CHAPTER 15

The particle filter

> In view of all that we have said in the foregoing sections, the many obstacles we appear to have surmounted, what casts the pall over our victory celebration? It is the curse of dimensionality, a malediction that has plagued the scientist from earliest days.
>
> —Richard Bellman [Bel61]

> We want now to point out that modern computing machines are extremely well suited to perform the procedures described.
>
> —Nicholas Metropolis and S. Ulam [Met49]

Particle filters had their beginnings in the 1940s with the work of Metropolis, and Norbert Wiener suggested something much like particle filtering as early as 1940 [Wie56]. But only since the 1980s has computational power been adequate for their implementation. Even now it is the computational burden of the particle filter that is its primary obstacle to more widespread use. The particle filter is a statistical, brute-force approach to estimation that often works well for problems that are difficult for the conventional Kalman filter (i.e., systems that are highly nonlinear). Particle filtering goes by many other names, including sequential importance sampling [Dou01, Chapter 11], bootstrap filtering [Gor93], the condensation algorithm [Isa96, Mac99], interacting particle approximations [Mor98], Monte Carlo filtering [Kit96], sequential Monte Carlo (SMC) filtering [And04, Cri02], and sur-

Optimal State Estimation, First Edition. By Dan J. Simon
ISBN 0471708585

vival of the fittest [Kan95]. A short discussion on the origins of particle filtering can be found in [Iba01]. Reference books on the particle filter include [Dou01, Ris04].

Particle filters had their origin in Nicolas Metropolis's work in 1949 [Met49], in which he proposed studying systems by investigating the properties of sets of particles rather than the properties of individual particles. He used the analogy of the card game of solitaire. What is the probability of success in a game of solitaire? The probability may be impossible to compute analytically (because of *all* of the possible permutations of play). But if a person plays several hundred games and succeeds in a certain proportion of those games, then the probability of success can be approximated on that basis:

$$\Pr(\text{Success}) \approx \frac{\text{Number of successes}}{\text{Number of trials}} \tag{15.1}$$

This simple idea hearkens back to the definition of probability in Section 2.1. Given the recent invention of the electronic computer at the time, Metropolis's work was certainly ahead of its time. Now that fast, parallel computers are available, his work is beginning to see its fruition in the methods described in this chapter.

As discussed in Chapter 13, the extended Kalman filter (EKF) is the most widely applied state estimation algorithm for nonlinear systems. However, the EKF can be difficult to tune and often gives unreliable estimates if the system nonlinearities are severe. This is because the EKF relies on linearization to propagate the mean and covariance of the state. Chapter 14 discussed the unscented Kalman filter and showed how it reduces linearization errors. We saw that the UKF can provide significant improvements in estimation accuracy over the EKF. However, the UKF is still only an approximate nonlinear estimator. The EKF estimates the mean of a nonlinear system with first-order accuracy, and the UKF improves on this by providing an estimate with higher-order accuracy. However, this simply defers the inevitable divergence that will occur when the system or measurement nonlinearities become too severe.

This chapter presents the particle filter, which is a completely nonlinear state estimator. Of course, there is no free lunch [Ho02]. The price that must be paid for the high performance of the particle filter is an increased level of computational effort. There may be problems for which the improved performance of the particle filter is worth the increased computational effort. There may be other applications for which the improved performance is not worth the extra computational effort. These trade-offs are problem dependent and must be investigated on an individual basis.

The particle filter is a probability-based estimator. Therefore, in Section 15.1, we will discuss the Bayesian approach to state estimation, which will provide a foundation for the derivation of the particle filter. In Section 15.2, we will derive the particle filter. In Section 15.3, we will explore some implementation issues and methods for improving the performance of the particle filter.

15.1 BAYESIAN STATE ESTIMATION

In this section, we will briefly discuss the Bayesian approach to state estimation. This is based on Bayes' Rule, which is discussed in Chapter 2. This section is based on the presentation given in [Gor93], which is similar to many other books and papers on the subject of Bayesian estimation [Dou01, Ris04].

Suppose we have a nonlinear system described by the equations

$$\begin{aligned} x_{k+1} &= f_k(x_k, w_k) \\ y_k &= h_k(x_k, v_k) \end{aligned} \tag{15.2}$$

where k is the time index, x_k is the state, w_k is the process noise, y_k is the measurement, and v_k is the measurement noise. The functions $f_k(\cdot)$ and $h_k(\cdot)$ are time-varying nonlinear system and measurement equations. The noise sequences $\{w_k\}$ and $\{v_k\}$ are assumed to be independent and white with known pdf's. The goal of a Bayesian estimator is to approximate the conditional pdf of x_k based on measurements $y_1, y_2, \cdots, y_k$. This conditional pdf is denoted as

$$p(x_k|Y_k) = \text{pdf of } x_k \text{ conditioned on measurements } y_1, y_2, \cdots, y_k \tag{15.3}$$

The first measurement is obtained at $k = 1$, so the initial condition of the estimator is the pdf of x_0, which can be written as

$$p(x_0) = p(x_0|Y_0) \tag{15.4}$$

since Y_0 is defined as the set of no measurements. Once we compute $p(x_k|Y_k)$ then we can estimate x_k in whatever way we think is most appropriate, depending on the problem. The conditional pdf $p(x_k|Y_k)$ may be multimodal, in which case we may not want to use the mean of x_k as our estimate. For example, suppose that the conditional pdf is computed as shown in Figure 15.1. In this case, the mean of x is 0, but there is zero probability that x is equal to 0, so we may not want to use 0 as our estimate of x. Instead we might want to use fuzzy logic and say that $\hat{x} = \pm 2$, each with a level of membership equal to 0.5 [Lew97].

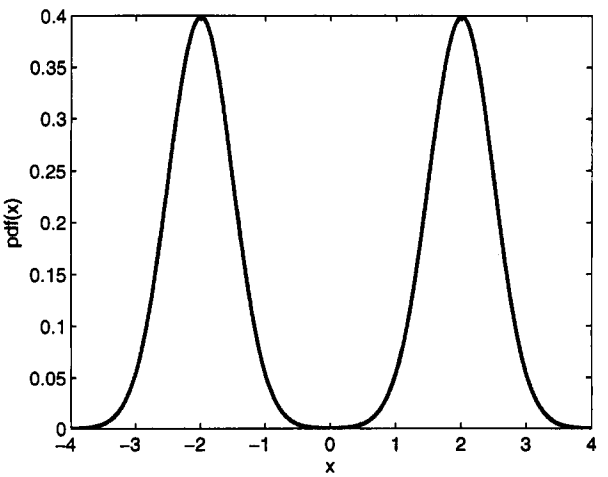


Figure 15.1 An example of a multimodal probability density function. What single number should be used as an estimate of x?

Our goal is to find a recursive way to compute the conditional pdf $p(x_k|Y_k)$. Before we find this conditional pdf, we will find the conditional pdf $p(x_k|Y_{k-1})$. This

is the pdf of x_k given all measurements *prior to* time k. We can use Equations (2.17) and (2.51) to write this pdf as

$$\begin{aligned} p(x_k|Y_{k-1}) &= \int p[(x_k, x_{k-1})|Y_{k-1}]\, dx_{k-1} \\ &= \int p[x_k|(x_{k-1}, Y_{k-1})]p(x_{k-1}|Y_{k-1})\, dx_{k-1} \end{aligned} \tag{15.5}$$

But notice from our system description in Equation (15.2) that x_k is entirely determined by x_{k-1} and w_{k-1}; therefore $p[x_k|(x_{k-1}, Y_{k-1})] = p(x_k|x_{k-1})$ and we see that

$$p(x_k|Y_{k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|Y_{k-1})\, dx_{k-1} \tag{15.6}$$

The second pdf on the right side of the above equation is not available yet, but it is available at the initial time [see Equation (15.4)]. Later in this section we will see how to compute it recursively. The first pdf on the right side of the above equation is available. The pdf $p(x_k|x_{k-1})$ is simply the pdf of the state at time k given a specific state at time $(k-1)$. We know this pdf because we know the system equation $f_k(\cdot)$ and we know the pdf of the noise w_k (see Section 2.3). For example, suppose that the system equation is given as $x_{k+1} = x_k + w_k$ and suppose that $x_{k-1} = 1$ and w_{k-1} is uniformly distributed on $[-1, 1]$. Then the pdf $p(x_k|x_{k-1})$ is uniformly distributed on $[0, 2]$.

Now consider the *a posteriori* conditional pdf of x_k. We can again use Equations (2.17) and (2.51) to write this pdf as

$$\begin{aligned} p(x_k|Y_k) &= \frac{p(Y_k|x_k)}{p(Y_k)}p(x_k) \\ &= \frac{p[(y_k, Y_{k-1})|x_k]}{p(y_k, Y_{k-1})} \underbrace{\frac{p(x_k|Y_{k-1})p(Y_{k-1})}{p(Y_{k-1}|x_k)}}_{p(x_k)} \\ &= \frac{p(x_k, y_k, Y_{k-1})}{p(x_k)p(y_k, Y_{k-1})} \frac{p(x_k, Y_{k-1})p(Y_{k-1})}{p(Y_{k-1})p(Y_{k-1}|x_k)} \end{aligned} \tag{15.7}$$

We can multiply both the numerator and denominator of this equation by $p(x_k, y_k)$ to obtain

$$p(x_k|Y_k) = \frac{p(x_k, y_k, Y_{k-1})p(x_k, Y_{k-1})p(Y_{k-1})}{p(x_k)p(y_k, Y_{k-1})p(Y_{k-1})p(Y_{k-1}|x_k)} \frac{p(x_k, y_k)}{p(x_k, y_k)} \tag{15.8}$$

Now we use the ratios of various joint pdfs to marginal pdfs in the above equation to obtain conditional pdfs. This gives

$$p(x_k|Y_k) = \frac{p[Y_{k-1}|(x_k, y_k)]p(y_k|x_k)p(x_k|Y_{k-1})}{p(y_k|Y_{k-1})p(Y_{k-1}|x_k)} \tag{15.9}$$

Note that y_k is a function of x_k, so $p[Y_{k-1}|(x_k, y_k)] = p(Y_{k-1}|x_k)$. These two terms cancel in the above equation and we obtain

$$p(x_k|Y_k) = \frac{p(y_k|x_k)p(x_k|Y_{k-1})}{p(y_k|Y_{k-1})} \tag{15.10}$$

All of the pdf's on the right side of the above equation are available. The pdf $p(y_k|x_k)$ is available from our knowledge of the measurement equation $h_k(\cdot)$ and our knowledge of the pdf of the measurement noise v_k. The pdf $p(x_k|Y_{k-1})$ is available from Equation (15.6). Finally, the pdf $p(y_k|Y_{k-1})$ is obtained (in the same way that Equation (15.5) was obtained) as follows:

$$\begin{aligned} p(y_k|Y_{k-1}) &= \int p[(y_k, x_k)|Y_{k-1}]\, dx_k \\ &= \int p[y_k|(x_k, Y_{k-1})]p(x_k|Y_{k-1})\, dx_k \end{aligned} \tag{15.11}$$

But y_k is completely determined by x_k and v_k, so $p[y_k|(x_k, Y_{k-1})] = p(y_k|x_k)$ and

$$p(y_k|Y_{k-1}) = \int p(y_k|x_k)p(x_k|Y_{k-1})\, dx_k \tag{15.12}$$

Both of the pdf's on the right side of the above equation are available as discussed above. $p(y_k|x_k)$ is available from our knowledge of the measurement equation $h(\cdot)$ and the pdf of v_k, and $p(x_k|Y_{k-1})$ is available from Equation (15.6).

Summarizing the development of this section, the recursive equations of the Bayesian state estimation filter can be summarized as follows.

The recursive Bayesian state estimator

1. The system and measurement equations are given as follows:

$$\begin{aligned} x_{k+1} &= f_k(x_k, w_k) \\ y_k &= h_k(x_k, v_k) \end{aligned} \tag{15.13}$$

   where $\{w_k\}$ and $\{v_k\}$ are independent white noise processes with known pdf's.

2. Assuming that the pdf of the initial state $p(x_0)$ is known, initialize the estimator as follows:

$$p(x_0|Y_0) = p(x_0) \tag{15.14}$$

3. For $k = 1, 2, \cdots$, perform the following.

   (a) The *a priori* pdf is obtained from Equation (15.6).

$$p(x_k|Y_{k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|Y_{k-1})\, dx_{k-1} \tag{15.15}$$

   (b) The *a posteriori* pdf is obtained from Equations (15.10) and (15.12).

$$p(x_k|Y_k) = \frac{p(y_k|x_k)p(x_k|Y_{k-1})}{\int p(y_k|x_k)p(x_k|Y_{k-1})\, dx_k} \tag{15.16}$$

Analytical solutions to these equations are available only for a few special cases. In particular, if $f(\cdot)$ and $h(\cdot)$ are linear, and x_0, $\{w_k\}$, and $\{v_k\}$ are additive, independent, and Gaussian, then the solution is the Kalman filter discussed in Chapter 5. This way of obtaining the Kalman filter is more complicated than the least squares approach that we used in Chapter 5. The Bayesian derivation of the

Kalman filter can be found in many references, including [Rho71], [Spa88, Chapter 6], [Ho64, Wes85], [Kit96a, Chapter 6]. When the Kalman filter is derived this way, then no conclusions can be drawn about the optimality of the filter when the noise is not Gaussian. In fact, other optimal (nonKalman) filters have been derived for other noise distributions [Ser81]. Nevertheless, the Bayesian derivation proves that when the noise is Gaussian, the Kalman filter is the optimal filter. However, the least squares derivation that we used in Chapter 5 shows that the Kalman filter is the optimal *linear* filter, regardless of the pdf of the noise.

15.2 PARTICLE FILTERING

In this section, we derive the basic idea of the particle filter. The particle filter was invented to numerically implement the Bayesian estimator of the previous section. The main idea is intuitive and straightforward. At the beginning of the estimation problem, we randomly generate a given number N state vectors based on the initial pdf $p(x_0)$ (which is assumed to be known). These state vectors are called particles and are denoted as $x_{0,i}^+$ $(i = 1, \cdots, N)$. At each time step $k = 1, 2, \cdots$, we propagate the particles to the next time step using the process equation $f(\cdot)$:

$$x_{k,i}^- = f_{k-1}(x_{k-1,i}^+, w_{k-1}^i) \quad (i = 1, \cdots, N) \tag{15.17}$$

where each w_{k-1}^i noise vector is randomly generated on the basis of the known pdf of w_{k-1}. After we receive the measurement at time k, we compute the conditional relative likelihood of each particle $x_{k,i}^-$. That is, we evaluate the pdf $p(y_k|x_{k,i}^-)$. As discussed in Section 15.1, this can be done if we know the nonlinear measurement equation and the pdf of the measurement noise. For example, if an m-dimensional measurement equation is given as $y_k = h(x_k) + v_k$ and $v_k \sim N(0, R)$ then a relative likelihood q_i that the measurement is equal to a specific measurement y^*, given the premise that x_k is equal to the particle $x_{k,i}^-$, can be computed as follows [compare with Equation (2.73)].

$$\begin{aligned} q_i &= P[(y_k = y^*)|(x_k = x_{k,i}^-)] \\ &= P[v_k = y^* - h(x_{k,i}^-)] \\ &\sim \frac{1}{(2\pi)^{m/2}|R|^{1/2}} \exp\left(\frac{-[y^* - h(x_{k,i}^-)]^T R^{-1}[y^* - h(x_{k,i}^-)]}{2}\right) \end{aligned} \tag{15.18}$$

The $\sim$ symbol in the above equation means that the probability is not really given by the expression on the right side, but the probability is directly proportional to the right side. So if this equation is used for all the particles $x_{k,i}^-$ $(i = 1, \cdots, N)$, then the *relative* likelihoods that the state is equal to each particle will be correct. Now we normalize the relative likelihoods obtained in Equation (15.18) as follows.

$$q_i = \frac{q_i}{\sum_{j=1}^N q_j} \tag{15.19}$$

This ensures that the sum of all the likelihoods is equal to one. Next we resample the particles from the computed likelihoods. That is, we compute a brand new set of particles $x_{k,i}^+$ that are randomly generated on the basis of the relative likelihoods q_i.

This can be done several different ways. One straightforward (but not necessarily efficient) way is the following [Ris04]. For $i = 1, \cdots, N$, perform the following two steps.

1. Generate a random number r that is uniformly distributed on $[0, 1]$.

2. Accumulate the likelihoods q_i into a sum, one at a time, until the accumulated sum is greater than r. That is, $\sum_{m=1}^{j-1} q_m < r$ but $\sum_{m=1}^{j} q_m \geq r$. The new particle $x_{k,i}^{+}$ is then set equal to the old particle $x_{k,j}^{-}$.

This resampling idea is formally justified in [Smi92], where it is shown that the ensemble pdf of the new particles $x_{k,i}^{+}$ tends to the pdf $p(x_k|y_k)$ as the number of samples N approaches ∞. The resampling step can be summarized as follows:

$$x_{k,i}^{+} = x_{k,j}^{-} \text{ with probability } q_j \quad (i, j = 1, \cdots, N) \tag{15.20}$$

This is illustrated in Figure 15.2.

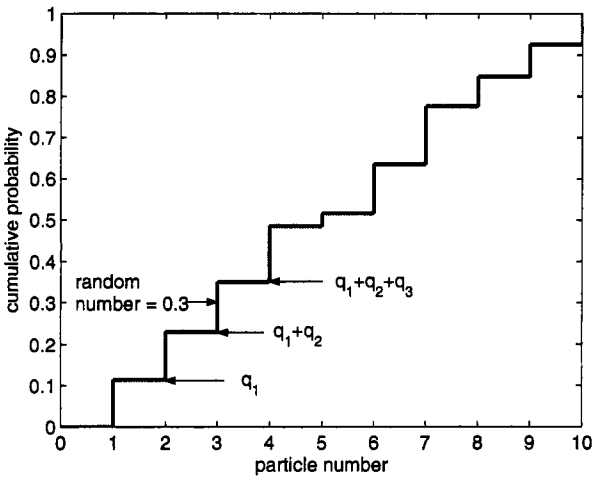


Figure 15.2 Illustration of resampling in the particle filter. For example, if a random number $r = 0.3$ is generated (from a distribution that is uniform on $[0, 1]$), the smallest value of j for which $\sum_{m=1}^{j} q_m \geq r$ is $j = 3$. Therefore the resampled particle is set equal to $x_{k,3}^{-}$.

The computational effort of the particle filter is often a bottleneck to its implementation. With this in mind, more efficient resampling methods can be implemented, such as order statistics [Car99, Rip87], stratified sampling and residual sampling [Liu98], and systematic resampling [Kit96]. Other ways of resampling have also been proposed [Mul91]. For example, the *a priori* samples $x_{k,j}^{-}$ $(j = 1, \cdots, N)$ could be accepted as *a posteriori* samples with a probability that is proportional to q_j. However, in this case additional logic must be incorporated to maintain a constant sample size N.

Now we have a set of particles $x_{k,i}^{+}$ that are distributed according to the pdf $p(x_k|y_k)$. We can compute any desired statistical measure of this pdf. For example,

if we want to compute the expected value $E(x_k|y_k)$ then we can approximate it as the algebraic mean of the particles:

$$E(x_k|y_k) \approx \frac{1}{N}\sum_{i=1}^{N} x_{k,i}^{+} \tag{15.21}$$

The particle filter can be summarized as follows.

The particle filter

1. The system and measurement equations are given as follows:

$$\begin{aligned} x_{k+1} &= f_k(x_k, w_k) \\ y_k &= h_k(x_k, v_k) \end{aligned} \tag{15.22}$$

   where $\{w_k\}$ and $\{v_k\}$ are independent white noise processes with known pdf's.

2. Assuming that the pdf of the initial state $p(x_0)$ is known, randomly generate N initial particles on the basis of the pdf $p(x_0)$. These particles are denoted $x_{0,i}^{+}$ $(i = 1, \cdots, N)$. The parameter N is chosen by the user as a trade-off between computational effort and estimation accuracy.

3. For $k = 1, 2, \cdots$, do the following.

   (a) Perform the time propagation step to obtain *a priori* particles $x_{k,i}^{-}$ using the known process equation and the known pdf of the process noise:

$$x_{k,i}^{-} = f_{k-1}(x_{k-1,i}^{+}, w_{k-1}^{i}) \quad (i = 1, \cdots, N) \tag{15.23}$$

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

   (b) 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 $p(y_k|x_{k,i}^{-})$ on the basis of the nonlinear measurement equation and the pdf of the measurement noise.

   (c) Scale the relative likelihoods obtained in the previous step as follows:

$$q_i = \frac{q_i}{\sum_{j=1}^{N} q_j} \tag{15.24}$$

   Now the sum of all the likelihoods is equal to one.

   (d) Generate a set of *a posteriori* particles $x_{k,i}^{+}$ on the basis of the relative likelihoods q_i. This is called the resampling step (for example, see Figure 15.2).

   (e) Now that we have a set of particles $x_{k,i}^{+}$ that are distributed according to the pdf $p(x_k|y_k)$, we can compute any desired statistical measure of this pdf. We typically are most interested in computing the mean and the covariance.

■ **EXAMPLE 15.1**

Suppose that we have a scalar system given by the following equations:

$$\begin{aligned} x_k &= \frac{1}{2}x_{k-1} + \frac{25x_{k-1}}{1+x_{k-1}^2} + 8\cos[1.2(k-1)] + w_k \\ y_k &= \frac{1}{20}x_k^2 + v_k \end{aligned} \tag{15.25}$$

where $\{w_k\}$ and $\{v_k\}$ are zero-mean Gaussian white noise sequences, both with variances equal to 1. This system has become a benchmark in the nonlinear estimation literature [Kit87, Gor93]. The high degree of nonlinearity in both the process and measurement equations makes this a difficult state estimation problem for a Kalman filter. We take the initial state as $x_0 = 0.1$, the initial state estimate as $\hat{x}_0 = x_0$, and the initial estimation covariance for the Kalman filter as $P_0^+ = 2$. We can simulate the EKF and the particle filter to estimate the state x. We used a simulation length of 50 time steps, and 100 particles in the particle filter. Figure 15.3 shows the EKF and particle filter estimates of the state. Not only is the EKF estimate poor, but the EKF *thinks* (on the basis of the computed covariance) that the estimate is much better than it really is. The true state is usually farther away from the estimated state than the 95% confidence measure of the EKF (as determined from the covariance P). On the other hand, Figure 15.3 shows that the particle filter does a nice job of estimating the state for this example. The RMS estimation errors for the Kalman and particle filters were 16.3 and 2.6, respectively.

Note that it might be possible to modify the Kalman filter to obtain better performance. For example, some of the procedures discussed in Section 5.5 to prevent divergence could improve the Kalman filter performance in this example. Sometimes, changing the coordinate system of the state space equation or measurement equation can improve performance [Aid83]. Nevertheless, this example shows the type of improvement that can be obtained with the use of particle filtering.

▽▽▽

15.3 IMPLEMENTATION ISSUES

In this section, we discuss a few implementation issues that often arise in the application of particle filters. The methods discussed in this section can significantly improve the performance of the particle filter, and in fact can make the difference between success and failure.

15.3.1 Sample impoverishment

Sample impoverishment occurs when the region of state space in which the pdf $p(y_k|x_k)$ has significant values does not overlap with the pdf $p(x_k|y_{k-1})$. This means that if all of our *a priori* particles are distributed according to $p(x_k|y_{k-1})$, and we then use the computed pdf $p(y_k|x_k)$ to resample the particles, only a few particles will be resampled to become *a posteriori* particles. This is because only a few of the

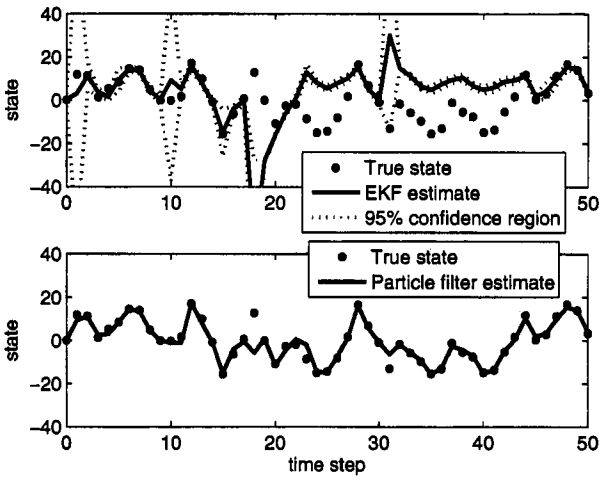


Figure 15.3 Example 15.1 results. Extended Kalman filter and particle filter estimation performance for a highly nonlinear scalar system.

a priori particles will be in a region of state space where the computed pdf $p(y_k|x_k)$ has a significant value. This means that the resampling process will select only a few distinct *a priori* particles to become *a posteriori* particles. Eventually, all of the particles will collapse to the same value.[1] This problem will be exacerbated if the measurements are not consistent with the process model (modeling errors). This can be overcome by a brute-force method of simply increasing the number of particles N, but this can quickly lead to unreasonable computational demands, and often simply delays the inevitable sample impoverishment. Other more intelligent ways of dealing with this problem can be used [Aru02, Gor93]. In the following subsections we discuss several remedies for sample impoverishment, including roughening, prior editing, regularized particle filtering, Markov chain Monte Carlo resampling, and auxiliary particle filtering.

15.3.1.1 Roughening Roughening can be used to prevent sample impoverishment, as shown in [Dou01, Chapter 14], [Gor93]. In this method, random noise is added to each particle after the resampling process. This is similar to adding artificial process noise to the Kalman filter (see Section 5.5). In the roughening approach, the *a posteriori* particles (i.e., the outputs of the resampling step) are modified as follows:

$$\begin{aligned} x_{k,i}^{+}(m) &= x_{k,i}^{+}(m) + \Delta x(m) \quad (m = 1, \cdots, n) \\ \Delta x(m) &\sim (0, KM(m)N^{-1/n}) \end{aligned} \tag{15.26}$$

$\Delta x(m)$ is a zero-mean random variable (usually Gaussian). K is a scalar tuning parameter, N is the number of particles, n is the dimension of the state space, and M is a vector containing the maximum difference between the particle elements

[1] This is called the black hole of particle filtering.

before roughening. The mth element of the M vector is given as

$$M(m) = \max_{i,j} |x^+_{k,i}(m) - x^+_{k,j}(m)| \quad (m = 1, \cdots, n) \tag{15.27}$$

where k is the time step, and i and j are particle numbers. K is a tuning parameter that specifies the amount of jitter that is added to each particle. In [Gor93] the value $K = 0.2$ is used.

■ EXAMPLE 15.2

In this example, we consider the same problem as discussed in Example 14.2. That is, we will try to estimate the altitude, velocity, and ballistic coefficient of a body as it falls toward earth. We use the extended Kalman filter, the unscented Kalman filter, and the particle filter to estimate the system state. A straightforward implementation of the particle filter does not work very well in this example. In order to get good results we had to use the roughening procedure of Equation (15.26) with a tuning parameter $K = 0.2$. We also had to constrain each particle's third element (ballistic coefficient) to a nonnegative value so that the integration of the time-update equations in the particle filter did not diverge. We used 1000 particles. Figure 15.4 shows typical EKF, UKF, and particle filter estimation error magnitudes for this system. It is seen that the particle filter provides performance on par with the UKF, but at the price of much higher computational effort. The UKF is essentially an "intelligent" particle filter with only seven particles (twice the number of states plus one), whereas the particle filter can be viewed as a "brute-force" filter with 1000 particles. Perhaps some additional modifications could be made to the particle filter to obtain better performance, but the same could be said for the UKF.

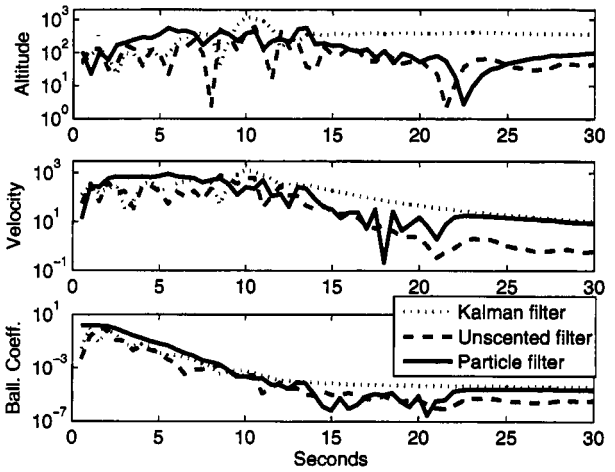


Figure 15.4 Example 15.2 results. Kalman filter, unscented filter, and particle filter estimation-error magnitudes.

▽▽▽

15.3.1.2 Prior editing If roughening does not prevent sample impoverishment, then prior editing can be tried. This involves rejection of an *a priori* sample if it is in a region of state space with small q_i. If an *a priori* sample is in a region of small probability, then it can be roughened as many times as necessary, using a procedure like Equation (15.26), until it is in a region of significant probability q_i. In [Gor93] prior editing is implemented as follows: if the magnitude of $[y_k - h(x_{k,i}^-)]$ is more than six standard deviations of the measurement noise, then it is highly unlikely to be selected as an *a posteriori* particle. In this case, $x_{k-1,i}^+$ is roughened and then passed through the system equation again to obtain a new $x_{k,i}^-$. This is repeated as many times as necessary until $x_{k,i}^-$ is in a region of nonnegligible probability.

15.3.1.3 Regularized particle filtering Another way of preventing sample impoverishment is through the use of the regularized particle filter (RPF) [Dou01, Chapter 12], [Ris04]. This performs resampling from a continuous approximation of the pdf $p(y_k|x_{k,i}^-)$ rather than from the discrete pdf samples used thus far. Recall in our resampling step in Equation (15.18) that we used the probability

$$q_i = P[(y_k = y^*)|(x_k = x_{k,i}^-)] \tag{15.28}$$

to determine the likelihood of selecting an *a priori* particle to be an *a posteriori* particle. Instead, we can use the pdf $p(x_k|y_k)$ to perform resampling. That is, the probability of selecting the particle $x_{k,i}^-$ to be an *a posteriori* particle is proportional to the pdf $p(x_k|y_k)$ evaluated at $x_k = x_{k,i}^-$. In the RPF, this pdf is approximated as

$$\hat{p}(x_k|y_k) = \sum_{i=1}^{N} w_{k,i} K_h(x_k - x_{k,i}) \tag{15.29}$$

where $w_{k,i}$ are the weights that are used in the approximation. Later on, we will see that these weights should be set equal to the q_i probabilities that were computed in Equation (15.18). $K_h(\cdot)$ is given as

$$K_h(x) = h^{-n} K(x/h) \tag{15.30}$$

where h is the positive scalar kernel bandwidth, and n is the dimension of the state vector. $K(\cdot)$ is a kernel density that is a symmetric pdf that satisfies

$$\begin{aligned} \int xK(x)\,dx &= 0 \\ \int ||x||_2^2 K(x)\,dx &< \infty \end{aligned} \tag{15.31}$$

The kernel $K(\cdot)$ and the bandwidth h are chosen to minimize a measure of the error between the assumed true density $p(x_k|y_k)$ and the approximate density $\hat{p}(x_k|y_k)$:

$$\{K(x), h\} = \text{argmin} \int \left[\hat{p}(x|y_k) - p(x|y_k)\right]^2 dx \tag{15.32}$$

In the classic case of equal weights ($w_{k,i} = 1/N$ for $i = 1, \cdots, N$) the optimal kernel is given as

$$K(x) = \begin{cases} \frac{n+2}{2v_n}(1 - ||x||_2^2) & \text{if } ||x||_2 < 1 \\ 0 & \text{otherwise} \end{cases} \tag{15.33}$$

where v_n is the volume of the n-dimensional unit hypersphere. $K(x)$ is called the Epanechnikov kernel [Dou01, Chapter 12].

An n-dimensional unit hypersphere is a volume in n dimensions in which all points are one unit from the origin [Cox73]. In one dimension, the unit hypersphere is a line with a length of two and a "volume" of two. In two dimensions, the unit hypersphere is a circle with a radius of one and volume π. In three dimensions, the unit hypersphere is a ball with a radius of one and volume $4\pi/3$. In n dimensions, the unit hypersphere has a volume $v_n = 2\pi v_{n-2}/n$.

If $p(x|y_k)$ is Gaussian with an identity covariance matrix then the optimal bandwidth is given as

$$h^* = \left[8v_n^{-1}(n+4)(2\sqrt{\pi})^n\right]^{1/(n+4)} N^{-1/(n+4)} \tag{15.34}$$

In order to handle the case of multimodal pdf's,[2] we should use $h = h^*/2$ [Dou01, Chapter 12],[Sil86]. These choices for the kernel and the bandwidth are optimal only for the case of equal weights and a Gaussian pdf, but they still are often used in other situations to obtain good particle filtering results. Instead of selecting *a priori* particles to become *a posteriori* particles using the probabilities of Equation (15.28), we instead select *a posteriori* particles based on the pdf approximation given in Equation (15.29). This allows more diversity as we perform the update from the *a priori* particles to *a posteriori* particles. In general, we should set the $w_{k,i}$ weights in Equation (15.29) equal to the q_i probabilities shown in Equation (15.28).

Since this procedure assumes that the true density $p(x_k|y_k)$ has a unity covariance matrix, we numerically compute the covariance of the $x_{k,i}^-$ at each time step. Suppose that this covariance is computed as S (an $n \times n$ matrix). Then we compute the matrix square root of S, denoted as A, such that $AA^T = S$ (e.g., we can use Cholesky decomposition for this computation). Then we compute the kernel as

$$K_h(x) = (\det A)^{-1} h^{-n} K(A^{-1}x/h) \tag{15.35}$$

The RPF resampling algorithm can be summarized as follows.

Regularized particle filter resampling

This resampling strategy replaces Step (3d) in the particle filter algorithm on page 468. We have an n-state system, the N *a priori* particles $x_{k,i}^-$, and the N corresponding (normalized) *a priori* probabilities q_i. Generate the *a posteriori* particles $x_{k,i}^+$ as follows.

1. Compute the ensemble mean μ and covariance S of the *a priori* particles as follows.

$$\begin{aligned} \mu &= \frac{1}{N}\sum_{i=1}^{N} x_{k,i}^- \\ S &= \frac{1}{N-1}\sum_{i=1}^{N}(x_{k,i}^- - \mu)(x_{k,i}^- - \mu)^T \end{aligned} \tag{15.36}$$

   Some authors use an N in the denominator of the S equation, but $(N-1)$ gives an unbiased estimate (see Problem 3.6).

[2] A multimodal pdf is one with more than one local maxima. See, for example, Figure 15.1.

2. Perform a square root factorization of S (e.g., a Cholesky factorization) to compute the $n \times n$ matrix A such that $AA^T = S$.

3. Compute the volume of the n-dimensional unit sphere as $v_n = 2\pi v_{n-2}/n$. The starting values for this recursion are $v_1 = 2$, $v_2 = \pi$, and $v_3 = 4\pi/3$.

4. Compute the optimal kernel bandwidth h as follows:

$$h = \frac{1}{2}\left[8v_n^{-1}(n+4)(2\sqrt{\pi})^n\right]^{1/(n+4)} N^{-1/(n+4)} \tag{15.37}$$

   The bandwidth h can be considered a tuning parameter for the particle filter.

5. Approximate the pdf $p(x_k|y_k)$ as follows:

$$\hat{p}(x_k|y_k) = \sum_{i=1}^{N} q_i K_h(x_k - x_{k,i}) \tag{15.38}$$

   where the kernel $K_h(x)$ is given as

$$K_h(x) = (\det A)^{-1} h^{-n} K(A^{-1}x/h) \tag{15.39}$$

   and the Epanechnikov kernel $K(x)$ is given as

$$K(x) = \begin{cases} \frac{n+2}{2v_n}(1 - ||x||_2^2) & \text{if } ||x||_2 < 1 \\ 0 & \text{otherwise} \end{cases} \tag{15.40}$$

   Note that other kernels can also be used in the pdf approximation (see Problem 15.14). Equation (15.38) must be implemented digitally, so the user must choose a certain number of digital values at which to evaluate Equation (15.38). As with the number of particles N, the number of digital values is a trade-off between computational resources and estimation accuracy.

6. Now that we have $\hat{p}(x_k|y_k)$ from the previous step, we generate the *a posteriori* particles $x_{k,i}^+$ by probabilistically selecting points from the pdf approximation $\hat{p}(x_k|y_k)$.

EXAMPLE 15.3

Consider the same system as in Example 15.1, except use a process-noise covariance of 0.001 and only three particles ($N = 3$). In this case, the particles in the standard particle filter can quickly degenerate into a single point, but the use of an RPF can prevent this degeneration, increase diversity among the particles, and provide a better state estimate. Twenty Monte Carlo runs of this system result in average RMS errors of 4.6 for the standard particle filter and 3.0 for the RPF. Figure 15.5 shows the improvement that is possible with the use of an RPF.

Figure 15.6 shows the difference between the resampling step of the standard particle filter and the RPF. The standard particle filter has an *a priori* pdf approximation that consists of the sum of impulse functions. Therefore, the *a posteriori* particles are all set equal to one of the *a priori* particles.

However, the RPF has a pdf approximation that is a continuous function of the state estimate. Therefore, the *a posteriori* particles can be equal to any value on the horizontal axis. Of course, when we implement the RPF we have to discretize the horizontal axis in order to choose the *a posteriori* particles, but we can use as fine a discretization as our computational resources will allow.

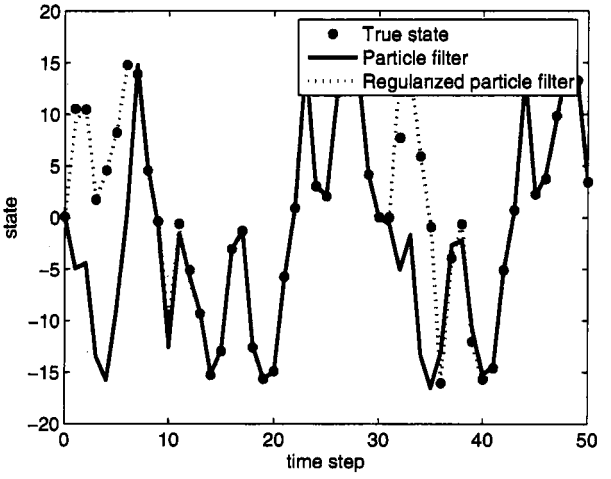


Figure 15.5 Particle filter estimation performance for the highly nonlinear scalar system of Example 15.3. This shows the improvement that is possible with the use of an RPF.

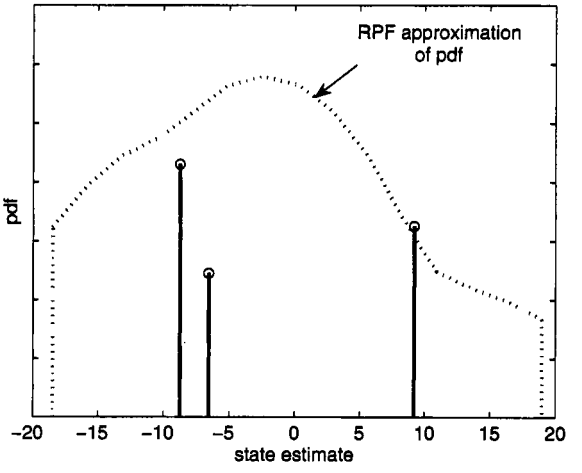


Figure 15.6 This shows the discrete pdf approximation of the standard particle filter (with three particles), and the continuous pdf approximation of the RPF. This plot is a snapshot of the pdf approximations at one time instant.

$\nabla\nabla\nabla$

15.3.1.4 Markov chain Monte Carlo resampling Another approach for preventing sample impoverishment is the Markov chain Monte Carlo (MCMC) move step [Gil01, Ris04]. This approach moves the *a priori* particle $x_{k,i}^-$ to a new randomly generated state $\tilde{x}_{k,i}^-$ if a uniformly distributed random number is less than an acceptance probability. The acceptance probability is computed as the probability that the *a priori* sample is consistent with the measurement, relative to the probability that the resampled state is consistent with the measurement. The Metropolis–Hastings acceptance probability [Rob99] is given as

$$\alpha = \min\left[1, \frac{p(y_k|\tilde{x}_{k,i}^-)}{p(y_k|x_{k,i}^-)} \frac{p(\tilde{x}_{k,i}^-|x_{k-1,i}^+)}{p(x_{k,i}^-|x_{k-1,i}^+)}\right] \tag{15.41}$$

The first fraction in the above equation is the ratio of the measurement probability conditioned on the new particle to the measurement probability conditioned on the old particle. The second fraction is the ratio of the probability of the new particle to the probability of the old particle, both conditioned on the particle at the previous time. The acceptance probability is the product of these two fractions, which increases as the probability of the new particle increases. The old *a priori* particle $x_{k,i}^-$ is therefore changed to a new particle $\tilde{x}_{k,i}^-$ if the old particle has a low probability of being selected with the resampling step. This helps to maintain diversity in the particles that come out of the resampling step.

15.3.1.5 Auxiliary particle filtering Another approach to evening out the probability of the *a priori* particles (and thus increasing diversity in the *a posteriori* particles) is called the auxiliary particle filter [Pit99, Ris04]. This approach was developed by augmenting each *a priori* particle by one element (an auxiliary variable). This increases the dimension of the problem and thus adds a degree of freedom to the choice of the resampling weights in Equation (15.19), which allows the resampling weights to be more evenly distributed. Recall from Section 15.2 that the resampling step of the standard particle filter is performed by selecting particles based on their probabilities. These probabilities are given by

$$q_i = P[(y_k = y^*)|(x_k = x_{k,i}^-)] \tag{15.42}$$

where y^* is the actual measurement at time k. The problem with this is that outliers in the batch of *a priori* particles are ignored due to their low probabilities, and the particles can therefore collapse into a single point. Auxiliary particle filtering addresses this issue by changing the resampling probability to the following:

$$q_i = \frac{P[(y_k = y^*)|(x_k = x_{k,i}^-)]}{P[(y_k = y^*)|\mu_{k,i}]} \tag{15.43}$$

where $\mu_{k,i}$ is some statistical characterization of x_k based on $x_{k,i}^-$. For example, we could use $\mu_{k,i} = E(x_k|x_{k,i}^-)$, or $\mu_{k,i} = \text{pdf}(x_k|x_{k,i}^-)$. So compared to the standard particle filter, the resampling probability of the auxiliary particle filter is smaller by a factor of $P[(y_k = y^*)|\mu_{k,i}]$. If the actual measurement is highly likely given $\mu_{k,i}$, then the actual measurement is highly likely given $x_{k,i}^-$. The auxiliary particle filter will then tend to decrease q_i relative to the standard particle filter. Likewise, the auxiliary particle filter will tend to increase q_i for highly unlikely particles. This tends to promote diversity in the population of particles.

Another easy way to smooth out the q_i probabilities is to use something like the following formula.

$$\tilde{q}_i = \frac{(\alpha - 1)q_i + \bar{q}}{\alpha} \tag{15.44}$$

where $\bar{q}$ is the sample mean of all of the q_i probabilities. The parameter $\alpha \in [1, \infty]$ controls how much regularization occurs. If $\alpha \to \infty$ then the regularized probabilities $\tilde{q}_i$ are equal to the standard probabilities q_i. If $\alpha = 1$ then all of the regularized probabilities $\tilde{q}_i$ are equal.

If the dynamics of the state-space system are linear, then there should not be any reason to use auxiliary particle filtering. The existence of outliers in the particles results from nonlinearities. This implies that the use of auxiliary particle filtering is more appropriate when the system nonlinearities are severe. In fact, if the nonlinearities are mild or nonexistent, then the use of auxiliary particle filtering could corrupt the probabilities q_i in an inappropriate way and degrade performance relative to the standard particle filter.

■ **EXAMPLE 15.4**

Consider the same system as in Examples 14.2 and 15.2. That is, we will try to estimate the altitude x_1, velocity x_2, and constant ballistic coefficient x_3 of a body as it falls toward earth. The equations for this system are given in Example 14.2. We use fourth-order Runge–Kutta integration with a step size of 0.5 sec to simulate the system for 30 seconds. We estimate the system states with the standard particle filter and the auxiliary particle filter. As mentioned in Example 15.2, a straightforward implementation of the particle filter does not work very well in this example. In Example 15.2, we used the roughening procedure of Equation (15.26). In this example, we use the auxiliary particle filter of Equation (15.44) with 200 particles. In the standard particle filter, the particles quickly collapse to a single point in state space. In the auxiliary particle filter with $\alpha = 1.1$ (obtained by manual tuning) the diversity of the particles is preserved. Averaged over 10 simulations, the use of the auxiliary particle filter improves altitude estimation by 73%, and improves velocity estimation by 55%. However, the auxiliary particle filter makes the estimate of the ballistic coefficient worse. This may be because the ballistic coefficient is not involved in any nonlinear dynamics in either the system equation or the measurement equation.

$\nabla\nabla\nabla$

15.3.2 Particle filtering combined with other filters

One approach that has been proposed for improving particle filtering is to combine it with another filter such as the EKF or the UKF [Wan01, Ris04]. In this approach, each particle is updated at the measurement time using the EKF or the UKF, and then resampling is performed using the measurement. This is like running a bank of N Kalman filters (one for each particle) and then adding a resampling step after each measurement. After $\bar{x}_{k,i}^-$ is obtained as shown in Equation (15.17), it can be refined using the EKF or UKF measurement-update equations. For example, if we want to combine the particle filter with the EKF, then after the measurement is

obtained at time k, $x_{k,i}^-$ is updated to $x_{k,i}^+$ according to the EKF equations shown in Section 13.2:

$$\begin{aligned} P_{k,i}^- &= F_{k-1,i}P_{k-1,i}^+F_{k-1,i}^T + Q_{k-1} \\ K_{k,i} &= P_{k,i}^-H_{k,i}^T(H_{k,i}P_{k,i}^-H_{k,i}^T + R_k)^{-1} \\ x_{k,i}^+ &= x_{k,i}^- + K_{k,i}\left[y_k - h(x_{k,i}^-)\right] \\ P_{k,i}^+ &= (I - K_{k,i}H_{k,i})P_{k,i}^- \end{aligned} \tag{15.45}$$

$K_{k,i}$ is the Kalman gain for the ith particle, and $P_{k,i}^-$ is the *a priori* estimation-error covariance for the ith particle. The partial derivative matrices F and H are defined as

$$\begin{aligned} F_{k-1,i} &= \left.\frac{\partial f}{\partial x}\right|_{x=x_{k-1,i}^+} \\ H_{k,i} &= \left.\frac{\partial h}{\partial x}\right|_{x=x_{k,i}^-} \end{aligned} \tag{15.46}$$

Next, resampling is performed as discussed in Section 15.2 to modify the $x_{k,i}^+$ particles (and their associated covariances $P_{k,i}^+$). This is another way to prevent sample impoverishment because the *a priori* particles $x_{k,i}^-$ are updated on the basis of the measurement at time k before they are resampled. The measurement updates of the particles could be performed with any type of filter – an EKF, a UKF, an H_∞ filter, another particle filter, and so on. The extended Kalman particle filter can be summarized as follows.

The extended Kalman particle filter

1. The system and measurement equations are given as follows:

$$\begin{aligned} x_{k+1} &= f_k(x_k, w_k) \\ y_k &= h_k(x_k, v_k) \end{aligned} \tag{15.47}$$

   where $\{w_k\}$ and $\{v_k\}$ are independent white noise processes with known pdf's.

2. Assuming that the pdf of the initial state $p(x_0)$ is known, randomly generate N initial particles on the basis of the pdf $p(x_0)$. These particles are denoted $x_{0,i}^+$ and their covariances are denoted $P_{0,i}^+ = P_0^+$ $(i = 1, \cdots, N)$. The parameter N is chosen by the user as a trade-off between computational effort and estimation accuracy.

3. For $k = 1, 2, \cdots$, do the following.

   (a) Perform the time propagation step to obtain *a priori* particles $x_{k,i}^-$ and covariances $P_{k,i}^-$ using the known process equation and the known pdf of the process noise:

$$\begin{aligned} x_{k,i}^- &= f_{k-1}(x_{k-1,i}^+, w_{k-1}^i) \\ P_{k,i}^- &= F_{k-1,i}P_{k-1,i}^+F_{k-1,i}^T + Q_{k-1} \\ F_{k-1,i} &= \left.\frac{\partial f}{\partial x}\right|_{x=x_{k-1,i}^+} \end{aligned} \tag{15.48}$$

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

(b) Update the *a priori* particles and covariances to obtain *a posteriori* particles and covariances:

$$\begin{aligned} H_{k,i} &= \left.\frac{\partial h}{\partial x}\right|_{x=x_{k,i}^-} \\ K_{k,i} &= P_{k,i}^- H_{k,i}^T (H_{k,i} P_{k,i}^- H_{k,i}^T + R_k)^{-1} \\ x_{k,i}^+ &= x_{k,i}^- + K_{k,i}\left[y_k - h(x_{k,i}^-)\right] \\ P_{k,i}^+ &= (I - K_{k,i} H_{k,i}) P_{k,i}^- \end{aligned} \tag{15.49}$$

(c) 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 $p(y_k|x_{k,i}^+)$ on the basis of the nonlinear measurement equation and the pdf of the measurement noise.

(d) Scale the relative likelihoods obtained in the previous step as follows:

$$q_i = \frac{q_i}{\sum_{j=1}^N q_j} \tag{15.50}$$

Now the sum of all the likelihoods is equal to one.

(e) Refine the set of *a posteriori* particles $x_{k,i}^+$ and covariances $P_{k,i}^+$ on the basis of the relative likelihoods q_i. This is the resampling step.

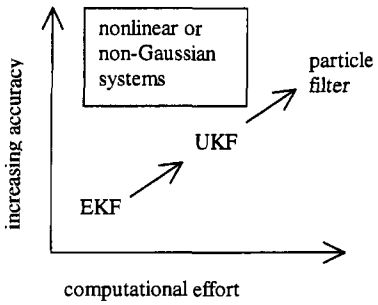
(f) Now we have a set of *a posteriori* particles $x_{k,i}^+$ and covariances $P_{k,i}^+$. We can compute any desired statistical measure of this set of particles. We typically are most interested in computing the mean and the covariance.

EXAMPLE 15.5

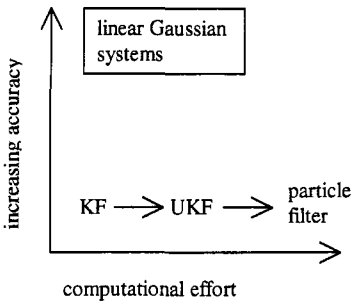
Consider the same system as in Example 15.4. That is, we will try to estimate the altitude x_1, velocity x_2, and constant ballistic coefficient x_3 of a body as it falls toward earth. The equations for this system are given in Example 14.2. We use fourth-order Runge–Kutta integration with a step size of 0.5 sec to simulate the system for 30 s. We use the standard particle filter and the EKF particle filter to estimate the states. The EKF particle filter updates the *a priori* particles at each time based on the measurement, and then the resampling step is performed as usual. In this example, we use 200 particles for the estimator. As mentioned in Example 15.4, in the standard particle filter the particles quickly collapse to a single trajectory. In Example 15.2, we used roughening to improve the particle filter. In Example 15.3, we used the regularized particle filter to improve performance. In Example 15.4, we used the auxiliary particle filter to improve performance. Here we use an EKF particle filter to improve performance. Averaged over 10 simulations, the use of the EKF particle filter improves altitude estimation accuracy by an astounding 99.6%, almost three orders of magnitude. The velocity estimation is only marginally improved, and the ballistic coefficient estimation is marginally degraded.

15.4 SUMMARY

In this chapter, we laid the foundation of Bayesian state estimation, and from there we developed the particle filter. In a linear system with Gaussian noise, the Kalman filter is optimal. In a system that is nonlinear, the Kalman filter can be used for state estimation, but the particle filter may give better results at the price of additional computational effort. In a system that has non-Gaussian noise, the Kalman filter is the optimal linear filter, but again the particle filter may perform better. The unscented Kalman filter provides a balance between the low computational effort of the Kalman filter and the high performance of the particle filter. This is depicted in Figure 15.7.



(a) The above figure depicts the increasing computational effort and increasing accuracy that is obtained by going from an EKF to a UKF to a particle filter. This applies to systems that are nonlinear or non-Gaussian.



(b) The above figure depicts the fact that the Kalman filter is optimal for linear Gaussian systems. Going from a Kalman filter to a UKF to a particle filter will increase computational effort but will not improve estimation accuracy.

Figure 15.7 State estimation trade-offs.

The particle filter has some similarities with the UKF (see Chapter 14) in that it transforms a set of points via known nonlinear equations and combines the results to estimate the mean and covariance of the state. However, in the particle filter the points are chosen randomly, whereas in the UKF the points are chosen on the basis of a specific algorithm. Because of this, the number of points used in a particle filter generally needs to be much greater than the number of points in a UKF. Another difference between the two filters is that the estimation error in a UKF does not converge to zero in any sense, but the estimation error in a particle filter does converge to zero as the number of particles (and hence the computational effort) approaches infinity.

Particle filters have found application in a wide variety of areas, including tracking problems [Ris04], demodulation of communication signals [Dou01, Chapter 4], estimation of ecological parameters and populations [Dou01, Chapter 5], image processing [Dou01, Chapter 16], neural network training [Dou01, Chapter 17], fault detection [deF02], speech recognition [Ver02], and pattern recognition [Dou01, Chapter 26]. Particle filtering is a growing area of research with many unexplored avenues and applications. Some of the more important areas of open research include the avoidance of sample impoverishment, methods for determining how many particles are required for a given problem, convergence results [Cri02], application to control and parameter estimation [Mor03, And04], connections with genetic algo-

rithms [Dou01, Chapter 20], real-time implementation issues [Kwo04], and hardware implementations of parallel particle filters (e.g., in field programmable gate arrays).

PROBLEMS

Written exercises

15.1 Consider the scalar system

$$\begin{aligned} x_{k+1} &= x_k + w_k, \quad w_k \sim U(-1,1) \\ y_k &= x_k + v_k, \quad v_k \sim U(-1,1) \end{aligned}$$

where $x_0 \sim U(-1,1)$. Suppose that the first measurement $y_1 = 1$.

a) Use the recursive Bayesian state estimator to find $\text{pdf}(x_1|Y_0)$ and $\text{pdf}(x_1|Y_1)$.

b) What is the Kalman filter estimate $\hat{x}_1^+$? How is $\hat{x}_1^+$ related to $\text{pdf}(x_1|Y_1)$?

15.2 Suppose the pdf of an RV x is given as

$$\text{pdf}(x) = \begin{cases} 1 - x/2 & x \in [0,2] \\ 0 & \text{otherwise} \end{cases}$$

The value of x can be estimated several ways.

a) The maximum-likelihood estimate is written as $\hat{x} = \text{argmax}_x \text{pdf}(x)$. Find the maximum-likelihood estimate of x.

b) The min-max estimate of x is that value of $\hat{x}$ that minimizes the magnitude of the maximum estimation error. Find the min-max estimate of x.

c) The minimum mean square estimate of x is that value of $\hat{x}$ that minimizes $E[(x - \hat{x})^2]$. Find the minimum mean square estimate of x.

d) The expected value estimate of x is given as $\hat{x} = E(x)$. Find $E(x)$.

15.3 Suppose you have a measurement $y_k = x_k^2 + v_k$, where v_k has a triangular pdf that is given as

$$\text{pdf}(v_k) = \begin{cases} 1/2 + v_k/4 & v_k \in [-2,0] \\ 1/2 - v_k/4 & v_k \in [0,2] \\ 0 & \text{otherwise} \end{cases}$$

Suppose that five *a priori* particles $x_{k,i}^-$ are given as -2, -1, 0, 1, and 2, and that the measurement is obtained as $y_k = 1$. What are the normalized likelihoods q_i of each *a priori* particle $x_{k,i}^-$?

15.4 Suppose you have a measurement $y_k = v_k/x_k$, where $v_k \sim N(9,1)$. Suppose that five *a priori* particles $x_{k,i}^-$ are given as 0.8, 0.9, 1.0, 1.1, and 1.2, and that the measurement is obtained as $y_k = 10$. What are the relative likelihoods q_i of each *a priori* particle $x_{k,i}^-$?

15.5 Suppose that five *a priori* particles are found to have probabilities 0.1, 0.1, 0.1, 0.2, and 0.5. The particles are resampled with the basic strategy depicted in Equation (15.20).

a) What is the probability that the first particle will be chosen as an *a posteriori* particle at least once?
b) What is the probability that the fifth particle will be chosen as an *a posteriori* particle at least once?
c) What is the probability that the five *a posteriori* particles will be equal to the five *a priori* particles (disregarding order)?

15.6 Suppose you have the five particles $x_{k,i}^+ = \{\ 1,\ \ 2,\ \ 3,\ \ -2,\ \ 6\ \}$. What would you propose to use for the estimate of x_k? What would you estimate as the variance of $\hat{x}_k$?

15.7 Suppose that you have five particles -1, -1, 0, 1, and 1. You want to use the roughening procedure of Section 15.3.1.1 to add a uniform random variable with a variance of $KMN^{-1/n}$ to each particle. What range of K will give a probability of at least $1/8$ that at least one of the roughened particles is less than -2?

15.8 Suppose you have the system equation $x_{k+1} = x_k$ and the measurement equation $y_k = x_k^2 + v_k$, where v_k has a triangular pdf that is given as

$$\text{pdf}(v_k) = \begin{cases} 1/2 + v_k/4 & v_k \in [-2, 0] \\ 1/2 - v_k/4 & v_k \in [0, 2] \\ 0 & \text{otherwise} \end{cases}$$

Suppose that five *a posteriori* particles $x_{k-1,i}^+$ are given as -2, -1, 0, 1, and 2, and that the measurement is obtained as $y_k = 1$. You want to use prior editing to ensure that the -2 particle has at least a 10% chance (after one roughening step) of being selected as an *a posteriori* particle at the next time step. What value of K should you use in your roughening step?

15.9 Suppose you have two particles -1 and $+1$, both with *a priori* probabilities $1/2$. Use the kernel bandwidth $h = 1$ with the regularized particle filter to find the pdf approximations $\hat{p}(x_k = -2|y_k)$, $\hat{p}(x_k = -1|y_k)$, $\hat{p}(x_k = 0|y_k)$, $\hat{p}(x_k = 1|y_k)$, and $\hat{p}(x_k = 2|y_k)$. For what values of x_k is the pdf approximation $\hat{p}(x_k|y_k)$ equal to zero?

15.10 Suppose you have N resampling probabilities q_i with sample mean μ and sample variance S. What is the sample mean and variance of the auxiliary probabilities given by Equation (15.44)?

Computer exercises

15.11 Plot the volume of the n-dimensional unit hypersphere as a function of n for $n \in [1, 20]$.

15.12 Consider two particles $x_1 = 1$ and $x_2 = 2$, with equal probabilities. Generate the approximate pdf using the Epanechnikov kernel with bandwidth $h = h^*$. Generate two separate plots (on the same figure) of the two individual terms in the summation of Equation (15.29), and also generate a plot (on the same figure) of their sum. Repeat for three particles $x_1 = 1$, $x_2 = 2$, and $x_3 = 3$ with equal probabilities.

15.13 Repeat Problem 15.12 with $h = h^*/2$ and with $h = 2h^*$. This shows that the bandwidth selection can have a strong effect on the pdf approximation.

15.14 Kernels other than the Epanechnikov kernel can also be used for pdf approximation [Sim98, Dev01]. Some of the more popular kernels can be described in one dimension as follows.

$$\text{Epanechnikov: } K(x) = \begin{cases} \frac{3}{4}(1-x^2) & |x| < 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Gaussian: } K(x) = (2\pi)^{-1/2}\exp(-x^2/2)$$

$$\text{Uniform: } K(x) = \begin{cases} \frac{1}{2} & |x| < 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Triangular: } K(x) = \begin{cases} 1-x^2 & |x| < 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Biweight: } K(x) = \begin{cases} \frac{15}{16}(1-x^2)^2 & |x| < 0 \\ 0 & \text{otherwise} \end{cases}$$

Bandwidth selection is another matter, but for this problem you can simply use the optimal Epanechnikov bandwidth for all of the kernels.

a) Repeat Problem 15.12 using Gaussian kernels.
b) Repeat Problem 15.12 using uniform kernels.
c) Repeat Problem 15.12 using triangular kernels.
d) Repeat Problem 15.12 using biweight kernels.

15.15 In this problem, we will explore the performance of the EKF and the particle filter for the system described in Example 15.1.

a) Run 100 simulations of the EKF and the particle filter with $N = 10$, $N = 100$, and $N = 1000$. What is the average RMS state-estimation error for each case?
b) Run 100 simulations of the EKF and the particle filter with $N = 100$ using $Q = 0.1$, $Q = 1$, and $Q = 10$. What is the average RMS state-estimation error for each case?