Discrete-time Kalman Filter and the Particle Filter*

(Com S 477/577 Notes)

Yan-Bin Jia

Nov 17, 2022

1 Discrete-Time System

A discrete-time system is a process that transforms input discrete-time signals into output discrete-time signals. Simply stated, it takes an input sequence and produces an output sequence. Such a system usually takes the form of

\[ x_k = F_{k-1}x_{k-1} + G_{k-1}u_{k-1} + w_{k-1}, \tag{1} \]

where the \( n \)-vectors \( x_k \) and \( x_{k-1} \) are the states at the current and previous time steps, the \( l \)-vector \( u_k \) is a known input, and the \( n \)-vector \( w_k \) is white noise with zero mean and covariance \( Q_k \). The matrices \( F_{k-1} \) and \( G_{k-1} \) have dimensions \( n \times n \) and \( n \times l \), respectively. We are interested in how the mean of the state \( x_k \) evolves with time.

From now on, we denote the mean of a random variable with an overbar \( \bar{\cdot} \). Take the expected value of both sides of equation (1):

\[ \bar{x}_k = E(x_k) = F_{k-1}\bar{x}_{k-1} + G_{k-1}u_{k-1}. \tag{2} \]

How does the covariance of \( x_k \) evolve with time? Subtraction of (2) from (1) yields

\[ x_k - \bar{x}_k = F_{k-1}(x_{k-1} - \bar{x}_{k-1}) + w_{k-1}. \]

The covariance \( P_k \) as an \( n \times n \) matrix is then computed as follows:

\[
P_k = E((x_k - \bar{x}_k) (x_k - \bar{x}_k)^T) \\
= E\left((F_{k-1}(x_{k-1} - \bar{x}_{k-1}) + w_{k-1}) (F_{k-1}(x_{k-1} - \bar{x}_{k-1}) + w_{k-1})^T\right) \\
= E\left(F_{k-1}(x_{k-1} - \bar{x}_{k-1}) (x_{k-1} - \bar{x}_{k-1})^T F_{k-1}^T + w_{k-1} w_{k-1}^T + F_{k-1}(x_{k-1} - \bar{x}_{k-1}) w_{k-1}^T + w_{k-1}(x_{k-1} - \bar{x}_{k-1})^T F_{k-1}^T\right).
\]

That \( x_{k-1} - \bar{x}_{k-1} \) and \( w_{k-1} \) are uncorrelated implies

\[
E((x_{k-1} - \bar{x}_{k-1}) w_{k-1}^T) = E(x_{k-1} - \bar{x}_{k-1}) E(w_{k-1}^T) = 0, \quad \text{(since } E(w_{k-1}^T) = 0)\]

*The material is adapted from Chapters 4.1, 5.1–5.3, and 15 in Dan Simon’s book Optimal State Estimation [1].
and thus

\[ E \left( w_{k-1} (x_{k-1} - \bar{x}_{k-1})^T \right) = 0. \]

Two of the summands in the equation for \( P_k \) vanish. The recursive covariance equation reduces to

\[ P_k = F_{k-1} P_{k-1} F_{k-1}^T + Q_{k-1}, \tag{3} \]

where \( Q_{k-1} = E(w_{k-1} w_{k-1}^T) \) is the covariance of \( w_{k-1} \). The above is called a discrete-time Lyapunov equation or a Stein equation.

**Example 1.** The following is a linear system that describes the population of a predator \( x_1 \) and that of its prey \( x_2 \), where the second subscript of each item denotes the time step:

\[
\begin{align*}
    x_{1,k+1} &= x_{1,k} - 0.8x_{1,k} + 0.4x_{2,k} + w_{1,k}, \\
    x_{2,k+1} &= x_{2,k} - 0.4x_{1,k} + u_k + w_{2,k}.
\end{align*}
\]

The first equation says that the predator population decreases due to overcrowding as well as the reduction in the prey population. The second equation says that the prey population decreases due to the predator population and increases on external food supply \( u_k \). Both populations are subject to random disturbances (with variances 1 and 2, respectively) due to environmental factors. We rewrite the system in the state space form as

\[
\begin{align*}
    x_{k+1} &= \begin{pmatrix} 0.2 & 0.4 \\ -0.4 & 1 \end{pmatrix} x_k + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u_k + w_k, \\
    w_k &\sim (0, Q),
\end{align*}
\]

where \( Q = \text{diag}(1, 2) \).

The mean and covariance of the populations evolve with time according to equations (2) and (3). Set \( u_k = 1 \) and the initial conditions as \( \bar{x}_0 = (10, 20)^T \) and \( P_0 = \text{diag}(40, 40) \). The figure\(^1\) below depicts the two means and the two diagonal elements of the covariance matrix for the first few steps.

---

\(^1\)Figure 4.1 on p. 110 in [1].
From the figure, we see that the mean and covariance eventually reach the steady-state values. The steady state value of the mean is the solution of the linear system after setting $w_k = 0$:

$$\bar{x} = \begin{pmatrix} 2.5 \\ 5 \end{pmatrix}.$$  

The steady-state value of $P$ found by control system software\footnote{For example, the MATLAB Control System Toolbox function DLYAP(F, Q).} is

$$P \approx \begin{pmatrix} 2.88 & 3.88 \\ 3.08 & 7.96 \end{pmatrix}.$$  

### 2 Four Estimates of a State

We would like to estimate the state, i.e., the values of all variables of a dynamic system as it evolves with time. Our method is to derive equations from the system to describe how the mean and covariance of its state $\mathbf{x}$ propagate with time. These equations form a new dynamic system which is simulated on a computer. They are the basis for the derivation of the Kalman filter whose estimate is the mean of the original system’s state $\mathbf{x}$, and whose covariance equals the covariance of $\mathbf{x}$. Every time a new measurement is taken, the mean and covariance of $\mathbf{x}$ are updated, in a similar way we recursively update the estimate of a constant before.

Consider a linear discrete-time system as follows:

$$\mathbf{x}_k = F_{k-1}\mathbf{x}_{k-1} + G_{k-1}\mathbf{u}_{k-1} + \mathbf{w}_{k-1},$$  

$$\mathbf{y}_k = H_k\mathbf{x}_k + \mathbf{v}_k.$$  

Here, the evolution of the state $\mathbf{x}$ and the output $\mathbf{y}$ are subject to some noise processes $\mathbf{w}$ and $\mathbf{v}$, respectively. These two noise are white, zero-mean, and uncorrelated. Let $Q_k$ and $R_k$ be their covariance matrices, respectively. Then the noise characteristics are given as follows:

$$\mathbf{w}_k \sim (0, Q_k),$$

$$\mathbf{v}_k \sim (0, R_k),$$

$$E(\mathbf{w}_k\mathbf{w}_j^T) = \begin{cases} Q_k, & \text{if } k = j; \\ 0, & \text{otherwise}, \end{cases}$$

$$E(\mathbf{v}_k\mathbf{v}_j) = \begin{cases} R_k, & \text{if } k = j; \\ 0, & \text{otherwise}, \end{cases}$$

$$E(\mathbf{v}_k\mathbf{w}_j) = 0.$$  

The goal is to estimate the state $\mathbf{x}_k$ based on the known system dynamics (4) and (5) and from the noisy measurements $\{\mathbf{y}_k\}$.

In estimating the state $\mathbf{x}_k$ at time $k$, if we have all the measurements up to and including time $k$ available, then we can form an \textit{a posteriori} (or a posterior) estimate denoted as $\hat{\mathbf{x}}_k^+$. One way is to compute the expected value of $\mathbf{x}_k$ as

$$\hat{\mathbf{x}}_k^+ = E(\mathbf{x}_k | \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k).$$
If we have only the measurements before time \( k \) available, then we can form \textit{a priori} (or a prior) estimate denoted as \( \hat{x}_k^- \). One way to form this latter estimate is to compute the expected value of \( x_k \) as

\[
\hat{x}_k^- = E(\hat{x}_k \mid y_1, y_2, \ldots, y_{k-1}).
\]

(12)

Note that \( \hat{x}_k^- \) and \( \hat{x}_k^+ \) are both estimates of \( x_k \). However, the former is our estimate before the measurement \( y_k \) is taken into account, while the latter is our estimate after the measurement \( y_k \) is taken into account. With more information, \( \hat{x}_k^+ \) is expected to be a better estimate than \( \hat{x}_k^- \).

A \textit{smoothed estimate} of \( x_k \) is formed if we also have measurements after time \( k \) available for use. One way of doing this is to compute the following expected value:

\[
\hat{x}_k|_{k+s} = E(\hat{x}_k \mid y_1, y_2, \ldots, y_{k+s}),
\]

(13)

where \( s > 0 \) depends on the specific problem that is being solved. A \textit{predicted estimate} is a prediction of \( x_k \) one step or more ahead of the available measurements. This is often computed as the expected value of \( x_k \) based on the limited measurements that are available:

\[
\hat{x}_{k|k-t} = E(\hat{x}_k \mid y_1, y_2, \ldots, y_{k-t}),
\]

(14)

where \( t > 0 \) is problem specific.

The figure below depicts the relationship between the posterior, prior, smoothed, and predicted state estimates. Suppose we are making estimates based on all measurements taken at time \( k \leq 5 \). An estimate of the state at \( k = 1 \) is a smoothed estimate. An estimate at \( k = 5 \) is the \textit{a posteriori} estimate. An estimate at \( k = 6 \) is the prior estimate. And an estimate of the state at \( k = 9 \) is a prediction.

Starting from now we denote \( \hat{x}_0^+ \) as our initial estimate of \( x_0 \) before the first measurement is taken (at time \( k = 1 \)). Given no measurements available, we form \( \hat{x}_0^+ \) as the expected value of the initial estimate:

\[
\hat{x}_0^+ = \hat{x}_0 = E(x_0).
\]

(15)

We let \( P_k^- \) be the covariance of the estimation error of \( \hat{x}_k^- \), and \( P_k^+ \) the covariance of the estimation error of \( \hat{x}_k^+ \), all at time \( k \):

\[
P_k^- = E((x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T),
\]

\[
P_k^+ = E((x_k - \hat{x}_k^+)(x_k - \hat{x}_k^+)^T).
\]
The next figure shows the relationships from time \( k - 1 \) to \( k \). After we process the measurement at time \( k - 1 \), we have a *posteriori* estimate \( \hat{x}_{k-1}^+ \) and the covariance \( P_{k-1}^+ \). At time \( k \), before we process the new measurement we compute a prior estimate \( \hat{x}_k^- \) and the covariance \( P_k^- \) of the estimate based on the system dynamics. Then we process the measurement \( y_k \) at time \( k \) to refine our estimate of \( x_k \). The resulting estimate that takes into account the measurement at time \( k \) is \( \hat{x}_k^+ \) with covariance \( P_k^+ \).

3 From *a posteriori* Estimate to *a priori* Estimate

The estimation process starts with \( \hat{x}_0^+ \), the best estimate of the initial state \( x_0 \). It then computes \( \hat{x}_0^- \). Recall from (2) that the mean of \( x \) propagates with time, so we obtain

\[
\hat{x}_0^- = F_0 \hat{x}_0^+ + G_0 u_0.
\]

This equation generalizes to obtain the prior estimate at time \( k \) from the posterior estimate at the previous time \( k - 1 \):

\[
\hat{x}_k^- = F_{k-1} \hat{x}_{k-1}^+ + G_{k-1} u_{k-1}.
\]

Namely, the state estimate propagates the same way that the mean of the state does. This update is based on the system dynamics (1).

Next, we derive the equations for updating the covariance of the state estimation error from time \( k - 1 \) to time \( k \) but before the measurement at time \( k \) is considered. We begin with \( P_0^+ \), the covariance of our initial estimate \( \hat{x}_0^+ \). The covariance represents the uncertainty in \( \hat{x}_0 \):

\[
P_0^+ = E\left((x_0 - \hat{x}_0)(x_0 - \hat{x}_0)^T\right) = E\left((x_0 - \hat{x}_0^+)(x_0 - \hat{x}_0^+)^T\right).
\]

If the initial state \( x_0 \) is perfectly known, then \( P_0^+ = 0 \); if it is completely unknown, then \( P_0^+ = \infty I \). The covariance of the state propagates with time according to (3), which is now rewritten as

\[
P_k^- = F_{k-1} P_{k-1}^+ F_{k-1}^T + Q_{k-1}.
\]

In particular, we use

\[
P_1^- = F_0 P_0^+ F_0^T + Q_0.
\]

for initialization.

4 From *a priori* Estimate to *a posteriori* Estimate

Now we need to update the state estimate at time \( k \) by taking into account the measurement \( y_k \) at time \( k \). In other words, we compute \( \hat{x}_k^+ \) from \( \hat{x}_k^- \), which was obtained without knowledge of \( y_k \).
Freeze the time at $k$ temporarily and imagine that a sequence of observations $y'_1, \ldots, y'_{l-1}$ have already been made, all at the time instant. Recall from the lecture on recursive least squares that the availability of a new observation $y'_l$ changes the estimate as follows:

\[
K'_l = P'_{l-1}H'^T_l (H'_l P'_{l-1} H'^T_l + R'_l)^{-1},
\]
\[
\hat{x}'_l = \hat{x}'_{l-1} + K'_l (y'_l - H'_l \hat{x}'_{l-1}),
\]
\[
P'_l = (I - K'_l H'_l) P'_{l-1},
\]

where $K', P', H'$, and $R'$ respectively carry the same meanings of $K$, $P$, $H$, and $R$ with their subscripts indicating the number of observations (hypothetically) made at the same time instant $k$. In particular, $\hat{x}'_{l-1} = \hat{x}^*_k$ is the estimate before the measurement $y'_l = y_k$ is processed, and $\hat{x}'_l = \hat{x}^*_k$ is the estimate based on $y_k$. The full correspondences between the terms at the $l$th iteration of the recursive least squares and those at the (current) $k$th step of estimation are shown below:

\[
\hat{x}'_{l-1} \leftrightarrow \hat{x}^-_k \quad P'_{l-1} \leftrightarrow P^-_k \quad \hat{x}'_l \leftrightarrow \hat{x}^+_k \quad P'_l \leftrightarrow P^+_k \quad y'_l \leftrightarrow y_k \quad K'_l \leftrightarrow K_k \quad H'_l \leftrightarrow H_k \quad R'_l \leftrightarrow R_k.
\]

Applying these replacements, we obtain the measurement update equations:

\[
K_k = P^*_k H'^T_k (H_k P^*_k H'^T_k + R_k)^{-1},
\]
\[
\hat{x}^+_k = \hat{x}^-_k + K_k (y_k - H_k \hat{x}^-_k),
\]
\[
P^+_k = (I - K_k H_k) P^*_k.
\]

## 5 Kalman Filter

Now we summarize all the equations into an algorithm referred to as the discrete-time Kalman filter. The system dynamics are given by (4) and (5) along with noise statistics (8)–(10).

1. Initialize the Kalman filter according to (15) and (16), namely,

\[
\hat{x}^+_0 = E(x_0) = \bar{x}_0,
\]
\[
P^+_0 = E ((x - \bar{x}_0)(x_0 - \bar{x}_0)^T).
\]

2. Iterate the following sequentially at each time step $k = 1, 2, \ldots$:

\[
P^-_k = F_{k-1} P^+_{k-1} F'^T_{k-1} + Q_{k-1},
\]
\[
\hat{x}^-_k = F_{k-1} \hat{x}^+_{k-1} + G_{k-1} u_{k-1}; \quad \text{(a priori state estimate)}
\]
\[
K_k = P^-_k H'^T_k (H_k P^-_k H'^T_k + R_k)^{-1},
\]
\[
\hat{x}^+_k = \hat{x}^-_k + K_k (y_k - H_k \hat{x}^-_k); \quad \text{(a posteriori state estimate)}
\]
\[
P^+_k = (I - K_k H_k) P^-_k (I - K_k H_k)^T + K_k R_k K^T_k
\]
\[
= (I - K_k H_k) P^-_k.
\]
Updates of the state estimate $\hat{x}_{k}^{-}$ in (20) and a priori covariance $P_{k}^{-}$ in (19) are according to the system dynamics described by (1). Those of the gain matrix in (21), the a posteriori state estimate $\hat{x}_{k}^{+}$ in (22), and the a posteriori covariance $P_{k}^{+}$ in (23) and (24) are through recursive least-squares optimization of $E(\|x - \hat{x}_{k}^{-}\|^2)$.

When $x_k$ is a constant, then $F_k = I$, the covariance matrix for noise $Q_k = 0$, and the input $u_k = 0$ for all $k$. So we have, by (19)–(20), (22), (24),

$$P_{k}^{-} = P_{k-1}^{+},$$
$$\hat{x}_{k}^{-} = \hat{x}_{k-1}^{+},$$
$$\hat{x}_{k}^{+} = \hat{x}_{k-1}^{+} + K_k (y_k - H_k \hat{x}_{k}^{-}),$$
$$P_{k}^{+} = (I - K_k H_k) P_{k-1}^{+}.$$

The Kalman filter consists of the last two equations above and (21). It reduces to the recursive least squares algorithm for estimating a constant vector.

Note that the calculation of $P_{k}^{-}$, $K_k$, and $P_{k}^{+}$ does not depend on the measurement $y_k$. It depends on the system parameters $F_k, H_k, Q_k$, and $R_k$. For efficiency, we can precalculate the Kalman gain $K_k$ up to large enough $k$ and store the data beforehand. When the dynamical system starts operating, only $\hat{x}^-$ and $\hat{x}^+$ need to be computed in real time. This also means that the performance of the filter can be investigated and evaluated before it is actually run. The reason is that the accuracy of the filter is indicated by $P_k$, which does not depend on the measurements.

**Example 2.** Consider a noise-free Newtonian system involving position $r$, velocity $v$, and constant acceleration $a$. Here, $r, v, a \in \mathbb{R}^3$. The system equation is

$$\begin{pmatrix}
\dot{r} \\
\dot{v} \\
\dot{a}
\end{pmatrix} = \begin{pmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
0 & 0 & 0
\end{pmatrix} \begin{pmatrix}
r \\
v \\
a
\end{pmatrix},$$

or rewritten succinctly as

$$\dot{x} = Ax.$$

The system has a solution $x = e^{At}x_0$, where

$$e^{At} = \sum_{j=0}^{\infty} \frac{(At)^j}{j!}.$$

Discretize the system with a sample time interval of $h$: $x_{k+1} = Fx_k$, where

$$F = e^{Ah} = I + Ah + \frac{(Ah)^2}{2!} + \cdots \approx \begin{pmatrix}
1 & h & \frac{h^2}{2} \\
0 & 1 & h \\
0 & 0 & 1
\end{pmatrix}.$$

There is no state noise or output noise. Hence $Q_k = 0$ and $R_k = 0$ for all $k$. Also, there is no input so $G_k = 0$. The Kalman filter for the system follows from equations (19) and (20):

$$P_{k}^{-} = FP_{k-1}^{+}F^T,$$
$$\hat{x}_{k}^{-} = F \hat{x}_{k-1}^{+}.$
The equations (21)–(24) are not needed since we do not make any measurements between time \( k^- \) and time \( k^+ \). For the same reason, we see that the covariance of the estimation error increases between time \((k-1)^+\) and time \( k^- \), as indicated in the first equation above.

Now suppose that we measure the position with a variance of \( \sigma^2 \):

\[
y_k = H_k x_k + v_k = (1 \ 0 \ 0) x_k + v_k,
\]

\[
v_k \sim (0, R_k),
\]

\[
R_k = \sigma^2.
\]

We denote by \( P_{k,ij}^- \) the \((i, j)\)-th entry of the prior covariance. Substituting \( H_k = (1, 0, 0) \) and \( R_k = \sigma^2 \) into (21), we obtain the Kalman gain:

\[
K_k = \frac{1}{P_{k,11}^- + \sigma^2} \begin{pmatrix} P_{k,11}^- \\ P_{k,12}^- \\ P_{k,13}^- \end{pmatrix}.
\]

Meanwhile, the posterior covariance follows from (24) as

\[
P_k^+ = P_k^- - K_k H_k P_k^-.
\]

We obtain its traces:

\[
\text{tr}(P_k^+) = \text{tr}(P_k^-) - \frac{(P_{k,11}^-)^2 + (P_{k,22}^-)^2 + (P_{k,33}^-)^2}{P_{k,11}^- + \sigma^2}.
\]

The above equation shows that the covariance decreases when we get a new measurement. Consequently, our state estimate improves.

The second figure shows the variance evolution when the Kalman filter runs for the first 60 time steps. It can be seen that, despite the fluctuations before and after each new measurement, the variance converges to a steady-state value.

4This and the next figures appear as Figures 5.3 and 5.4 on p. 134 of [1].
The next figure\(^5\) shows the position-measurement error (with a standard deviation of 30) and the error of the posterior estimate \(\hat{x}^+\). The estimate error starts out with a standard deviation close to 30, and decreases to about 11 by the end of the simulation. The standard deviations are obtained from the covariances.

6 One-step Kalman Filter

This section combines the \textit{a priori} and \textit{a posteriori} Kalman filter equations into one. At time \(k+1\), the prior state estimate is, by (20),

\[
\hat{x}_{k+1} = F_k \hat{x}_k^+ + G_k u_k.
\]

Now, we substitute (22) into the above and obtain

\[
\hat{x}_{k+1}^- = F_k \left( \hat{x}_k^- + K_k (y_k - H_k \hat{x}_k^-) \right) + G_k u_k
\]

\[
= F_k (I - K_k H_k) \hat{x}_k^- + F_k K_k y_k + G_k u_k.
\]

\(^5\)Figure 5.5 on p. 135 of [1].
Similarly, we can obtain a one-step expression for the prior covariance. First, increase the index in (19) by one:

\[ P_{k+1}^- = F_k P_k^- F_k^T + Q_k. \]

Next, substitute (24) into the above:

\[
\begin{aligned}
P_{k+1}^- &= F_k (P_k^- - K_k H_k P_k^-) F_k^T + Q_k \\
&= F_k P_k^- F_k^T - F_k K_k H_k P_k^- F_k^T + Q_k.
\end{aligned}
\]  

(28)

The Kalman filter at each iteration step sequentially executes (21), (27), and (28).

Similarly, we can derive one-step expressions for the a posterior state estimate and covariance as follows:

\[
\begin{aligned}
\hat{x}_{k}^+ &= (I - K_k H_k)(F_{k-1} \hat{x}_{k-1}^+ + G_{k-1} u_{k-1}) + K_k y_k, \\
P_{k}^+ &= (I - K_k H_k)(F_{k-1} P_{k-1}^+ F_{k-1}^T + Q_{k-1}).
\end{aligned}
\]  

(29)  
(30)

The Kalman filter sequentially evaluates (21), (29), and (30) at each iteration step.

7 The Particle Filter

The Kalman filter, given its linear nature, has difficulty solving estimation problems that typically involve highly nonlinear systems. For such systems, the extended Kalman filter (EKF) is the most widely applied state estimation technique. However, when the nonlinearities are high, the EKF often gives unreliable estimates since it employs linearization to propagate the mean and covariance of the state estimate. The particle filter is a statistical, brute-force approach to estimation that often works well in such a situation.

The particle filter is a nonlinear state estimator that trades computational effort for the degree of nonlinearity. A probability-based estimator, it has proven very effective in machine learning, mapping, and learning robots [2].

At the beginning of estimation, we randomly generate a number of state vectors based on the probability distribution of the initial state (which is assumed to be known). These state vectors are called particles. At each time step, we propagate the particles to the next time step using the process equation. Compute the likelihood of each particle based on the measurement taken at the step. Then generate a set of new particles based on the computed relative likelihoods. Iterate for a number of steps. Output the mean and covariance of the particles at the last step.

More formally, suppose the nonlinear system and measurement equations are given as follows for the time step \( k \):

\[
\begin{aligned}
x_{k+1} &= f_k(x_k, w_k), \\
y_k &= h_k(x_k, v_k).
\end{aligned}
\]

where \( w_k \) and \( v_k \) are the process noise and measurement noise, respectively, with known probability density functions (pdf’s). The functions \( f_k \) and \( h_k \) vary with time (i.e., the time step). The pdf \( p(x_0) \) of the initial state is assumed to be known.

The particle filtering algorithm executes the following steps:
1. Randomly generate \( n \) initial particles based on \( p(x_0) \), and denote them as \( x_0^+ \), \( i = 1, \ldots, n \). The parameter \( n \) is chosen as a trade-off between computational effort and estimation accuracy.

2. Repeat the following for \( k = 1, 2, \ldots, \).

**Propagation** For each particle \( x_{k-1,i}^+ \) perform the time propagation step to obtain *a priori* particles \( x_{k,i}^- \), based on the system dynamics:

\[
x_{k,i}^- = f_{k-1}(x_{k-1,i}^+, w_{k-1}^i), \quad \text{for } i = 1, \ldots, n,
\]

where each noise vector \( w_{k-1}^i \) is randomly generated based on the known pdf of \( w_{k-1} \).

**Likelihood determination** Compute the relative likelihood \( q_i \) of each particle \( x_k^i \), conditioned on the measurement \( y_k \). This is done by evaluating the pdf \( p(y_k \mid x_{k,i}^-) \), since we know the pdf of the measurement noise. For instance, if an \( m \)-dimensional measurement is given as

\[
y_k = h(x_k) + v_k
\]

where \( v_k \sim N(0, R) \). Then the relative likelihood of a specific measurement \( y^* \) can be computed as follows:

\[
q_i = \Pr(y_k = y^* \mid x = x_{k,i}^-) \\
= \Pr(v_k = y^* - h(x_{k,i}^-)) \\
\sim \frac{1}{(2\pi)^{m/2}|R|^{1/2}} \exp \left( -\frac{(y^* - h(x_{k,i}^-))^T R^{-1} (y^* - h(x_{k,i}^-))}{2} \right).
\]  

(31)

The \( \sim \) symbol means that the probability is proportional to the right hand side. We use it to compare one relative likelihood with another because all are scaled by the same factor.

**Normalization** Scale the relative likelihoods obtained in the previous step, for example, (31), as follows

\[
q_i = \frac{q_i}{\sum_{j=1}^n q_j}
\]

After the scaling, all the likelihoods add up to one.

**Resampling** Generate \( n \) posterior particles \( x_{k,i}^+ \) based on the relative likelihoods \( q_i \).

In each iteration step, we have a set of particles \( x_{k,i}^+ \) distributed according to the pdf \( p(x_k \mid y_k) \). We can compute the mean and covariance.

**Example 3.** We consider a benchmark problem in nonlinear estimation where the system is given as follows:

\[
x_k = \frac{1}{2} x_{k-1} + \frac{25 x_{k-1}}{1 + x_{k-1}^2} + 8 \cos(1.2(k - 1)) + w_k, \\
y_k = \frac{1}{20} z_k^2 + v_k,
\]

where \( \{w_k\} \) and \( \{v_k\} \) are Gaussian white noise sequences with zero means, and unit variances.
The high degree of nonlinearity in the process and the measurement makes the problem difficult for a Kalman filter. We take the initial state $x_0 = 0.1$ and initial estimate $\hat{x}_0 = x_0$, and the initial estimation covariance for the Kalman filter as $P_0^+ = 2$. Simulate the extended Kalman filter (EKF) [1] and the particle filter over 50 time steps (using 100 particles).

The next figure compares the EKF and particle filter estimates. Not only is the EKF estimate poor, but also the EKF thinks (based on the computed covariance) that the estimate is better than it really is. In comparison, the particle filter does a better job. The estimation errors for the Kalman and particle filters are 16.3 and 2.6, respectively.

![Simulation Results](image)

References


---

6Figure 15.3 on p. 470 of [1]

7measured as the roots of the mean squares.