$) and extrapolate the values of the state $x[\cdot]$, in real time. Formally, it is a device which implements an *estimation policy*

$$\hat{x}[n] = K_e(\mathbf{y}_n^{\text{data}}), \tag{2}$$

where $\hat{x}[n]$ is an estimate of the state $x[n]$ and $\mathbf{y}_n^{\text{data}} = (y[0], \dots, y[n-1], y[n])$ is a *batch of measurement data*. Avoiding arbitrarily large batches, let's limit ourselves to policies which only consider the N_e+1 past measurements,

$$\mathbf{y}_n^{\text{data}} = (\underbrace{y[n-N_e]}_{\mathbf{y}_n^{\text{data}}[0]}, \quad \dots, \quad \underbrace{y[n-1]}_{\mathbf{y}_n^{\text{data}}[N_e-1]}, \quad \underbrace{y[n]}_{\mathbf{y}_n^{\text{data}}[N_e]}),$$

such that $y_n^{\text{data}}[k] = y[n-N_e+k]$. Our task is to design the policy $K_e(\cdot)$ to have $\hat{x}[n] \approx x[n]$, that is, the estimate is *as close as possible to the true state*. While this seems challenging (as don't know the "true value" $x[n]$), our knowledge about the system and the noise/disturbance statistics can be exploited to devise an *optimal* estimator. In the following, we formalize this optimal estimation task as a nonlinear optimization problem.

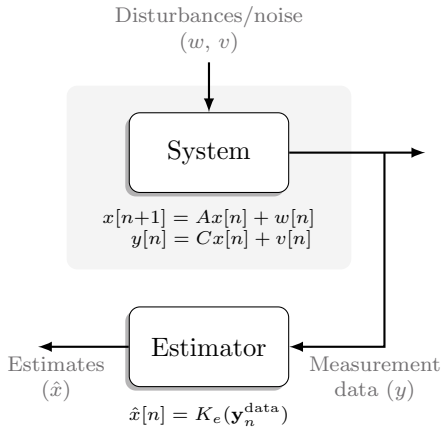

Figure 1. Block diagram of a state estimator.

1.1 Optimal estimation problems

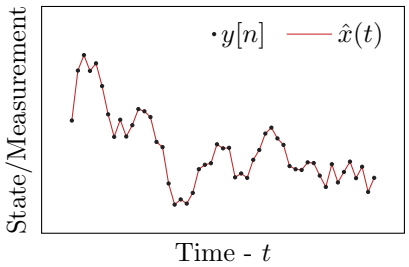

Figure 2. Overfitting the data

Let's momentarily ignore the state dynamics (Eq. 1a). If our measurement model (Eq. 1b) is correct, then the difference between the datapoint $y^{\text{data}}[k]$ and the emission $Cx[k]$ is exactly the measurement noise $v[k]$, that is,

$$v[k] = y^{\text{data}}[k] - C\hat{x}[k]$$

$$\Updownarrow$$

$$\delta_v[k] = S_v^{-1}\left(y^{\text{data}}[k] - C\hat{x}[k]\right)$$

where we used the fact that $v[k] = S_v\delta_v[k]$ for the standard Gaussian noise $\delta_v[k] \sim \mathcal{N}(0, I_{N_y})$. Using this identity, the optimal estimate (in a least-square sense) is the vector $\hat{x}[k]$ which minimizes the *measurement error* $\|\sigma_v[k]\|_2^2$. Unfortunately, by ignoring the state dynamics, there is a risk of *overfitting*: The estimates $\hat{x}[\cdot]$ perfectly matches the measurement data $y_n^{\text{data}}[\cdot]$, without accounting for the fact that they are noisy (Figure 2).

Let's now momentarily ignore the measurement model (Eq. (1b)). If our dynamical model (Eq. 1a) is correct, then the difference between the next state $\hat{x}[k+1]$ and the transition $A\hat{x}[k]$ is exactly the process disturbance $w[k]$, that is,

$$w[k] = \hat{x}[k+1] - A\hat{x}[k]$$
$$\Updownarrow$$
$$\delta_w[k] = S_w^{-1}\left(\hat{x}[k+1] - A\hat{x}[k]\right)$$

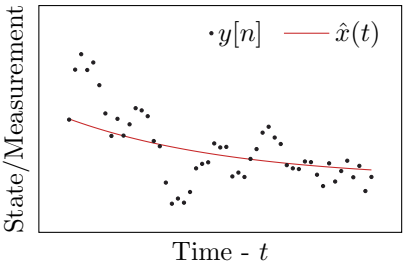

Figure 3. Underfitting the data

where we again used the fact that $w[k] = S_w\delta_w[k]$ for the standard Gaussian noise $\delta_w[k] \sim \mathcal{N}(0, I_{N_x})$. In this case, the optimal estimate (in a least-square sense) is the vector $\hat{x}[k]$ which minimizes the *model error* $\|\sigma_w[k]\|_2^2$. However, ignoring the measurement data leads to *underfitting*: The estimates $\hat{x}[\cdot]$ evolve deterministically, ignoring the effect of the random disturbances (Figure 3).

Clearly, we would prefer estimates $\hat{x}[\cdot]$ which balance the measurement error $\|\delta_v[\cdot]\|_2^2$ with the model error $\|\delta_w[\cdot]\|_2^2$, and vice-versa. In this direction, we propose to obtain estimates by solving the *(regularized) least-squares problem*

$$\underset{x[\cdot]}{\text{minimize}} \quad \sum_{k=0}^{N_e-1} \widetilde{L}_e(x[k], x[k+1]) + V_{f|e}(x[N_e]) \tag{3}$$

with the *stagewise cost function* $\widetilde{L}_e(\cdot)$ and *terminal cost function* $V_{f|e}(\cdot)$ defined as

$$\widetilde{L}_e(x[k], x[k+1]) = \|S_v^{-1}\left(y_n^{\text{data}}[k] - Cx[k]\right)\|_2^2 + \|S_w^{-1}\left(x[k+1] - Ax[k]\right)\|_2^2;$$
$$V_{f|e}(x[N_e]) = \|S_v^{-1}\left(y_n^{\text{data}}[N_e] - Cx[N_e]\right)\|_2^2.$$

A solution to Problem (3) consists of the estimates $\hat{\mathbf{x}} = (\hat{x}[0], \ldots, \hat{x}[N_e])$ associated with $\mathbf{y}_n^{\text{data}} = (y[n-N_e], \ldots, y[n])$, such that $\hat{x}[k]$ is the estimate of the state $x[n-N_e+k]$ (remember that!). The problem is unconstrained and (strictly) convex, meaning that it can be solved efficiently and the solution is unique. Moreover, this least-square problems is equivalent to a *maximum a posteriori* estimator, with the errors $\|\delta_v\|_2^2$ and $\|\delta_w\|_2^2$ corresponding to the likelihood and regularization terms. In this sense, the solution $\hat{\mathbf{x}}$ is the most likely estimation given the data and the model.

Linear-quadratic estimator form

Let's introduce $w[k] = x[k+1] - Ax[k]$ as a decision variable and define a linear error function $\epsilon(x[k]) = Cx[k] - y_n^{\text{data}}$. After some tedious algebra, Problem (3) can be equivalently formulated as the *optimal estimation problem (OEP)*

$$\underset{x[\cdot],\ w[\cdot]}{\text{minimize}} \quad \sum_{k=0}^{N_e-1} \begin{bmatrix} \epsilon(x[k]) \\ w[k] \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} Q_e & \\ & R_e \end{bmatrix} \begin{bmatrix} \epsilon(x[k]) \\ w[k] \end{bmatrix} + \epsilon(x[k])^{\mathsf{T}} Q_e \epsilon(x[k]) \tag{4a}$$

$$\text{subject to} \quad x[k+1] = Ax[k] + w[k], \qquad k = 0, \ldots, N_e - 1. \tag{4b}$$

with $Q_e = (S_v^2)^{-1}$ and $R_e = (S_w^2)^{-1}$. Problem (4) has a familiar structure: it resembles an *optimal control problem*! In fact, all the particularities of optimal control problems are applicable for optimal estimation problems, including their solution methods. Here, the *tuning matrices* (Q_e, R_e) control the tradeoff between minimizing $\epsilon(x[k]) = Cx[k] - y^{\text{data}}[k]$ (the measurement error) and minimizing $w[k]$ (the model error). If $Q_e \succ R_e$ (or, equivalently, $S_v^2 \prec S_w^2$), then our estimator prioritizes matching the measurement data over trusting the model predictions, and vice-versa. This is intuitive, as large covariance matrices imply more uncertainty about the data/predictions. Problem (4) can also be extended with linear constraints on $x[\cdot]$ and $w[\cdot]$, if necessary. Of course, this problem still differs from a standard LQR problem in some aspects: it has no boundary conditions and it minimizes a linear function of the state ($\epsilon(\cdot)$) rather than the state itself. Aside from these technicalities, it is interpreted similarly to an optimal control problem.

2 Receding-horizon estimation policies

An estimator based on Problem (4) is very simple: we program our hardware (microcontroller, computer, etc.) to numerically solve this OEP, then collect the relevant optimal estimates. Formally, the estimation policy would be

$$\hat{x}[n] = \underbrace{e_{N_e} \circ \texttt{OEP}(\mathbf{y}_n^{\text{data}})}_{K_{\text{RHE}}(\mathbf{y}_n^{\text{data}})}, \qquad \text{where} \quad \texttt{OEP}(\mathbf{y}_n^{\text{data}}) = \arg\min_{\mathbf{x}} \left\{ V_{e|N_e}(\mathbf{x}, \mathbf{y}_n^{\text{data}}) \mid \mathbf{x} \in \mathcal{X}_{N_e}^e(\mathbf{y}_n^{\text{data}}) \right\} \tag{5}$$

where e_{N_e} is a N_eth-element selector and $\texttt{OEP}(\mathbf{y}_n^{\text{data}})$ returns a solution to the OEP (4), parametrized by $\mathbf{y}_n^{\text{data}}$. Here, $V_{e|N_e}(\cdot)$ denotes the objective function and $\mathcal{X}_{N_e}^e(\cdot)$ denotes the feasible set, which in this case is the whole $\mathbb{R}^{(N_e+1)N_x}$.

The *receding-horizon estimation policy* $K_{\text{RHE}}(\cdot)$ produces the most likely state estimates given a finite collection of measurements, in real-time. Using this policy, the estimator (i) solves the OEP to obtain the $\hat{\mathbf{x}} = (\hat{x}[0], \dots, \hat{x}[N_e])$, then (ii) emits the last $\hat{x}[N_e]$ as the estimate of the current (unknown) state $x[n]$. Subsequently, the estimator *shifts* the estimation horizon such that the measurement dataset becomes $y_{n+1}^{\text{data}} = (y[n-N_e+1], \dots, y[n+1])$, then repeats the algorithm. A state estimator based on this policy is known as a *moving-horizon estimator* (MHE, Figure 4).

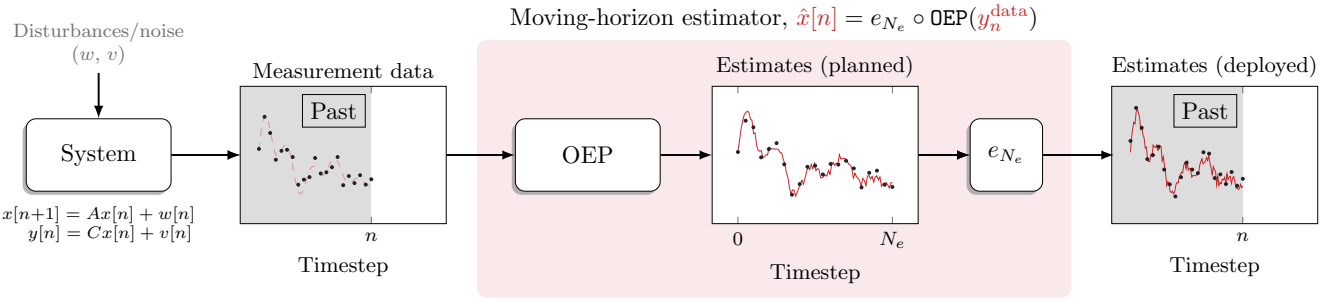

Figure 4. Illustration of the moving-horizon estimator. In this strategy, the OEP is solved for each nth timestep given the current measurement batch $\mathbf{y}_n^{\text{data}}$. The estimator computes the whole sequence $\hat{\mathbf{x}} = (\hat{x}[0], \dots, \hat{x}[N_e])$, but only the last is actually considered. The procedure is then repeated at the next estimation cycle, when a new measurement arrives. The estimation cycles have a frequency of $1/\Delta t$.

The MHE is structurally similar to an MPC, just as OEPs are structurally similar to OCPs. Therefore, many of the previous discussions on receding-horizon control also extends to receding-horizon estimation. Here, however, there is an important particularity: the MPC operation only information about the *current* state, while the MHE operation requires information about the *current and past* measurements. In this sense, the MPC is said to be a *memoryless* device, whereas the MHE is a *memory-based* device. A direct consequence is that the physical implementation of MHEs require devices with enough memory capabilities to store (and update) the measurement dataset. Thankfully, for linear systems with AWGN, this memory overhead can be alleviated using observers/estimators in which the estimates are updated recursively. One such observer is the celebrated Kalman filter (KF) and its extensions [1].

The batch of measurement data can be defined recursively as $\mathbf{y}_n^{data} = \left[\mathbf{y}_{n-1}^{data}[2{\rightsquigarrow}N_e], y[n]\right]$.

Linearization-based estimation of nonlinear systems

Until now, we discussed estimator design with the underlying assumption that the system to be estimated follows a linear dynamic model. However, as we already know, most relevant applications concern nonlinear systems. Therefore, we need a strategy to extend the above results to such a class of systems.

The task of estimating the state of a nonlinear system can be approached through a linearization method. Considering a steady-state x_{ref} such that $0 = f(x_{\text{ref}}, 0)$ and $y_{\text{ref}} = g(x_{\text{ref}}, 0)$, we have that

$$\begin{cases} \frac{\mathrm{d}}{\mathrm{d}t}x(t) = f(x(t), w^c(t)) \\ \quad y(t) = g(x(t), v^c(t)) \end{cases} \xrightarrow{\text{Linearization and discretization}} \begin{cases} x^\delta[n+1] = A_{\Delta t}x^\delta[n] + w[n] \\ \quad y^\delta[n] = Cx^\delta[n] + v[n] \end{cases} \tag{6}$$

with output deviation $y^\delta[n] = y[n] - y_{\text{ref}}$ and (w^c, v^c) being the continuous-time white noise processes such that $w[k] \sim \mathcal{N}(0, S_w^2)$ and $v[k] \sim \mathcal{N}(0, S_v^2)$. Using this linearized model, we design the linear-quadratic estimation problem

$$\underset{x[\cdot],\ w[\cdot]}{\text{minimize}} \quad \sum_{k=0}^{N_e-1} \begin{bmatrix} \epsilon(x^\delta[k]) \\ w[k] \end{bmatrix}^\mathsf{T} \begin{bmatrix} Q_e & \\ & R_e \end{bmatrix} \begin{bmatrix} \epsilon(x^\delta[k]) \\ w[k] \end{bmatrix} + \epsilon(x^\delta[k])^\mathsf{T} Q_e \epsilon(x^\delta[k]) \tag{7a}$$

$$\text{subject to} \quad x^\delta[k+1] = A_{\Delta t}x^\delta[k] + w[k], \qquad k = 0, \dots, N_e - 1. \tag{7b}$$

with $y_n^{\delta,\text{data}}[k] = y_n^{\text{data}}[k] - y_{\text{ref}}$. The estimation policy is $\hat{x}^\delta[n] = K_{\text{RHE}}(y_n^{\delta,\text{data}})$ or, in terms of the original signals,

$$\hat{x}[n] = x_{\text{ref}} + K_{\text{RHE}}(y_n^{\text{data}} - y_{\text{ref}}). \tag{8}$$

Thus, when estimating the state of nonlinear systems, we simply *shift* the measurement data as $y_n^{\text{data}} - y_{\text{ref}}$, solve the LQ Problem (7), which is structurally identical to the original problem, then *shift* the solution by x_{ref}. Similarly to linearization-based control, this policy produces accurate estimates *as long as the state remains in the vicinity of the linearization point*. If the system escapes the region of attraction of x_{ref}, more advanced methods are required.

References

[1] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl. *Model Predictive Control: Theory, Computation and Design*. 2nd ed. Nob Hill Publ., LLC., 2020. ISBN: 978-0-9759377-5-4.