

Figure 18.3: Lowpass filters can be used to smooth noisy estimates

dynamic systems which unfortunately lags the estimate (makes it more sluggish). If the estimates are applied in feedback control system, this lag may reduce the stability of the control system. How can you find the allowable amount of filtering, for example the maximum time-constant of the lowpass filters?⁴

18.5 Estimating parameters and disturbances with Kalman Filter

In many applications the Kalman Filter is used to estimate parameters and/or disturbances in addition to the “ordinary” state variables. One example is dynamic positioning systems for ship position control where the Kalman Filter is used to estimate environmental forces acting on the ship (these estimates are used in the controller as a feedforward control signal).

These parameters and/or disturbances *must be represented as state variables*. They represent additional state variables. The original state vector is *augmented* with these new state variables which we may denote the *augmentative states*. The Kalman Filter is used to estimate the augmented state vector which consists of both the original state variables and the augmentative state variables. But how can you model these augmentative state variables? The augmentative model must be in the form of a difference equation because that is the model form when designing a Kalman Filter. To set up an augmentative model you must make an assumption about the behaviour of the augmentative state. Let us look at some augmentative models.

⁴By simulating the system.

- **Augmentative state is (almost) constant:** The most common augmentative model is based on the assumption that the augmentative state variable x_a is slowly varying, almost constant. The corresponding differential equation is

$$\dot{x}_a(t) = 0 \quad (18.61)$$

Discretizing this differential equation with the Euler Forward method gives

$$x_a(k+1) = x_a(k) \quad (18.62)$$

which is a difference equation ready for Kalman Filter algorithm. It is however common to assume that the state is driven by some noise, hence the augmentative model become:

$$x_a(k+1) = x_a(k) + w_a(k) \quad (18.63)$$

where w_a is white process noise with assumed auto-covariance on the form $R_{w_a}(L) = Q_a\delta(L)$. As pointed out in Section 18.3, the variance Q_a can be used as a tuning parameter of the Kalman Filter.

- **Augmentative state has (almost) constant rate:** The corresponding differential equation is

$$\ddot{x}_a = 0 \quad (18.64)$$

or, in state space form, with $x_{a1} \equiv x_a$,

$$\dot{x}_{a1} = x_{a2} \quad (18.65)$$

$$\dot{x}_{a2} = 0 \quad (18.66)$$

where x_{a2} is another augmentative state variable. Applying Euler Forward discretization with sampling interval h [sec] to (18.65) – (18.66) and including white process noise to the resulting difference equations gives

$$x_{a1}(k+1) = x_{a1}(k) + hx_{a2}(k) + w_{a1}(k) \quad (18.67)$$

$$x_{a2}(k+1) = x_{a2}(k) + w_{a2}(k) \quad (18.68)$$

Once you have defined the augmented model, you can design and implement the Kalman Filter in the usual way. The Kalman Filter then estimates both the original states and the augmentative states.

The following example shows how the state augmentation can be done in a practical (simulated) application. The example also shows how to use functions in LabVIEW and in MATLAB to calculate the steady state Kalman Filter gain.

Example 18.2 Kalman Filter for estimating level and flow

This example is essentially the same as Example 17.3 where an *observer* was used.

Figure 17.7 shows the system as drawn on the front panel of a LabVIEW based simulator. In stead of the observer (see down left on the front panel) a Kalman Filter is now used. We will design a steady state Kalman Filter to estimate the outflow F_{out} . The level h is measured.

Mass balance of the liquid in the tank is (mass is ρAh)

$$\rho A_{tank} \dot{h}(t) = \rho K_p u - \rho F_{out}(t) \quad (18.69)$$

$$= \rho K_p u - \rho F_{out}(t) \quad (18.70)$$

After cancelling the density ρ the model is

$$\dot{h}(t) = \frac{1}{A_{tank}} [K_p u - F_{out}(t)] \quad (18.71)$$

We assume that the unknown outflow is slowly changing, almost constant. We define the following augmentative model:

$$\dot{F}_{out}(t) = 0 \quad (18.72)$$

The model of the system is given by (18.71) – (18.72). Although it is not necessary, it is convenient to rename the state variables using standard names. So we define

$$x_1 = h \quad (18.73)$$

$$x_2 = F_{out} \quad (18.74)$$

The model (18.71) – (18.72) is now

$$\dot{x}_1(t) = \frac{1}{A_{tank}} [K_p u(t) - x_2(t)] \equiv f_{cont_1} \quad (18.75)$$

$$\dot{x}_2(t) = 0 \equiv f_{cont_2} \quad (18.76)$$

Applying Euler Forward discretization with time step T_s and including white disturbance noise in the resulting difference equations yields

$$x_1(k+1) = x_1(k) + \underbrace{\frac{T_s}{A_{tank}} [K_p u(k) - x_2(k)]}_{f_1(\cdot)} + w_1(k) \quad (18.77)$$

$$x_2(k+1) = \underbrace{x_2(k)}_{f_2(\cdot)} + w_2(k) \quad (18.78)$$

or

$$x(k+1) = f[x(k), u(k)] + w(k) \quad (18.79)$$

w_1 and w_2 are independent (uncorrelated) white process noises with assumed variances $R_{w_1}(L) = Q_1\delta(L)$ and $R_{w_2}(L) = Q_2\delta(L)$ respectively. Here, Q_1 and Q_2 are variances. The multivariable noise model is then

$$w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \quad (18.80)$$

with auto-covariance

$$R_w(L) = Q\delta(L) \quad (18.81)$$

where

$$Q = \begin{bmatrix} Q_{11} & 0 \\ 0 & Q_{22} \end{bmatrix} \quad (18.82)$$

Assuming that the level x_1 is measured, we add the following measurement equation to the state space model:

$$y(k) = g[x_p(k), u(k)] + v(k) = x_1(k) + v(k) \quad (18.83)$$

where v is white measurement noise with assumed variance

$$R_v(L) = R\delta(L) \quad (18.84)$$

where R is the measurement variance.

The following numerical values are used:

$$\text{Sampling time: } T_s = 0.1 \text{ s} \quad (18.85)$$

$$A_{tank} = 0.1 \text{ m}^2 \quad (18.86)$$

$$K_p = 0.002 \text{ (m}^3/\text{s)/V} \quad (18.87)$$

$$Q = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.0001 \end{bmatrix} \quad (18.88)$$

$$R = 0.0001 \text{ m}^2 \text{ (Gaussian white noise)} \quad (18.89)$$

We will set the initial estimates as follows:

$$x_{1_p}(0) = x_1(0) = y(0) \text{ (from the sensor)} \quad (18.90)$$

$$x_{2_p}(0) = 0 \text{ (assuming no information about initial value)} \quad (18.91)$$

The Kalman Filter algorithm is as follows: The predicted level measurement is calculated according to (18.35):

$$y_p(k) = g[x_p(k), u(k)] = x_{1_p}(k) \quad (18.92)$$

with initial value as given by (18.90). The innovation variable is calculated according to (18.36), where y is the level measurement:

$$e(k) = y(k) - y_p(k) \quad (18.93)$$

The corrected state estimate is calculated according to (18.37):

$$x_c(k) = x_p(k) + Ke(k) \quad (18.94)$$

or, in detail:

$$\begin{bmatrix} x_{1c}(k) \\ x_{2c}(k) \end{bmatrix} = \begin{bmatrix} x_{1p}(k) \\ x_{2p}(k) \end{bmatrix} + \underbrace{\begin{bmatrix} K_{11} \\ K_{21} \end{bmatrix}}_K e(k) \quad (18.95)$$

This is the applied estimate!

The predicted state estimate for the next time step, $x_p(k+1)$, is calculated according to (18.38):

$$x_p(k+1) = f[x_c(k), u(k)] \quad (18.96)$$

or, in detail:

$$\begin{bmatrix} x_{1p}(k+1) \\ x_{2p}(k+1) \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} x_{1c}(k) + \frac{T_s}{A_{\text{tank}}} [K_p u(k) - x_{2c}(k)] \\ x_{2c}(k) \end{bmatrix} \quad (18.97)$$

To calculate the steady state Kalman Filter gain K_s the following information is needed, cf. Figure 18.2:

$$A = I + T_s \left. \frac{\partial f_{\text{cont}}}{\partial x} \right|_{x_p(k), u(k)} \quad (18.98)$$

$$= I + T_s \left[\begin{array}{cc} \left. \frac{\partial f_{\text{cont}1}}{\partial x_1} = 0 \right|_{x_p(k), u(k)} & \left. \frac{\partial f_{\text{cont}1}}{\partial x_2} = -\frac{1}{A_{\text{tank}}} \right|_{x_p(k), u(k)} \\ \left. \frac{\partial f_{\text{cont}2}}{\partial x_1} = 0 \right|_{x_p(k), u(k)} & \left. \frac{\partial f_{\text{cont}2}}{\partial x_2} = 0 \right|_{x_p(k), u(k)} \end{array} \right] \quad (18.99)$$

$$= \begin{bmatrix} 1 & -\frac{T_s}{A_{\text{tank}}} \\ 0 & 1 \end{bmatrix} \quad (18.100)$$

$$G = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I_2 \text{ (identity matrix)} \quad (18.101)$$

$$C = \left. \frac{\partial g(\cdot)}{\partial x} \right|_{x_p(k), u(k)} \quad (18.102)$$

$$= \begin{bmatrix} 1 & 0 \end{bmatrix} \quad (18.103)$$

$$Q = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.0001 \end{bmatrix} \text{ (initially, may be adjusted)} \quad (18.104)$$

$$R = 0.000001 \text{ m}^2 \quad (18.105)$$

Figure 18.4 shows the “real” (simulated) and estimated level and outflow.

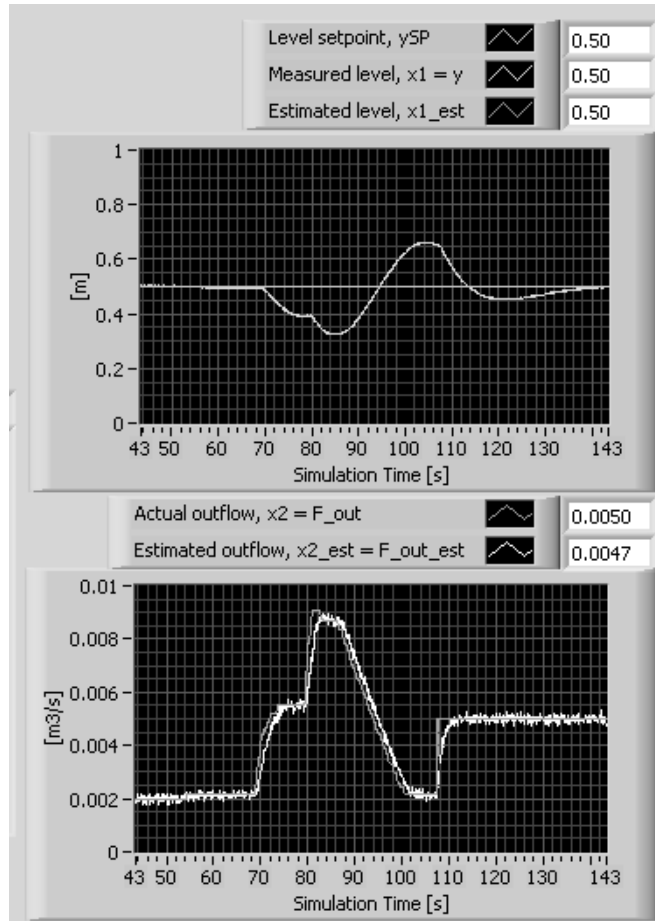


Figure 18.4: Example 18.2: Responses in the level and the real and estimated outflow

We see from the lower chart in the figure that the Kalman Filter estimates the outflow well, and with zero error in steady state.

Figure 18.5 shows how the steady state Kalman Filter gain K_s is calculated using the **Kalman Gain** function. The figure also shows how to check for observability with the **Observability Matrix** function. The linear continuous-time model is discretized using the **Convert Continuous to Discrete** function. Figure 18.6 shows the values of the Kalman gains.

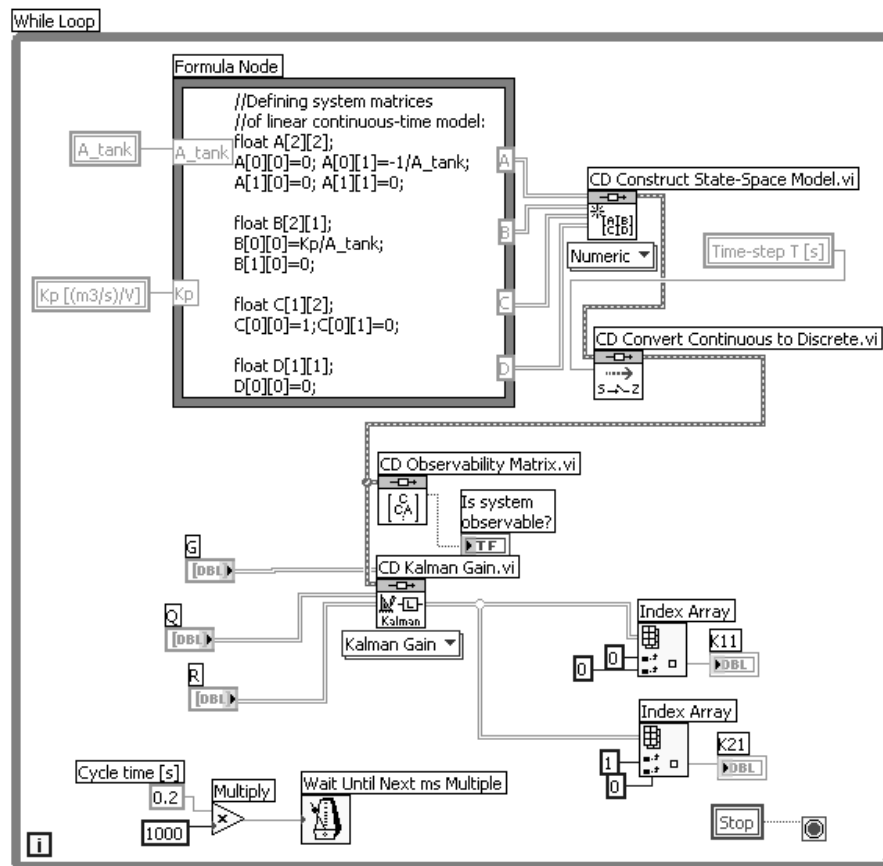


Figure 18.5: Example 18.2: Calculation of the steady state Kalman Filter gain with the Kalman Gain function, and checking for observability with the Observability Matrix function.

Figure 18.7 shows the implementation of the Kalman Filter equations in a Formula Node. (The Formula Node is just one part of the block diagram. The total block diagram consists of one While loop where the Kalman gains are calculated, and one Simulation loop containing the Formula Node, PID controller, and the tank simulator.)

The steady state Kalman Gain K_s is calculated in the LabVIEW program. Alternatively, K_s can be calculated in MATLAB, as shown below. The `dlqe`⁵ function belongs to the Control System Toolbox.

```
A_tank=0.1;
Kp=0.002;
```

⁵ dlqe = Discrete-time linear quadratic estimator.

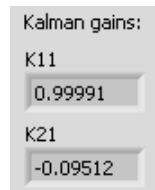


Figure 18.6: Example 18.2: The Kalman gains

```

A_cont=[0,-1/A_tank;0,0]
B_cont=[Kp/A_tank;0];
C_cont=[1,0];
D_cont=[0];
Ts=0.1;
%Generates a state-space model:
sys_cont=ss(A_cont,B_cont,C_cont,D_cont);
sys_disc=c2d(sys_cont,Ts); %Discretizing
A=sys_disc.a;
C=sys_disc.c;
G=[1,0;0,1]
C=[1,0]
Q=[0.01,0;0,1e-4]
R=[1e-6];
[Ks,Pp,Pc,E] = dlqe(A,G,C,Q,R)

```

K_s is the steady state Kalman Filter gain. (P_p and P_c are steady state estimation error auto-covariances, cf. (18.48) and (18.47). E is a vector containing the eigenvalues of the Kalman Filter, cf. (18.54).)

MATLAB answers

```

Ks =
0.99991
-0.09512

```

which are the same values as calculated by the LabVIEW function **Kalman Gain**, cf. Figure 18.7.

[End of Example 18.2]