EE231 Lecture Note

Lecture 10 Kalman Filter

Table of Contents

1 Discretization of state equation

The Kalman filter discussed in this lecture is based on the discrete-time system model. Discrete-time control algorithms can be directly coded in software and executed by embedded controllers. A continuous time state equation can be transformed to its discrete-time equivalent using the state equation's transition matrix Φ(t) as explained below. Consider the state equation (1).

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{1}$$

The state variable at t_2, i.e., $x(t_2)$ can be derived from the state variable at t_1 i.e. $x(t_1)$, the input at t_1, i.e., $u(t_1)$, and the system's state transition matrix $\Phi(t_2 - t_1)$ by (2).

$$x(t_2) = \Phi\left(t_2 - t_1\right) x\left(t_1\right) + \int_{t_1}^{t_2} \Phi\left(t_2 - \tau\right) Bu\left(\tau\right) d\tau \tag{2}$$

The integration in (2) is the convolution integral. (2) can be rewritten as (3) where $(t_2 - t_1)=T$ and k is the time step index integer.

$$x\left[T\left(k+1\right)\right] = \Phi\left(T\right) x\left[Tk\right] + \int_{Tk}^{T(k+1)} \Phi\left(T\left((k+1)-\tau\right)\right) Bu(\tau) d\tau \tag{3}$$

We can assume the input u(t) is held constant between samples. In this case, (3) can be simplified into (4) or (5).

$$x\left[T\left(k+1\right)\right] = \underbrace{\Phi\left(T\right)}_{A_d} x(kT) + \underbrace{\left[\int_0^T \Phi\left(\tau\right) d\tau\right] B}_{B_d} u(kT) \tag{4}$$

$$x\left(k+1\right) = A_d x(k) + B_d u(k) \tag{5}$$

A_d and B_d in (5) are as defined in (4). Equation (5) is in the standard discrete-time state equation form. Since the output equation is algebraic, the discrete-time version of the equation has the same C and D matrix as those in (1).

$$y(k) = Cx(k) + Du(k) \tag{6}$$

Example: Consider the following state equation.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$
$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{7}$$

The transition matrix of the A matrix is

$$\Phi(t) = \begin{bmatrix} 1 & \frac{1}{4}\left(1-e^{-4t}\right) \\ 0 & e^{-4t} \end{bmatrix}. \tag{8}$$

If the sampling interval T is 0.2second, then

$$A_d = \Phi(0.2) = \begin{bmatrix} 1 & 0.1377 \\ 0 & 0.449 \end{bmatrix} \quad \text{and} \quad B_d = \int_0^{0.2} \Phi(t)\,dt \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.0156 \\ 0.1377 \end{bmatrix} \tag{9}$$

Equation (10) is the equivalent discrete-time state equation of (7).

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 0.1377 \\ 0 & 0.449 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0.0156 \\ 0.1377 \end{bmatrix} u$$
$$y(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \tag{10}$$

The following Matlab script performs this conversion.

```
A=[0 1;0 -4];
B=[0;1];
C=[1 0];
D=0;
syst=ss(A,B,C,D);
Dsyst=c2d(syst,0.2)
[Ad,Bd,Cd,Dd,Ts]=ssdata(Dsyst)
```

2 Means and Variances of Random Variables

The discussion of Kalman filter in this lecture note involves two basic properties of random variables: mean and variance. This section is only a brief review of definitions and interpretations of these two properties. Students are encouraged to review this material from a course on random variables or random processes.

2.1 Mean

The mean value of a discrete value random variable is defined in (11) where p_i is the probability function of x_i. For a continuous value random variable, the mean value is defined in (12) where $f(x)$ is the probability density function of x. The mean value can be thought of as the 'average' value of the random variable and hence, it is also called the 'expected value'. The symbol 'μ' is often used for the mean value.

$$E(x) = \sum_{\text{all } i} x_i p_i = \mu_x \tag{11}$$

$$E(x) = \int x f(x) dx = \mu_x \tag{12}$$

2.2 Variance and Covariance

The variance of a discrete random variable is defined in (13) and, in (14), for continuous value random variable.

$$Var(x) = E\left[\left(x-\mu_x\right)^2\right] = \sum_{\text{all i}} \left(x_i - \mu_x\right)^2 p_i \tag{13}$$

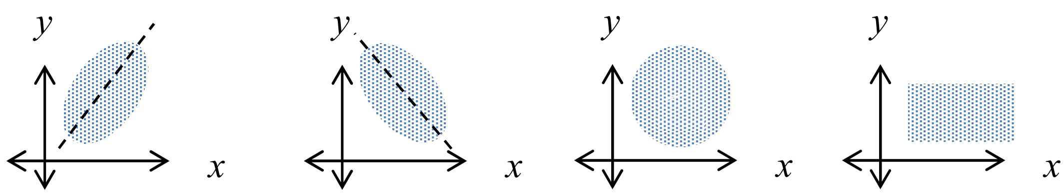
$$Var(x) = E\left[\left(x-\mu_x\right)^2\right] = \int \left(x-\mu_x\right)^2 f(x) dx \tag{14}$$

Variance is a measure of how dispersed a random variable is around its mean value. The square root of the variance is the 'standard deviation' (σ) of the random variable. For any normal distribution random variable, approximately 68% of the data falls within one standard deviation of the mean. Approximately 95% of the data falls within two standard deviations of the mean. Approximately 99.7% of the data falls within three standard deviations of the mean.

The <u>covariance</u> of two discrete value random variables *x* and *y* is defined in (15).

$$\mathrm{cov}(x,y) = \sum_{\text{all i,j}} p_{ij}\left(x_i - \mu_x\right)\left(y_j - \mu_y\right) \tag{15}$$

The shaded areas in the following figures are all possible values of (*x,y*) and the density functions are constants over the areas.



cov(x,y)>0   cov(x,y)<0   cov(x,y)=0   cov(x,y)=0 & independent

Intuitively, covariance is a measure of how two random variables are linearly related statistically. The <u>covariance matrix</u> of the vector V=[x y] is a square matrix as defined in (16). A covariance matrix characterizes not only the variance of each random variable but also the correlation between the two variables.

$$\mathrm{cov}\left(\mathrm{V}\right) = \begin{bmatrix} \mathrm{var}(x) & \mathrm{cov}(x,y) \\ \mathrm{cov}(x,y) & \mathrm{var}(y) \end{bmatrix} \tag{16}$$

2.3 Function of Random Variables

The mean of the sum of two random variables is the sum of the means of the two random variables as described in (17).

$$E(x+y) = E(x) + E(y) \tag{17}$$

The variance of the sum of two random variables is given in (18).

$$\mathrm{var}(x+y) = \mathrm{var}(x) + \mathrm{var}(y) + 2\,\mathrm{cov}(x,y) \tag{18}$$

If *x* and *y* are independent or uncorrelated, i.e. cov(*x,y*)=0,

$$\mathrm{var}(x+y)=\mathrm{var}(x)+\mathrm{var}(y)\ . \tag{19}$$

If the variance of the random variable x is R, i.e.

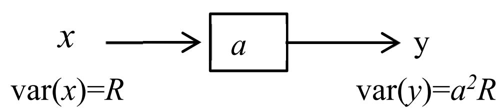
$$\mathrm{var}(x)=R \tag{20}$$

and y is related to x by a scaling factor, a, i.e., the linear function (21)

$$y=ax \tag{21}$$

then the variance of the variable y is

$$\mathrm{var}(y)=a^2R\ . \tag{22}$$



In the vector case, if the covariance matrix of the vector $V=(x, y)$ is R (a square matrix), i.e.

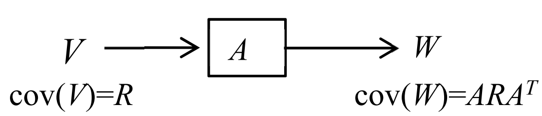
$$\mathrm{cov}(V)=R \tag{23}$$

and W is related to V by a matrix multiplication i.e., (24) below

$$W=AV\ , \tag{24}$$

then the covariance matrix of W is given in (25).

$$\mathrm{cov}(W)=ARA^T\ . \tag{25}$$



3 Kalman Filter

Kalman filter is an algorithm that produces the 'best estimate', in a statistical sense, of the state variable x of a dynamical system (i.e. the plant) based on the plant's dynamics (i.e., its model) and from the plant's input and measuring the plant's output. The filter's structure is similar to that of a state observer. A state observer is able to produce an estimate $\hat{x}$ that converges to the true state under a noise-free and disturbance-free condition. However, a real system is always subjected to disturbance and measurement noise. In the Kalman filter formulation, disturbance and measurement noise are modeled by random variables with known means and variances. Unlike the usual signal filter that attenuates the noise component only based on its frequency content, the Kalman filter takes advantage of the knowledge of the plant dynamics and the statistical property of the disturbance and the noise and comes up with the 'best' estimate of the plant's state variable x <u>in the sense that the estimate's variance is minimized</u>.

3.1 Kalman filter for a scalar algebraic system

The plant is represented by the algebraic equation (26). The plant has three known constant parameters B, G, and C. u is the input and y the output. ω represents the input disturbance and v the measurement noise. ω and v are random variables with zero mean and variances Q and R, respectively. The Kalman filter is represented by equations (27), (28), and (29). The objective of the filter is to estimate the state variable x. Note that the Kalman filter's structure is similar to that of the state observer discussed before; it contains a model of the plant and its inputs are the plant's input (u) and output (y). Figure 1 shows the structures of the Kalman filter and the plant.

Plant:

$$\begin{aligned} x &= Bu + G\omega \\ y &= Cx + v \end{aligned} \tag{26}$$

Kalman filter:

$$\bar{x} = Bu \tag{27}$$

$$\bar{y} = C\bar{x} = CBu \tag{28}$$

$$\hat{x} = \bar{x} + K(y - \bar{y}) \tag{29}$$

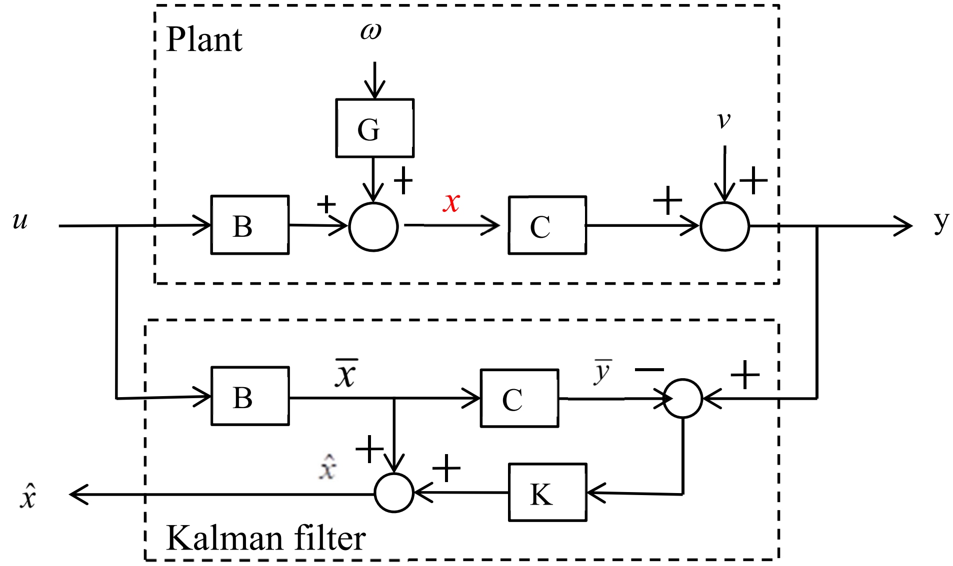


Figure 1

Kalman filter determines the 'best' state estimates in two steps:

Step 1: It produces a 'preliminary' estimate $\bar{x}$ based only on the model of the plant. The measurement of the plant output is NOT used. This step includes equation (27) and (28).

Step 2: It measures the plant output y and use this information to updates the preliminary estimate $\bar{x}$ and produces the final estimate $\hat{x}$. This update is done by adding the term $K(y-\bar{y})$ (called 'innovation') to the preliminary estimate $\bar{x}$ as in (29). The gain K is called 'Kalman gain'. The Kalman gain is designed to minimize the uncertainty in $\hat{x}$ or, equivalently, the variance of the error $\hat{e} = x - \hat{x}$. This derivation is shown below. Equation (30) is an expression of $\hat{e}$.

$$\begin{aligned} \hat{e} = x - \hat{x} &= (Bu + G\omega) - [Bu + K(y - \bar{y})] \\ &= G\omega - K[C(Bu + G\omega) + v - CBu] \\ &= (1 - KC)G\omega - Kv \end{aligned} \tag{30}$$

Assuming the random variables ω and v are not statistically related, from (19) and (22), the variance of $\hat{e}$, as denoted by $\hat{P}$, is given in (31).

$$\mathrm{var}(\hat{e}) = \hat{P} = (1 - KC)^2 G^2 Q + K^2 R \tag{31}$$

To find the K that minimizes $\hat{P}$, we take the derivative of $\hat{P}$ with respect to K and set the result to zero to solve for K^* as in (32) and (33).

$$\begin{aligned}\frac{\partial \hat{P}}{\partial K} &= -2C(1-KC)G^2Q + 2KR \\ &= -2CG^2Q + 2KC^2G^2Q + 2KR\end{aligned} \tag{32}$$

$$\Rightarrow \quad 2\left[-CG^2Q + K^*(C^2G^2Q + R)\right] = 0$$

$$K^* = \frac{CG^2Q}{C^2G^2Q + R} \tag{33}$$

Equations (27)~(29) and (33) are the **<u>Kalman filter</u>** for the algebraic system (26). To better understand the Kalman gain K^*, we consider the following two cases:

(Case I) The measurement noise is zero, i.e., v=0 or R=0. In this case, the Kalman gain is

$$K^* = \frac{CG^2Q}{C^2G^2Q + R} = \frac{1}{C}. \tag{34}$$

In this case, the state is exactly determined (not estimated) as shown by (35).

$$\hat{x} = Bu + \frac{1}{C}\left(y - CBu\right) = Bu + \frac{1}{C}\left[C(Bu + G\omega) - CBu\right] = Bu + G\omega = x \tag{35}$$

(Case II) The measurement noise is very high, i.e., $R=\infty$. In this case, from (34), $K^*=0$. Therefore,

$$\hat{x} = Bu + 0(y - CBu) = Bu \ . \tag{36}$$

In this case the measurement y is not used at all (since it is too noisy) and the best estimate depends only on the input and the model of the system, i.e. $\hat{x} = Bu$.

From these two cases, we see that the innovation term $K(y-\overline{y})$ is weighted by the Kalman gain K according to the measurement noise level (quantified by R) relative to the level of the disturbance ω (quantified by Q). In Figure 2, a potentiometer is used to show this trade-off.

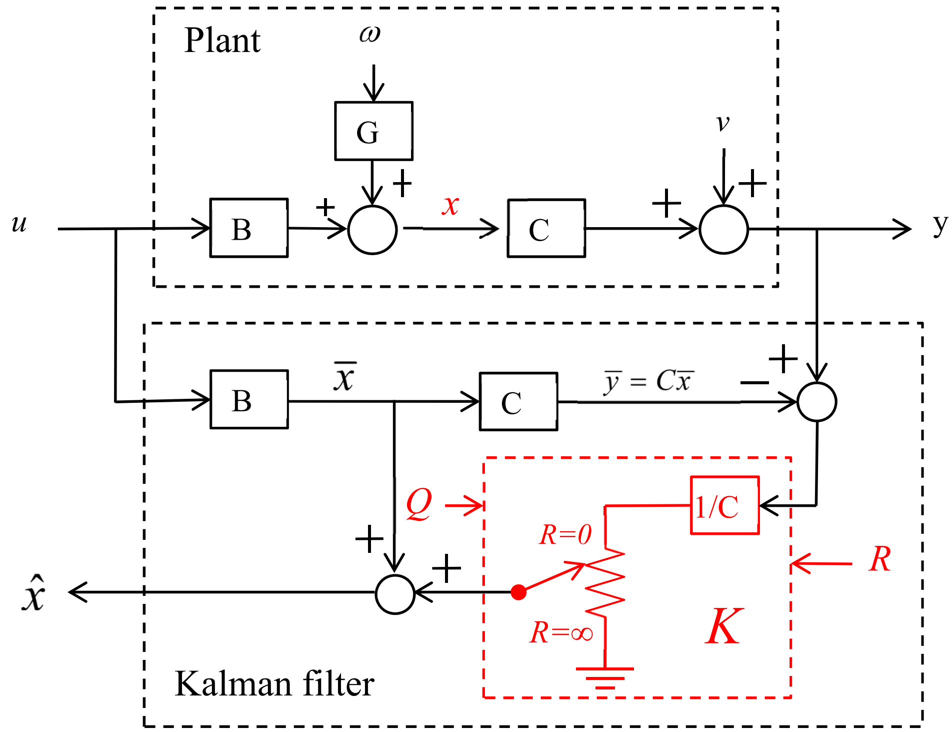


Figure 2

Note that, Kalman filter does not ignore or reduce the effect of the disturbance ω on x. In fact, it is important that it estimates the actual x including the effect of the disturbance ω.

3.2 Kalman filter for discrete-time linear dynamical systems

In the previous section, an algebraic system is used to demonstrates the basic concept of the Kalman filter. Such a system, however is not applicable to most practical systems since most real systems are dynamical systems. In this section, we consider a plant described by a linear discrete-time state equation as in (37). The Kalman filter equations are (38) to (41).

Plant:

$$x_{i+1} = Ax_i + Bu_i + G\omega_i$$
$$y_i = Cx_i + v_i \tag{37}$$

Kalman filter:

$$\bar{x}_{i+1} = A\hat{x}_i + Bu_i \tag{38}$$

$$\hat{x}_i = \bar{x}_i + K_i(y_i - C\bar{x}_i) \tag{39}$$

$$K_i = \frac{C\bar{P}_i}{C^2\bar{P}_i + R}. \tag{40}$$

$$\bar{P}_{i+1} = A^2\left(1 - CK_i\right)\bar{P}_i + G^2Q \tag{41}$$

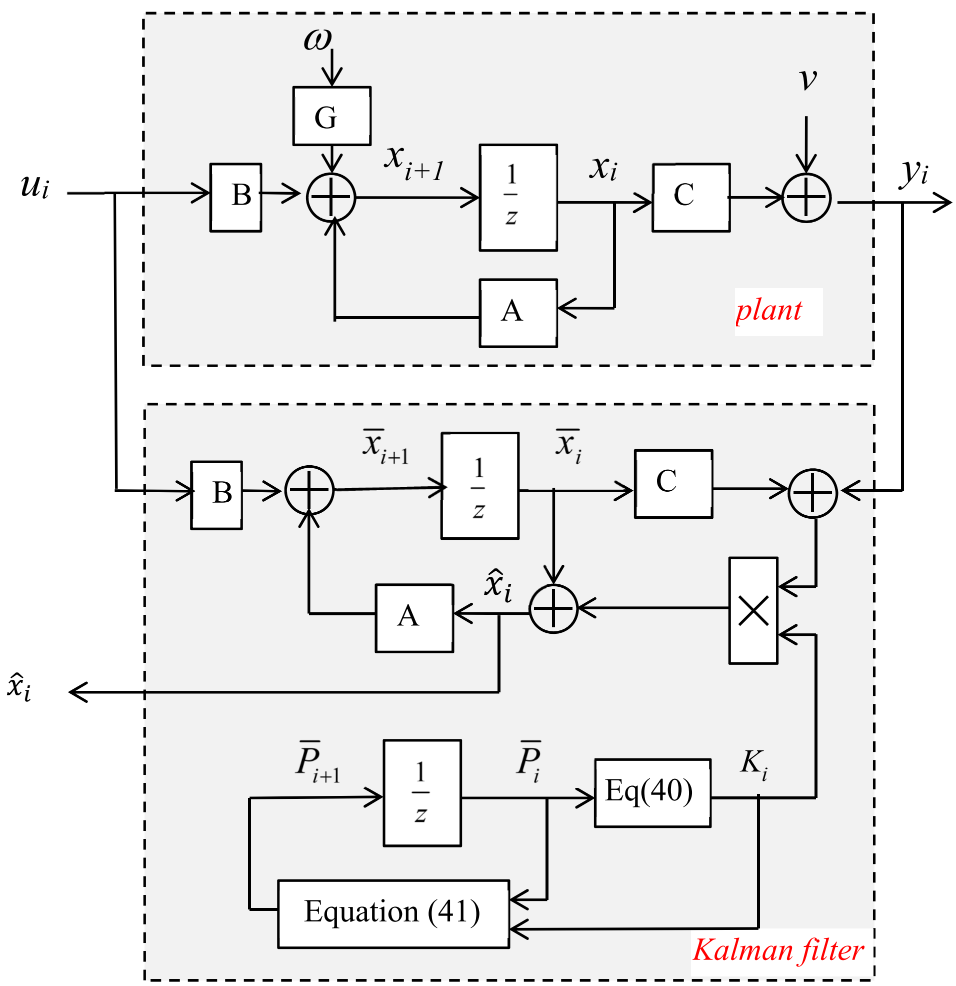
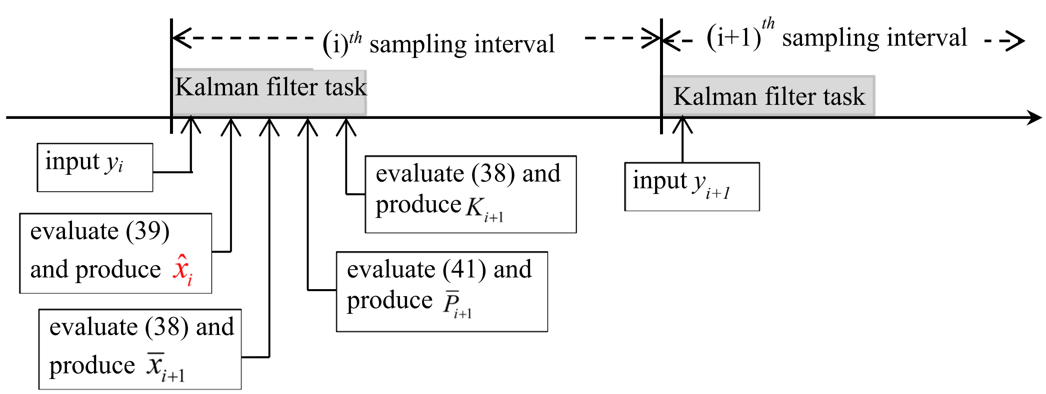


Figure 3

Equation (38) is a model of the plant. This model involves two versions of the state variables: $\bar{x}_i$ and $\hat{x}_i$. The relationship between $\bar{x}_i$ and $\hat{x}_i$ is same as in the algebraic system case. In other words, $\bar{x}_i$ is the state variable that is based on the plant model (38) and has not been 'modified' or 'improved' based on the latest measurement of the plant output y_i. $\hat{x}_i$ is the state variable that is 'modified' or 'improved' using the information from the current measurement y_i. This 'improvement' is done according to (39) where 'K_i' is the Kalman gain. Similar to the Kalman gain in (33) for the algebraic system, this Kalman gain in (39) determines how much the information from the measurement y_i should be used to update the state variable $\bar{x}_i$ to produce the final estimate $\hat{x}_i$. Just like the *K* in (33), the *K* in (40) takes the value of *1/C* if the measurement noise's variance *R* is zero and 0 if *R* is infinity.

In the algebraic system, the variance of $\bar{x}_i$ is simply *Q*, i.e. the variance of the disturbance ω. In the case of a dynamical system, $\bar{x}_i$ is the state variable of the system and the disturbance ω is an input to the system. In this case, the variance of $\bar{x}_i$ is due to a 'filtered' (by the plant dynamics) disturbance ω. For this reason, the variance of $\bar{x}_i$ is determined numerically by iterating the difference equation (41) which represents the effect of the 'filtering' on the variance of $\bar{x}_i$ due to the disturbance ω input. Equation (38)~(41) should be executed by an embedded controller at a fixed interval or a fixed frequency. The following figure shows the evaluation timing of the equations.



3.2.1 Derivation of the Kalman gain (40)

The estimation error $x_i - \hat{x}_i$ can be expressed as (42) where we used (37) and (39).

$$\begin{aligned} x_i - \hat{x}_i &= x_i - \overline{x}_i - K_i(y_i - C\overline{x}_i) \\ &= x_i - \overline{x}_i - K_i(Cx_i + v_i - C\overline{x}_i) \\ &= \left(x_i - \overline{x}_i\right) - K_i C\left(x_i - \overline{x}_i\right) - K_i v_i \\ &= (1 - K_i C)\left(x_i - \overline{x}_i\right)_i - K_i v_i \end{aligned} \tag{42}$$

From (42), we can derive (43) where $\hat{P}_i$ is the variance of the error $x_i - \hat{x}_i$ and $\overline{P}_i$ is the variance of the error $x_i - \overline{x}_i$.

$$\hat{P}_i = (1 - K_i C)^2 \overline{P}_i + K_i^2 R \tag{43}$$

To find the K_i that minimizes $\hat{P}_i$, we differentiate (43) with respect to K_i and then set the result to zero to solve for K_i as in (44).

$$\begin{aligned} \frac{\partial \hat{P}_i}{\partial K_i} &= -2(1 - K_i C)C\overline{P}_i + 2K_i R \\ &= -2C\overline{P}_i + 2K_i C^2 \overline{P}_i + 2K_i R \\ &= -2C\overline{P}_i + 2K_i\left(C^2 \overline{P}_i + R\right) = 0 \end{aligned} \tag{44}$$

The solution of (44) is (45) which is the same equation as (40).

$$K_i = \frac{C\overline{P}_i}{C^2 \overline{P}_i + R}. \tag{45}$$

3.2.2 Derivation of $\overline{P}_i$ (41)

Subtracting equation (38) from (37), we obtain (46) :

$$\begin{array}{lll} (37) & \Rightarrow & x_{i+1} = Ax_i + \cancel{Bu_i} + G\omega_i \\ (38) & \Rightarrow & \overline{x}_{i+1} = A\hat{x}_i + \cancel{Bu_i} \\ \hline \multicolumn{3}{c}{x_{i+1} - \overline{x}_{i+1} = A(x_i - \hat{x}_i) + G\omega_i} \end{array} \tag{46}$$

From (46), we see that, in this case, the variance $\overline{P}_{i+1}$ is related to the variance $\hat{P}_i$ by (47).

$$\overline{P}_{i+1} = A^2 \ \overline{P}_i + G^2 Q \tag{47}$$

Equation (48) is obtained by substituting (45) into (43).

$$\begin{aligned}\hat{P}_i &= (1-K_i C)^2 \overline{P}_i + K_i^2 R_i = \overline{P}_i - 2CK_i\overline{P}_i + (C^2\overline{P}_i + R)K_i^2 \\ &= \overline{P}_i - 2CK_i\overline{P}_i + \frac{(C\overline{P}_i)^2}{C^2\overline{P}_i + R} \\ &= \overline{P}_i - 2CK_i\overline{P}_i + CK_i\overline{P}_i \\ &= \left(1 - CK_i\right)\overline{P}_i\end{aligned} \tag{48}$$

Now, substituting (48) into (47), we obtain (49) which is (41).

$$\overline{P}_{i+1} = A^2\left(1-CK_i\right)\overline{P}_i + G^2 Q \tag{49}$$

3.2.3 Multi-dimensional case

For notational simplicity, all the above analyses and equations are for a scalar system. Fortunately, the Kalman filter equation (38) and (39) directly apply to multi-dimensional systems without modification. The vector forms of these equations (40) and (41) are shown in (50) and (51) below.

$$K_i = \frac{C\overline{P}_i}{C^2\overline{P}_i + R} \quad \longrightarrow \quad K_i = \overline{P}_i C^T\left(C\overline{P}_i C^T + R\right)^{-1} \tag{50}$$

$$\overline{P}_{i+1} = A^2\left(1-CK_i\right)\overline{P}_i + G^2 Q \quad \longrightarrow \quad \overline{P}_{i+1} = A\left(1-CK_i\right)\overline{P}_i A^T + GQG^T \tag{51}$$

3.3 Steady state solution of the Kalman gain K_i.

It is obvious from Figure 3 that the iteration of $\overline{P}_i$ (i.e. equations (41) or (51)) is isolated from the rest of the system and does not depend on any real-time signal in the system. For this reason, the sequence $\overline{P}_i$ can be computed offline. Furthermore, the iteration often converges to a steady state value, i.e., $\overline{P}_i \to \overline{P}_\infty$ after some iterations. This steady state value can be calculated off line based on (52) and (53) for the case that the plant output is measured at every sampling interval,

$$\overline{P}_\infty = A\left(1 - CK_\infty\right)\overline{P}_\infty A^T + GQG^T \tag{52}$$

$$K_\infty = \overline{P}_\infty C^T\left(C\overline{P}_\infty C^T + R\right)^{-1} \tag{53}$$

Substituting (53) into (52), we obtain (54) which is in the form of a discrete time Riccati equation.

$$\overline{P}_\infty = A\overline{P}_\infty A^T - A\overline{P}_\infty C^T\left(R + C\overline{P}_\infty C^T\right)^{-1} C\overline{P}_\infty A^T + GQG^T \tag{54}$$

Equations (53) and (54) can be solved by using the Matlab command **dlqe** (discrete time linear quadratic estimator).

```
[K,P,Z,E] = dlqe(A,G,C,Q,R)
```

The returned value `K` is the steady state Kalman gain, `P` is $\overline{P}_\infty$, the steady state solution of the Riccati equation (54), `Z` is the steady state $\hat{P}_\infty$ from (50), and `E` is the eigenvalues of the observer dynamics, i.e., the Kalman filter dynamics.

3.4 Multiple sampling rate systems

In some applications, the plant output measurements y_i may not be available at every iteration of the Kalman filter and control algorithm. In such a case, instead of equation (38), equation (55) can be iterated at a higher rate between two measurements. The matrix A_o in (55) should be determined for the higher iteration rate. The A matrix in (51) should be determined for the rate of the measurement.

$$\overline{x}_{k+1} = A_o\overline{x}_k + Bu_k \tag{55}$$

The state estimate generated by (55) between measurements is entirely based on the plant model and is running open-loop. This state estimate bridges the time between measurements and provides a smoother state estimate and control.

3.5 Examples

3.5.1 A second-order system example

Consider the continuous time system (56) where u is the input and ω and v are the disturbance input and measurement noise, respectively. In this example, the plant output is measured at every iteration.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -5 & -1 \\ -2 & -10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 10 \\ 20 \end{bmatrix} u + \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix}$$
$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \tag{56}$$

In this example, ω is represented by a 2D random variables [ω_1 ω_2] with covariance matrix Q and the measurement noise is represented by [v_1 v_2] with covariance matrix R. The values of Q and R used in the simulation are shown in (57).

$$Q = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \qquad R = \begin{bmatrix} 0.5 & 0 \\ 0 & 1 \end{bmatrix} \tag{57}$$

The input to the plant (u) is a 0.5Hz 10v magnitude square wave signal. Figure 4 shows the response of the plant (the state vector x) in the ideal case, i.e., where there is no disturbance and no measurement noise, i.e., when ω=v=0.

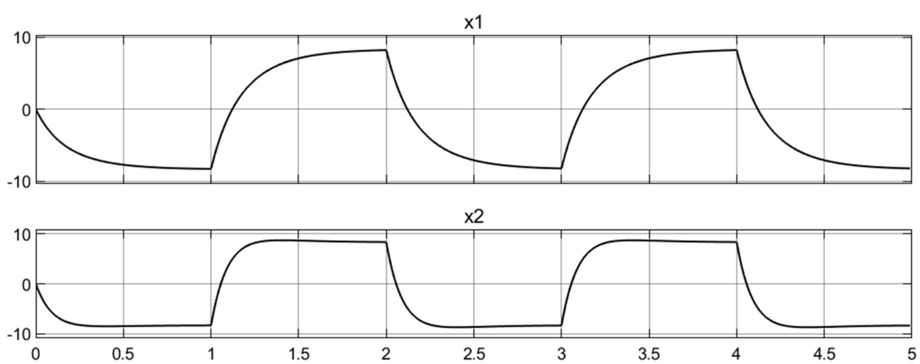


Figure 4 Ideal case responses

The first two graphs in Figure 5 are the <u>actual</u> state variable (x_1 and x_2, in black) and the measured state variables (in red). The disturbance ω and measurement noise v with the covariance matrices

(57) are shown in the last two graphs in Figure 5. As shown, the disturbance ω change its value every 0.1 second and the measurement noise is sampled at 0.01 second interval.

The objective is to estimate the actual state x (as shown in black in the first two graphs in Figure 5 but not the state in the ideal case as in Figure 4. While the effect of the disturbance is not desirable, the estimator should accurately estimate the state with the effect of the disturbance such that the effect of the disturbance can be rejected by the feedback control, if so desired.

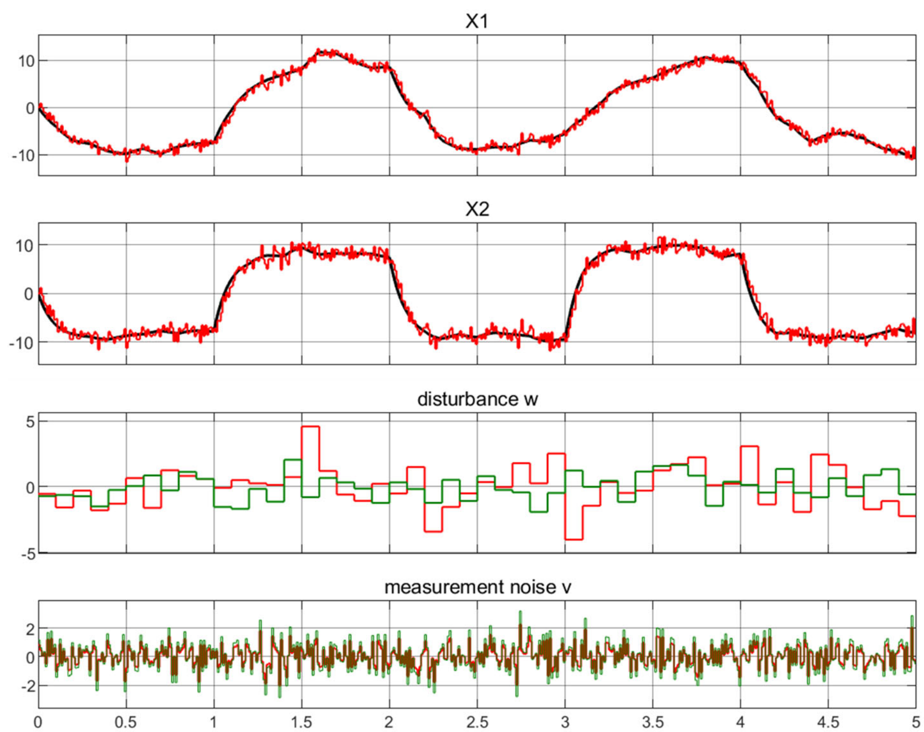


Figure 5 Plant responses and measured responses with Q=diag(2,1) R=diag(0.5 1)

One might attempt to filter out the noise by a simple low-pass filter, like the one in (58).

$$H(s) = \frac{15}{s+15} \tag{58}$$

Figure 6 shows the measured state after the filter (58). As shown, the noise components are effectively removed by the filter. However, the filter introduced a significant amount of time delay. This delay can be detrimental to the system stability if the filtered signal is used for feedback control.

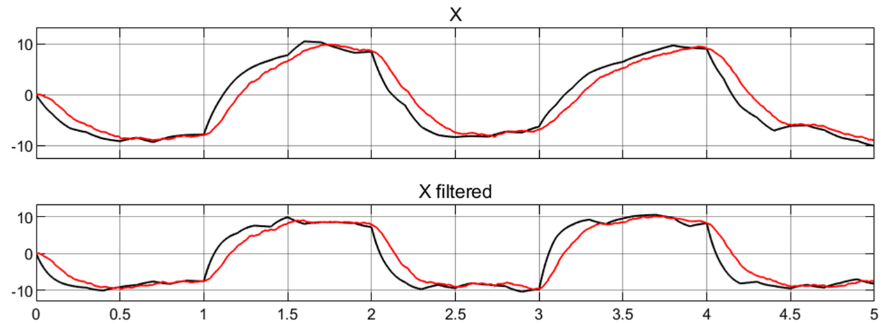


Figure 6 Using simple 1st order filter, not Kalman filter.

The following Matlab script converts the plant model (56) to a discrete time equivalent at 0.01 second sampling interval. The script also determines the steady state Kalman gain. Equation (59) is the discrete time equivalent system of (56).

```
A=[-5 -1;-2 -10];
B=[10;20];
C=eye(2);
D=0;
syst=ss(A,B,C,D);        % input u
Dsyst=c2d(syst,0.01)
[dA,dB,dC,dD,Ts]=ssdata(Dsyst)

syst1=ss(A,eye(2)*10,C,D);   % disturbance w input
Dsyst1=c2d(syst1,0.01)
[dA1,dB1,dC1,dD1,Ts1]=ssdata(Dsyst1)

R=[0.5 0; 0 1];
Q=[2 0;0 1]
G=dB1;
[K,P,Z,E]=dlqe(dA,G,dC,Q,R) % find steady state Kalman gain K
```

$$\begin{bmatrix} x1_{i+1} \\ x2_{i+1} \end{bmatrix} = \begin{bmatrix} 0.9513 & -0.009275 \\ -0.01856 & 0.9049 \end{bmatrix} \begin{bmatrix} x1_i \\ x2_i \end{bmatrix} + \begin{bmatrix} 0.09659 \\ 0.1894 \end{bmatrix} u + \underbrace{\begin{bmatrix} 0.0975 & -4.757\times10^{-3} \\ -9.514\times10^{-3} & 0.0952 \end{bmatrix}}_{=dB1=G} \begin{bmatrix} \omega1_i \\ \omega2_i \end{bmatrix} \quad (59)$$

$$\begin{bmatrix} y1_i \\ y2_i \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x1_i \\ x2_i \end{bmatrix} + \begin{bmatrix} v1_i \\ v2_i \end{bmatrix}$$

Figure 7 shows the block diagram of the Kalman filter. The user defined function block on the left side is equation (53) and the one on the right is (54).

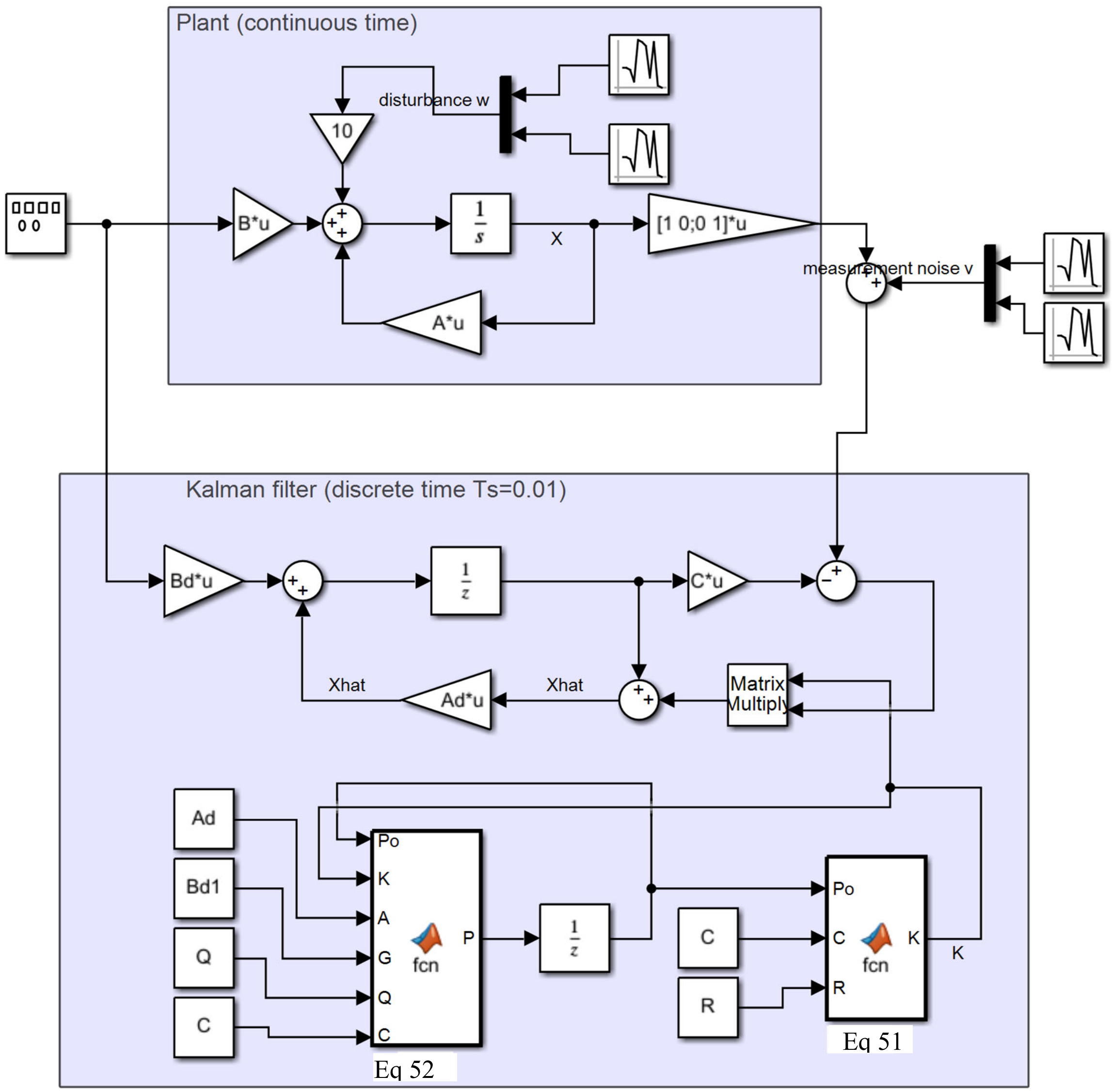


Figure 7

Figure 8 shows the result of the simulation. As shown, the estimated state (xhat, in red) closely tracks the actual state (x, in black). Unlike those in Figure 6, the estimated state does not lag behind the actual state by much and is nearly noise free.

The Kalman filter block in Figure 7 is available as a standard block in the Simulink library. To reach the block, follow this link: Control System toolbox → State Estimation → Kalman filter or simply search for ‘Kalman filter’.

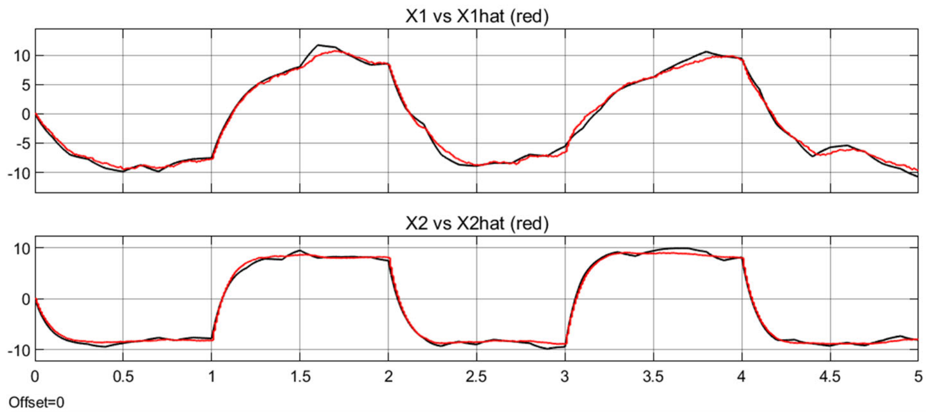


Figure 8 Actual state and estimated state with Q=diag(2,1) R=diag(0.5 1)

Figure 9 shows the convergence of the Kalman gain. As shown, it converges to

$$K = \begin{bmatrix} 0.1444 & -0.0051 \\ -0.01 & 0.0412 \end{bmatrix}. \tag{60}$$

This result agrees with the result from the Matlab command (55).

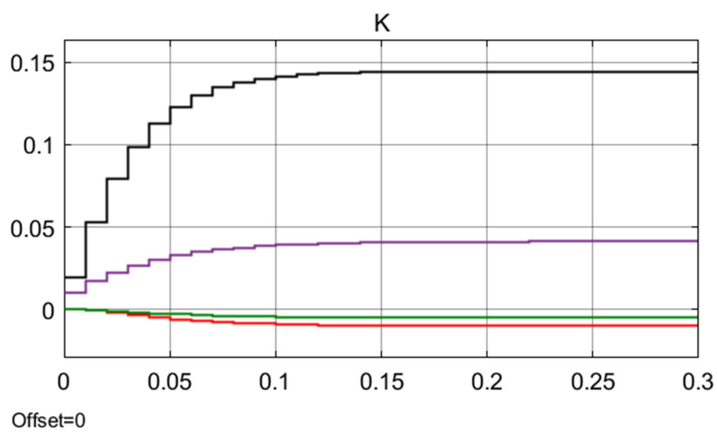


Figure 9 The Kalman gain K_i converges to a steady state value, i.e., K_∞

As shown in Figure 9, the Kalman gain converges quickly. In the 30 seconds simulation time, the gain converges within 0.3 seconds. This shows that, unless the characteristics of the disturbance or measurement noise change with time (which is not the usual case), the steady-state gain can be used without significantly affecting the filter performance.

In the above simulation, the covariance matrix R of the measurement noise is $R = \begin{bmatrix} 0.5 & 0 \\ 0 & 1 \end{bmatrix}$.

Figure 10 shows the result from a simulation run with a higher measurement noise $R = \begin{bmatrix} 5 & 0 \\ 0 & 10 \end{bmatrix}$ while the disturbance remains the same. In this case, the steady state Kalman gain is

$$K = \begin{bmatrix} 0.0306 & -0.0019 \\ -0.0038 & 0.0052 \end{bmatrix} \tag{61}$$

This gain K is smaller than the one in (60). This means that the estimate relies less on the measured state (since it is too noisy) and more on the model. The model, however, does not reflect the effect of the disturbance on the state variable. This can be seen from the lack of a 'bump' on x1hat between t=1.5 and t=2 in Figure 10, unlike the estimate in Figure 8.

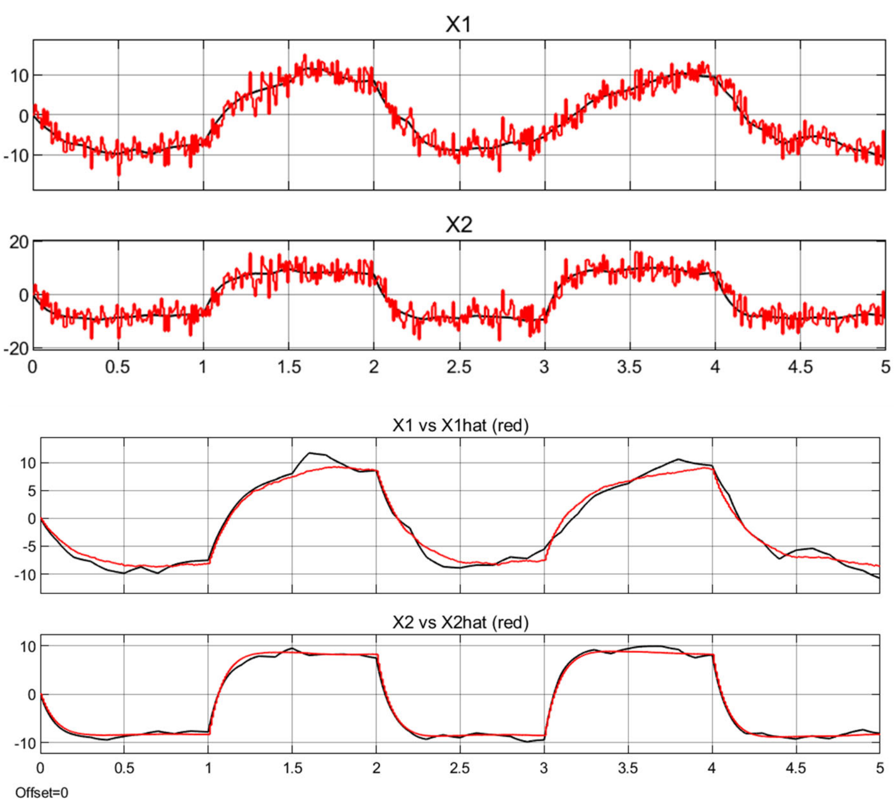


Figure 10 Actual state, measured state, and estimated state with Q=diag(2,1) R=diag(5 10)

Figure 11 shows the result from a simulation run with very small measurement noise $R = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.2 \end{bmatrix}$ while the disturbance remains the same. As expected, since the measurement noise is very small, the Kalman gain can be set to high.

$$K = \begin{bmatrix} 0.3273 & -0.006 \\ -0.0121 & 0.1348 \end{bmatrix} \tag{62}$$

A higher Kalman gain means that the estimate relies heavily on the measurement since the measurement noise is low. As shown in Figure 11, the state estimate tracks the actual state very closely in this case.

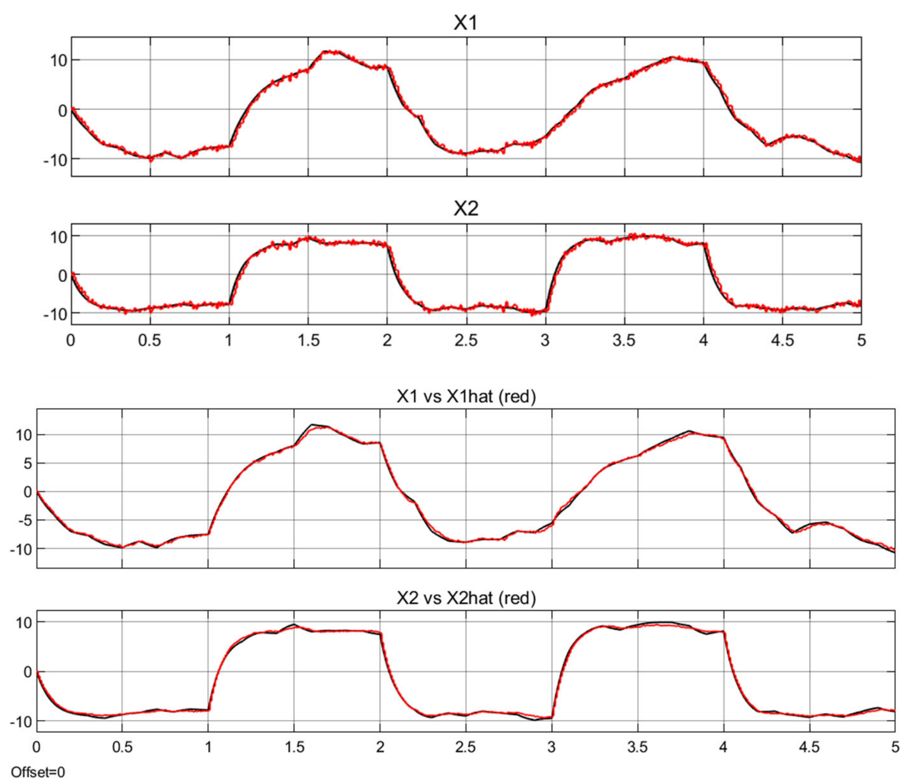


Figure 11 Actual state, measured state, and estimated state with Q=diag(2,1) R=diag(0.1 0.2)

3.5.2 Position tracking example

Figure 12 shows a rocket flying along a straight line. Equation (63), i.e., Newton's Law, can be used to model the dynamic of this one-dimensional motion. A speed dependent resistance force (the drag foce) is modeled by the term '$b\dot{x}$'. u represents the thrust force. Gravitational force is not considered.

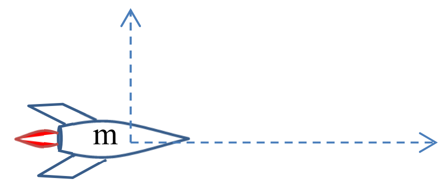


Figure 12

$$m\ddot{x} = -b\dot{x} + u \tag{63}$$

Equation (63) can be arranged into a state equation form (64). The disturbance (ω) and measurement noise (v) are included in this model. In this example, the plant output is measured at every iteration.

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ 0 & -b/m \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} \omega \\ y &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + v \end{aligned} \tag{64}$$

The objective is to estimate the position of the rocket 'x' from knowing the thrust u and a noisy position measurement, y. The thrust is commanded by the on-board computer so it is known. For simplicity, we let m=1 and b=0.01.

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ 0 & -0.01 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \omega \\ y &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + v \end{aligned} \tag{65}$$

The following Matlab script converts the system (65) to the discrete time system (66) at 0.1 second sampling interval. Figure 12 shows the Simulink model of the overall system. The steady state Kalman gain is used in this example.

```
A=[0 1;0 -0.01];
B=[0; 1];
C=[1 0];
D=0;
syst=ss(A,B,C,D);
Dsyst=c2d(syst,0.1)
[dA,dB,dC,dD,Ts]=ssdata(Dsyst)
R=10;
Q=10;
G=dB;
[K,P,Z,E]=dlqe(dA,G,dC,Q,R)
```

$$\begin{bmatrix} x1_{i+1} \\ x2_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & 0.1 \\ 0 & 0.999 \end{bmatrix} \begin{bmatrix} x1_i \\ x2_i \end{bmatrix} + \begin{bmatrix} 0.005 \\ 0.1 \end{bmatrix} u_i + \begin{bmatrix} 0.005 \\ 0.1 \end{bmatrix} \omega$$
$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x1_i \\ x2_i \end{bmatrix} + v \qquad (66)$$

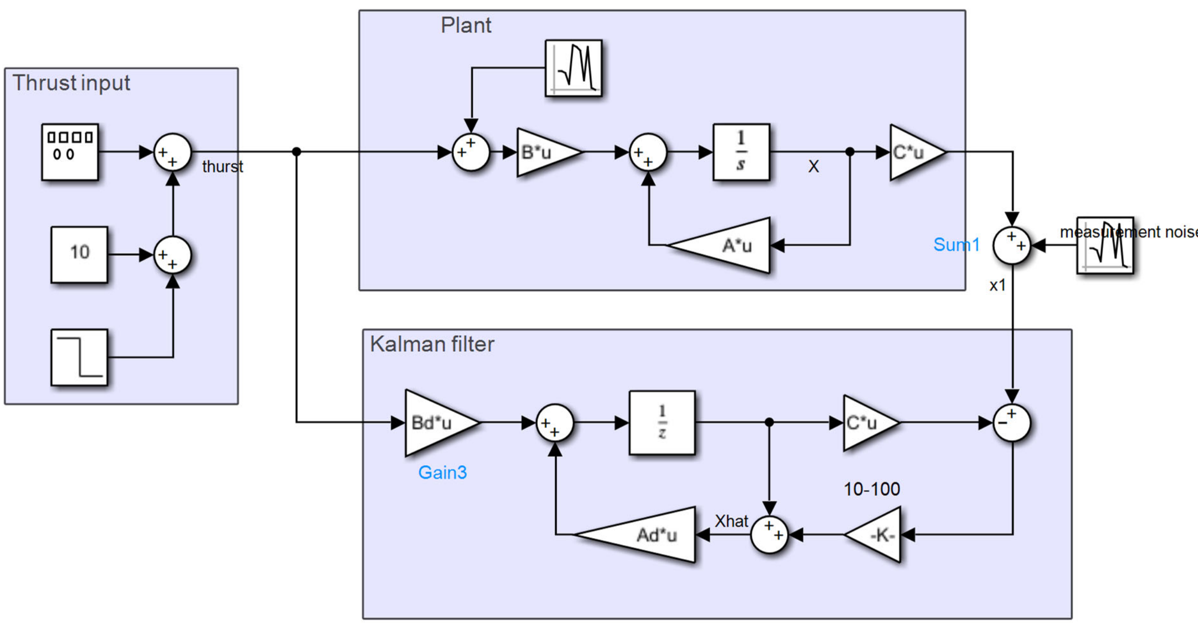


Figure 13

Figure 14 shows the square wave input u (the thrust) and the response of the system (position and speed) under the ideal condition i.e. when there is no disturbance or noise.

Figure 15 is the result of a simulation run with Q=10 and R=10. Figure 15 shows the measured position, the actual speed, the disturbance, and the noise. Note that, unlike the ideal condition in Figure 14 where the position oscillates between 0 and -50, the actual position x (with measurement noise) drifts toward the negative side. This is caused by a small bias in the disturbance ω. This bias simulates the effect of a biased disturbance due to, for example, the wind condition.

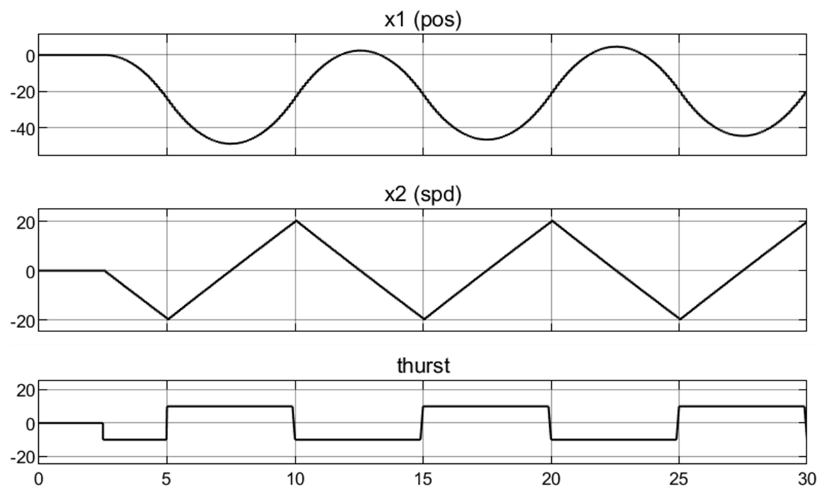


Figure 14 Ideal case plant responses

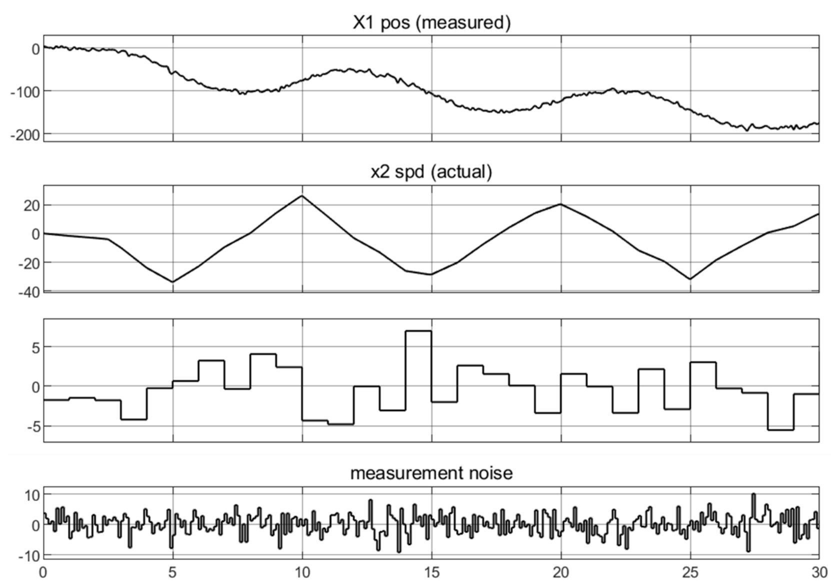


Figure 15 Plant responses with Q=10 (with a small negative bias) and R=10.

The `dlqe` instruction in the script gives the following stead state Kalman gain

$$K = \begin{bmatrix} 0.131 \\ 0.0919 \end{bmatrix}. \tag{67}$$

The first graph in Figure 16 includes three waveforms: the actual x (black), the measured x (red), and the estimated x (green). As shown, the estimated position (green) follows the actual position closely. The estimate error (est err, the middle graph) is within ±5.

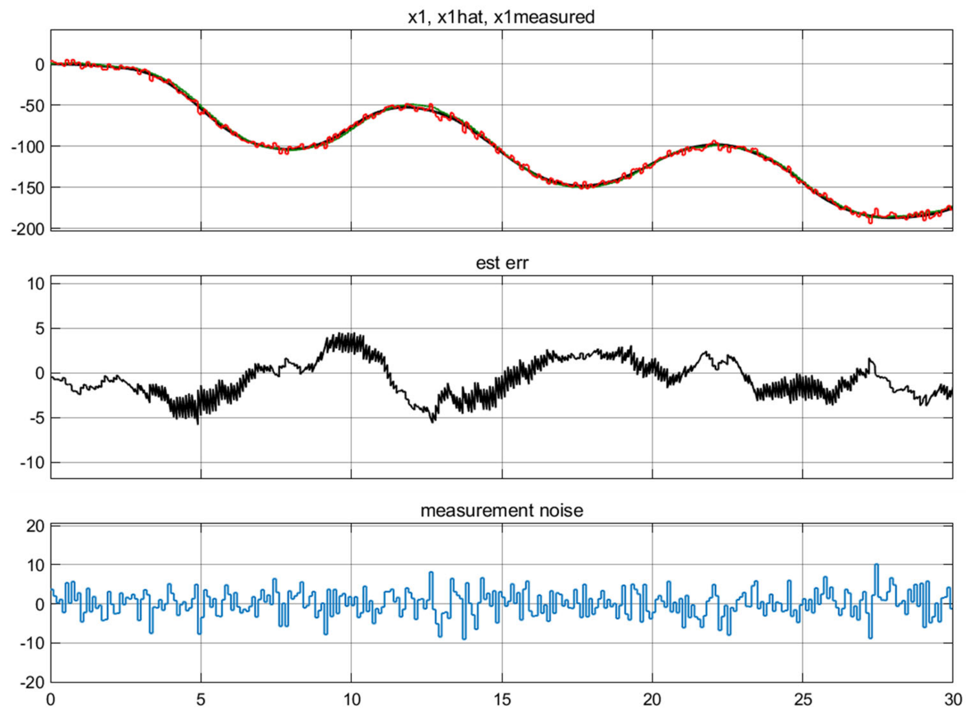


Figure 16 Actual state, measured state and the estimated state with Q=10 and R=10.

Figure 17 show the result of a simulation with a higher measurement noise, Q=10 and R=100. In this case the Kalman gain is

$$K = \begin{bmatrix} 0.0755 \\ 0.0296 \end{bmatrix}. \tag{68}$$

Since the measurement noise is high, the estimate relies less on the measurement (since K is smaller). Although the noise amplitude is more than twice as high as in the case R=Q=10, the estimate error in Figure 17 is only slightly higher than that in Figure 16.

Figure18 show the result of a simulation with a smaller measurement noise, Q=10 and R=1. In this case the Kalman gain is $K = \begin{bmatrix} 0.2215 \\ 0.2768 \end{bmatrix}$. As shown, since the measurement noise is low, the estimate can rely on the measurement and produces a nearly perfect estimate.

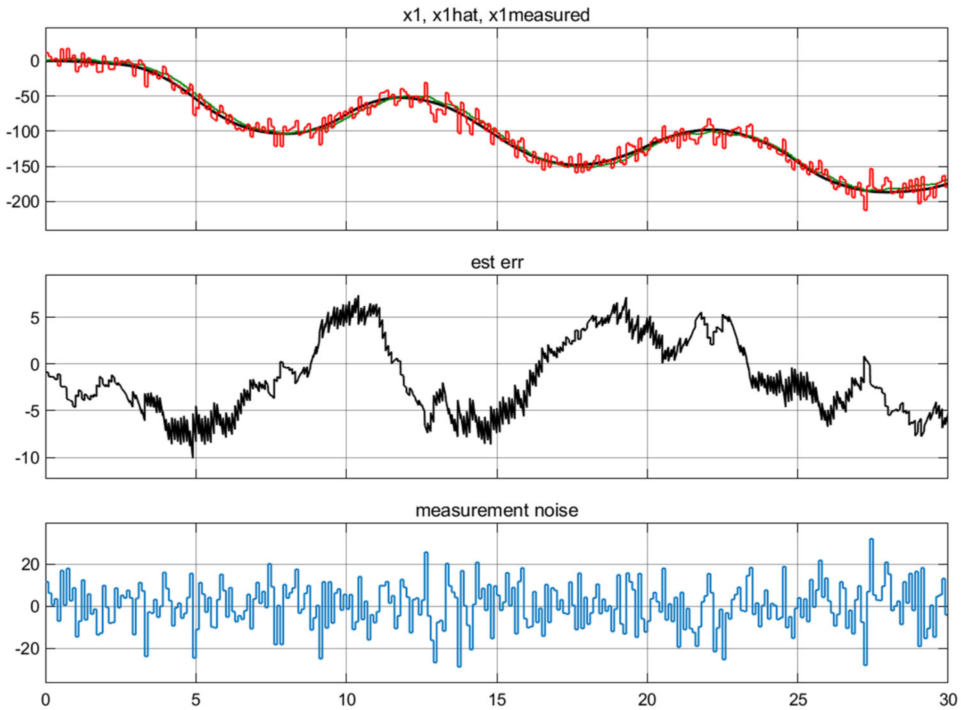


Figure 17 Actual state, measured state and the estimated state with Q=10 and R=100.

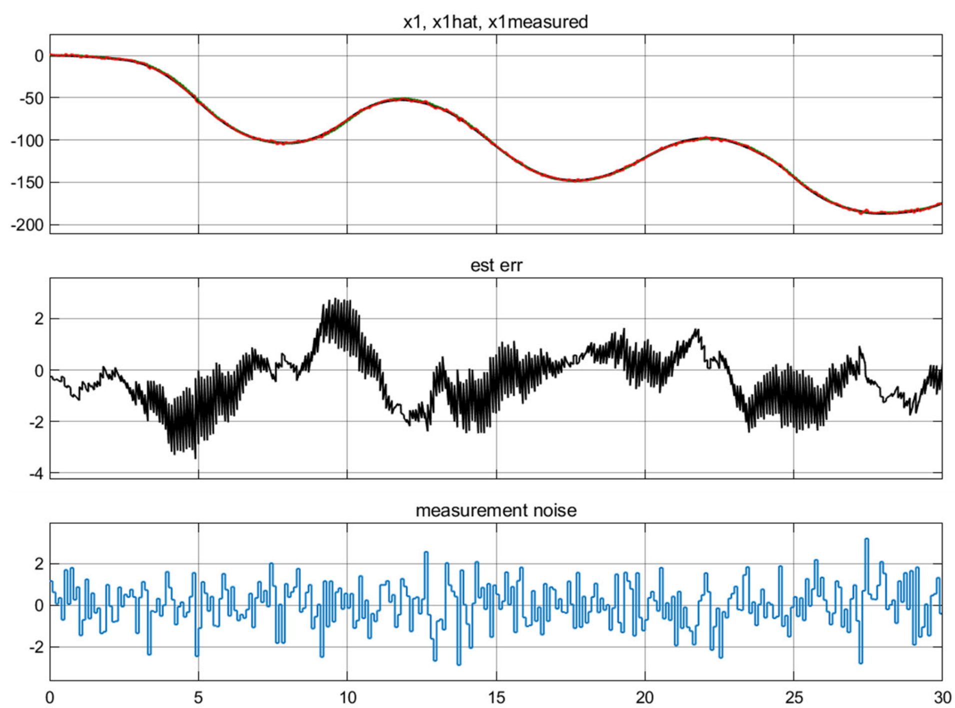


Figure 18 Actual state, measured state and the estimated state with Q=10 and R=1.

3.5.3 A multi-rate accelerometer to position filter example

Before GPS, inertial guidance systems were used widely for providing location information for ships and aircrafts. The onboard accelerometer in an inertial guidance system measures the acceleration without referencing any stationary point. The acceleration is first integrated to produce the speed and integrated again to produce the position information. Over a short time period, such a system provides accurate speed and position information. Over a longer time period, however, the speed and position from the integrators inevitably drift off due to the offset in the accelerometer signal and any small biased in the system. The integrators' output can be corrected whenever the position information is available from, for example, a GPS receiver. In this example, the input to the filter is the acceleration and the outputs are the estimated speed and position. The acceleration signal is measured and integrated by the filter at the sampling frequency of 10Hz. The actual position and speed are measured only every 1 second and are used for correcting the estimator. The structure of the overall system is similar to that in Example 2 and is shown in Figure 19. The model uses two unit-delay blocks at a sampling frequency of 0.1 second (10Hz) for the double integration. The values of these integrators are modified by the innovation term every second when the position and speed are measured. The steady-state Kalman gain is used in this model.

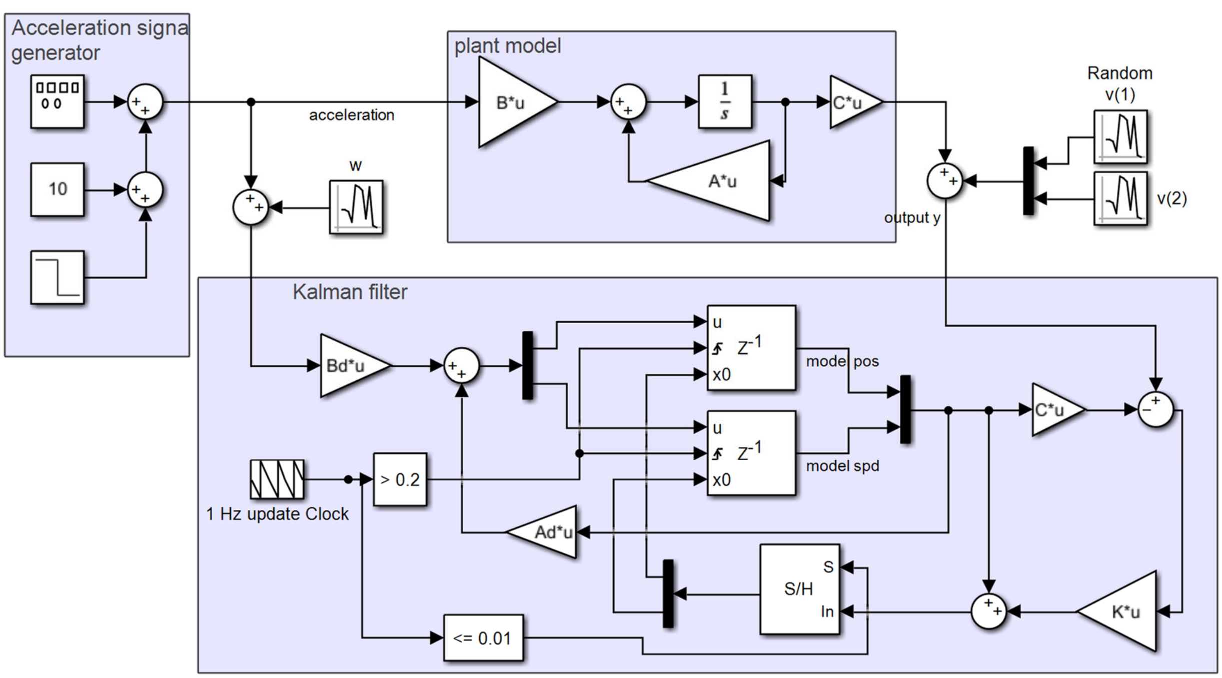


Figure 19

Note that, in this example, the random variable ω is not a disturbance to the plant but rather the acceleration measurement noise. Nevertheless, the same Kalman filter formulation can be used in determining the Kalman gain in this case. This is because the Kalman filter formulation is based on the error between the state and the state estimate. This error can be due to a disturbance or a measurement noise on the input signal. The objective of the Kalman filter algorithm is to minimize this error. The following Matlab script is used for discretizing the plant and for determining the steady-state Kalman gain.

```
A=[0 1;0 0];
B=[0;1];
C=[1 0; 0 1];
D=0;
syst=ss(A,B,C,D);
Dsyst=c2d(syst,0.1)  % 0.1 sec. model sampling freq.
Dsyst1=c2d(syst,1)   % 1 sec. measurement/update freq.
[dA,dB,dC,dD,Ts]=ssdata(Dsyst)
[dA1,dB1,dC1,dD1,Ts]=ssdata(Dsyst1)
R=[10 0; 0 10];
Q=5;
G=dB1;
[K,P,Z,E]=dlqe(dA1,G,dC1,Q,R)
```

The last graph in Figure 20 shows the measured (with noise) and the actual acceleration which is a square wave. The first graph in Figure 20 includes three signals: the actual position (black), the measured position (red), and a filtered measured position (green) by the low-pass filter in (69).

$$H(s)=\frac{1}{s+1}. \tag{69}$$

As shown, the measured position is only updated every one second. This step changes along with the measurement noise are smoothed out by the low-pass filter (69) but the filter introduces a significant amount of time lag. Such a filtered position cannot be used for feedback control purposes because of the time lag. The second graph in Figure 20 shows the same signals (as in the top graph) for speed.

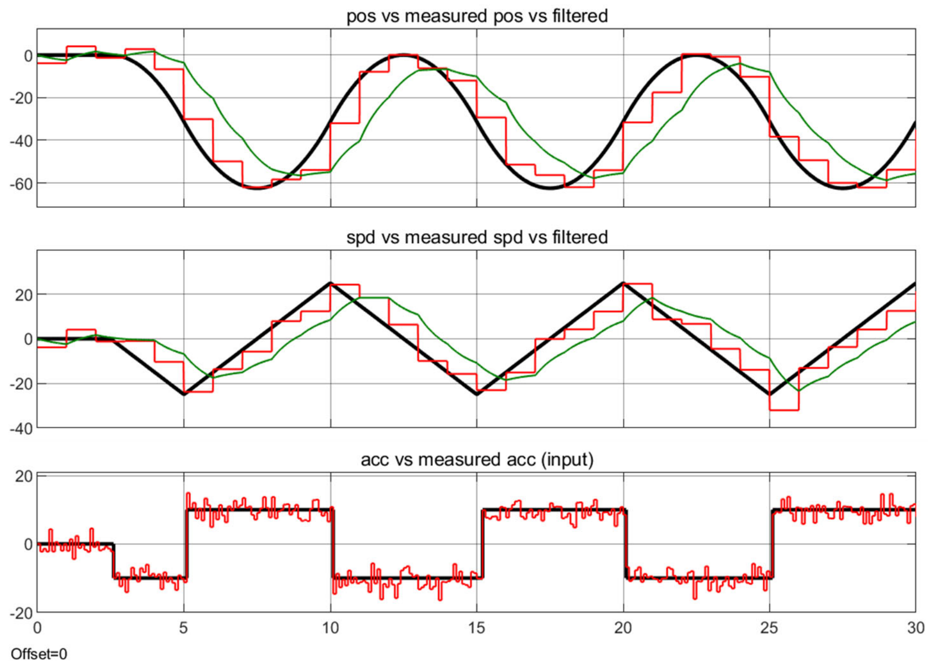


Figure 20 Actual position (black), the measured position (red), and a filtered measured position (green) by the low-pass filter in (69). Q=5 R=diag(10, 10).

Figure 21 shows the results from simply integrating the measured acceleration, i.e., the input. As shown, both the speed and position drift off substantially over a period of 30 seconds. This drift will continue with time.

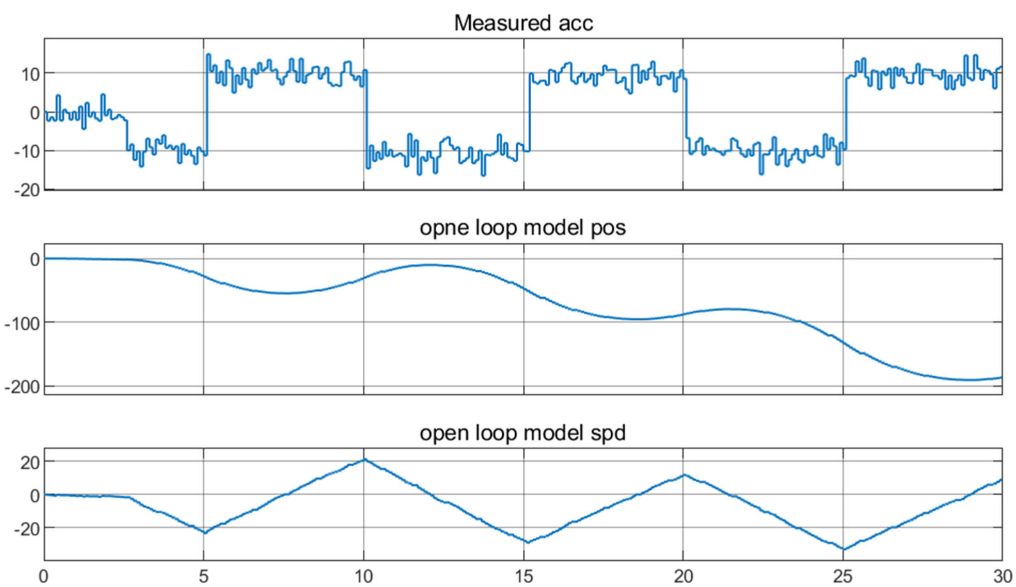


Figure 21 Open loop estimation by integrating the acceleration. Q=5 R=diag(10, 10).

With the noise covariances Q=5 and R=[10 0; 0 10], the steady state Kalman gain produced y the **dlqe** instruction is

$$K = \begin{bmatrix} 0.516 & 0.2158 \\ 0.2158 & 0.3685 \end{bmatrix}. \tag{70}$$

Note that, the G matrix used in the **dlqe** instruction is determined with the sampling rate of 1 second, not the 0.1 section for the discrete time plant model. Figure 22 shows the output of the Kalman filter (in red). Note that the estimate state follows the actual state between measurements but undergoes a jump every second due to the update made by the Kalman filter.

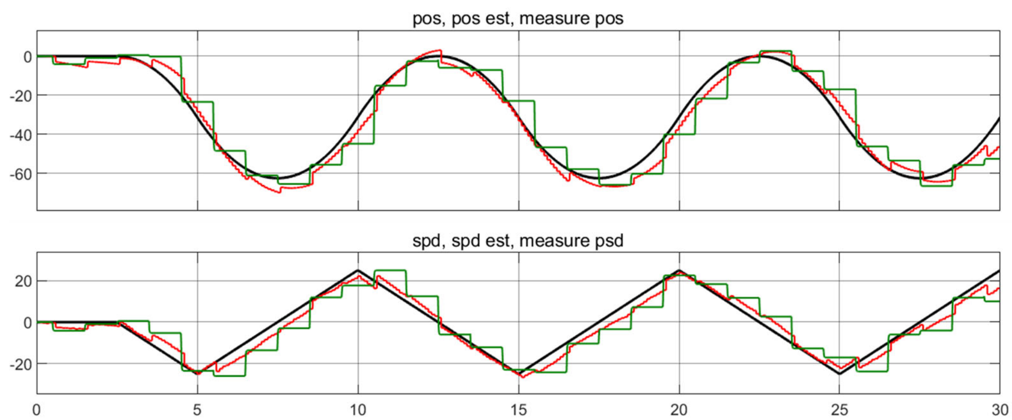


Figure 22 Actual state (black), Output of the Kalman filter (red), measurement (green). Q=5 R=diag(10, 10).

Figure 23 is a zoom-in view of Figure 22. At t=12.5, the estimated position (red) is higher than the actual position (black) and the measured position is lower than the actual position due to the noise. At this point, the position estimate was updated but it didn't correct all the way to match the measured value (since measured value contains noise). At t=13.5, the estimate was lower than the actual and the measurement error is smaller this time. The estimate was corrected once again toward the measured value. A very important point to see from this figure is that the curvature of the position estimate between t=12.5 and t=13.5, follows the actual position closely, this is due to the model in the Kalman filter and the measured acceleration.

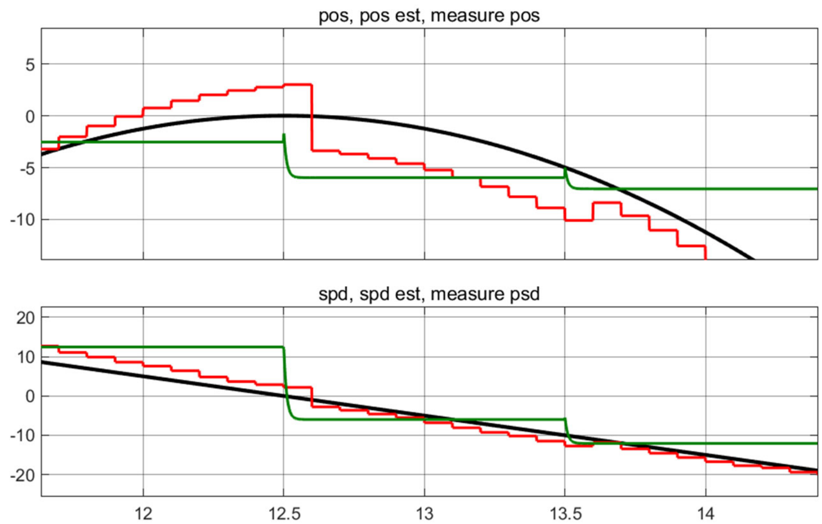


Figure 23 A zoomed-in view of Figure 22.

As shown in Figure 22, the estimate error is quite noticeable. This is unavoidable due to the position measurement noise. If the noise is higher, this error will be even higher as shown in the next simulation where R=[100 0; 0 100] while Q stays the same Q=5. For this higher noise case, the steady state Kalman gain is

$$K = \begin{bmatrix} 0.4268 & 0.1288 \\ 0.1288 & 0.106 \end{bmatrix}. \tag{71}$$

Figure 24 shows the result of this simulation. As expected, the estimate error is higher.

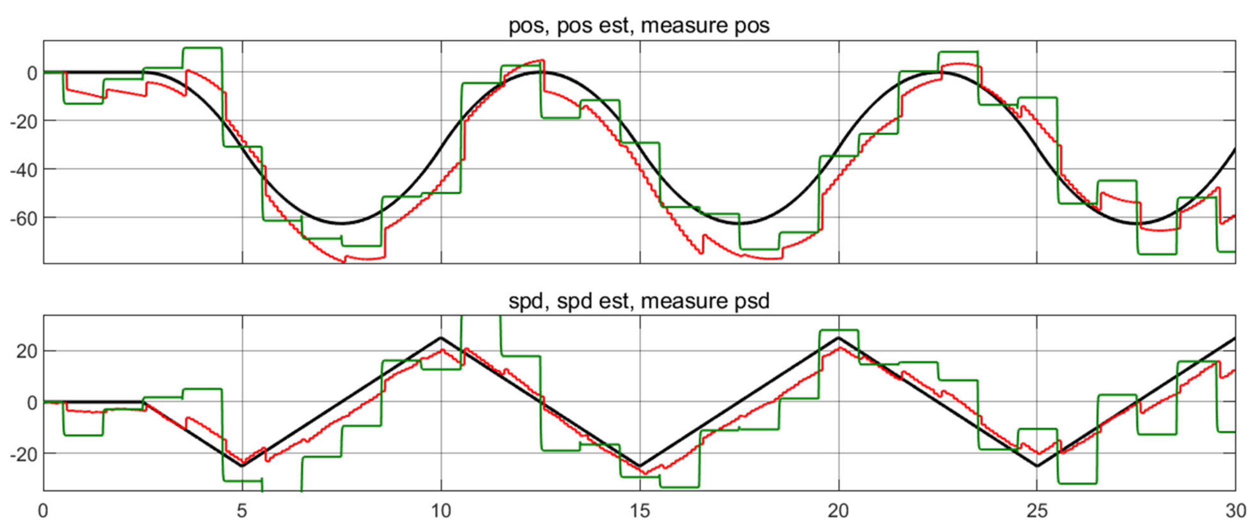


Figure 24 Actual state (black), Output of the Kalman filter (red), measurement (green). Q=5 R=diag(100, 100).

We now consider the case where the position measurement is small, R=[1 0; 0 1] while Q stays the same Q=5.   In this higher noise case, the steady state Kalman gain is

$$K = \begin{bmatrix} 0.5479 & 0.2326 \\ 0.2326 & 0.7317 \end{bmatrix}. \tag{72}$$

Figure 25 show the simulation result. As expected, the estimated state follows the actual one closely.

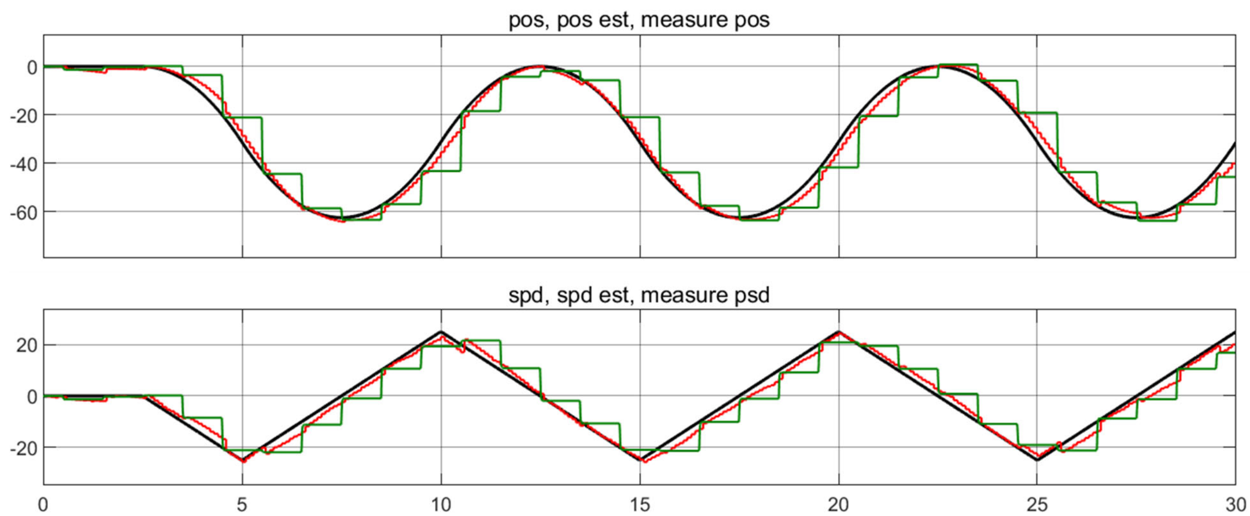


Figure 25 Actual state (black), Output of the Kalman filter (red), measurement (green).   Q=5 R=diag(1, 1).