Chapter 11

Information Filtering/SRIF

The *a posteriori* state estimate error covariance matrix $P(k+1)$ can be defined in terms of the following (alternative) update formula:

$$P(k+1) = \left[\bar{P}^{-1}(k+1) + \underbrace{H^{\mathrm{T}}(k+1)R^{-1}(k+1)H(k+1)}_{\text{matrix squaring operation}} \right]^{-1}$$

The problem with this traditional approach to updating the error covariance matrix is that the matrix squaring operation is a bad idea numerically—it squares the condition number of the H matrices[1]. Note that $\bar{P}(k+1)$ may also be ill-conditioned for the same reason; so may $R(k+1)$.

Let's write $P(k+1)$ in terms of another update formula:

$$P(k+1) = \bar{P}(k+1) - W(k+1)S(k+1)W^{\mathrm{T}}(k+1)$$

Here, infinite numerical precision ensures $P(k+1) > 0$ but roundoff errors (e.g., from non-infinite numerical precision) can make $P(k+1)$ indefinite (not positive definite) or asymmetric.

One can use Joseph's form of $P(k+1)$ update to ensure $P(k+1) = P^{\mathrm{T}}(k+1) > 0$, but this doesn't help the accuracy of $P(k+1)$. A deterioration in accuracy of $P(k+1)$ can lead to garbage results.

To address these problems, Bar Shalom introduces the square root covariance filter, which keeps track of the square root of the covariance matrix. But this requires updating a Cholesky factorization. As it turns out, it's easier to work with the square root of the information matrix (the inverse of the covariance matrix). The resulting filter, the square-root information filter (SRIF), is more efficient than the square-root covariance filter when $n_z > n_x, n_v$; otherwise the square-root covariance filter is faster.

As a prelude the the SRIF, we'll first look at the traditional (non-square-root) information filter.

[1] Note that the notion of condition number applies to non-square matrices such as $H(k+1)$—the condition number is just the ratio of the largest to the smallest singular value.

11.1 Information Filter

Strategy: Instead of keeping track of $\hat{x}(k+1)$, $P(k+1)$, $\bar{x}(k+1)$, $\bar{P}(k+1)$, we keep track of

$$\begin{aligned}
\bar{y}(k) &= \bar{P}^{-1}(k)\bar{x}(k) \\
\hat{y}(k) &= P^{-1}(k)\hat{x}(k) \\
\bar{\mathscr{I}}(k) &= \bar{P}^{-1}(k) \\
\mathscr{I}(k) &= P^{-1}(k)
\end{aligned}$$

where $\mathscr{I}(k)$ is known as the information matrix and is equal to the inverse of the covariance matrix $P(k)$; $\mathscr{I}(k)$ is analagous to the Fisher information matrix—the bigger it is, the more certain the associated estimates.

We can substitute these definitions into the Kalman filter to get the information filter. Let

$$A(k) = F^{-\mathrm{T}}(k)\mathscr{I}(k)F^{-1}(k)$$

After much algebra, including the matrix inversion lemma, we arrive at the following propagation and update equations.

Propagation

$$\bar{y}(k+1) = \left\{I - A(k)\Gamma(k)\left[\Gamma^T(k)A(k)\Gamma(k) + Q^{-1}(k)\right]^{-1}\Gamma^T(k)\right\}\left[F^{-\mathrm{T}}(k)\hat{y}(k) + A(k)G(k)u(k)\right]$$

$$\bar{\mathscr{I}}(k+1) = A(k) \underbrace{-A(k)\Gamma(k)\left[\Gamma^T(k)A(k)\Gamma(k) + Q^{-1}(k)\right]^{-1}\Gamma^T(k)A(k)}_{\text{Information is decreasing due to process noise}}$$

Process noise decreases the information during the propagation step. This is similar to a hole in a metaphorical information bucket. If $Q(k) = 0$, i.e., there is no process noise, then there is no information leaking out and

$$\bar{\mathscr{I}}(k+1) = A(k) = F^{-\mathrm{T}}(k)\mathscr{I}(k)F^{-1}(k)$$

Update

$$\hat{y}(k+1) = \bar{y}(k+1) + H^{\mathrm{T}}(k+1)R^{-1}(k+1)z(k+1)$$

$$\mathscr{I}(k+1) = \bar{\mathscr{I}}(k+1) + \underbrace{H^{\mathrm{T}}(k+1)R^{-1}(k+1)H(k+1)}_{\text{measurements add information}}$$

Here the information is increasing due to new measurements. This is similar to an information pipe filling the metaphorical information bucket.

We can recover $\hat{x}$ and P by

$$\begin{aligned}
\hat{x}(k+1) &= \mathscr{I}^{-1}(k+1)\hat{y}(k+1) \\
P(k+1) &= \mathscr{I}^{-1}(k+1)
\end{aligned}$$

Remarks.

1. The information filter is more efficient than Kalman Filter if $n_z > n_x, n_v$ and if $R(k)$ is diagonal, but usually this is not the case.

2. For linear systems we can pick the initial state estimate arbitrarily so long as we set $\mathscr{I}(0) = 0$. This represents the diffuse prior, i.e., it indicates we no idea of our initial state. A diffuse initial prior cannot be as easily done with the regular Kalman Filter. We could set the error covariance matrix to infinity, but finite numerical precision limits our ability to do so in real systems. Note that we cannot compute $\hat{x}(k) = \mathscr{I}^{-1}(k)\hat{y}(k)$ until $\mathscr{I}(k)$ becomes invertible. If the system is observable, then $\mathscr{I}(k)$ will eventually become invertible. Waiting for $\mathscr{I}(k)$ to become invertible is like waiting large enough k so that H^k has full column rank in the batch initialization problem.

3. This form of the information filter still involves squaring in the $H^T R^{-1} H$ terms.

11.2 Square Root Information Filter

The square root information filter (SRIF) is a particular implementation of the information filter that involves no squaring of terms.

11.2.1 Setup

Define

$$\begin{aligned}
R_{xx}^T(k)R_{xx}(k) &= \mathscr{I}(k) \\
\bar{R}_{xx}^T(k)\bar{R}_{xx}(k) &= \bar{\mathscr{I}}(k) \\
z_x(k) &= R_{xx}(k)\hat{x}(k) \\
\bar{z}_x(k) &= \bar{R}_{xx}(k)\bar{x}(k)
\end{aligned}$$

where $R_{xx}(k)$ and $\bar{R}_{kk}(k)$ are the Cholesky factorizations of $\mathscr{I}(k)$ and $\bar{\mathscr{I}}(k)$, respectively. Also let

$$\begin{aligned}
R_a^T(k)R_a(k) &= R(k) \\
H_a(k) &= R_a^{-T}(k)H(k) \\
z_a(k) &= R_a^{-T}(k)z(k) \\
w_a(k) &= R_a^{-T}(k)w(k)
\end{aligned}$$

so that the transformed measurement equation becomes

$$z_a(k) = H_a(k)x(k) + w_a(k)$$

where

$$\begin{aligned}
E\left[w_a(k)\right] &= 0 \\
E\left[w_a(k)w_a^T(j)\right] &= I\delta_{kj}
\end{aligned}$$

The dynamics model remains unchanged:

$$x(k+1) = F(k)x(k) + G(k)u(k) + \Gamma(k)v(k)$$

$$\hat{v}(k) \triangleq E\left[v(k)|Z^k\right] = 0$$

$$E\left[v(k)v^T(j)\right] = \delta_{kj}Q(k)$$

Cholesky factorize the $Q(k)$ matrix:

$$R_{vv}^T(k)R_{vv}(k) = Q^{-1}(k)$$

Note that this factorization-and-inverse operation is most accurate numerically when we perform the Cholesky factorization first, followed by the inverse. In Matlab:

$$R_{vv}(k) = [\texttt{inv}(\texttt{chol}(Q(k)))]^{\mathrm{T}}$$

With the previous definitions, we can write out the following so-called square-root information (SRI) equations (also known as data equations):

$$z_x(k) = R_{xx}(k)x(k) + w_x(k), \qquad w_x(k) \sim (0, I) \tag{11.1}$$

$$z_v(k) = 0 = R_{vv}(k)v(k) + w_v(k), \qquad w_v(k) \sim (0, I) \tag{11.2}$$

$$\mathbb{E}\left[w_x(k)w_v^T(k)\right] = 0 \tag{11.3}$$

One can think of these SRI equations as storing, or "encoding," the state and process noise estimates and their covariances. We can recover our estimates $\hat{x}(k)$ and $\hat{v}(k)$, and their corresponding covariance matrices $P(k)$ and $Q(k)$, from the SRI equations so long as R_{xx} is invertible.[2] The state estimate $\hat{x}(k)$ can be recovered from (11.1) as follows:

$$\begin{aligned}
x(k) &= R_{xx}^{-1}(k)\left[z_x(k) - w_x(k)\right] \\
\hat{x}(k) &= E\left[x(k)|Z^k\right] \\
&= R_{xx}^{-1}(k)\underbrace{E\left[z_x(k)|Z^k\right]}_{z_x(k)} - R_{xx}^{-1}(k)\underbrace{E\left[w_x(k)|Z^k\right]}_{0} \\
&= R_{xx}^{-1}(k)z_x(k)
\end{aligned}$$

$P(k)$ can be recovered from the same SRI equation by

$$\begin{aligned}
\tilde{x}(k) &= x(k) - \hat{x}(k) = -R_{xx}^{-1}(k)w_x(k) \\
P(k) &= E\left[\tilde{x}(k)\tilde{x}^T(k)|Z^k\right] \\
&= R_{xx}^{-1}(k)\underbrace{E\left[w_x(k)w_x^T(k)\right]}_{I}R_{xx}^{-T}(k) \\
&= R_{xx}^{-1}(k)R_{xx}^{-T}(k) \\
&= \mathscr{I}^{-1}(k)
\end{aligned}$$

Likewise, $\hat{v}(k)$ and $Q(k)$ can be recovered from (11.2) by

$$\begin{aligned}
&\hat{v}(k) = E\left[v(k)|k\right] = R_{vv}^{-1}(k)\mathbb{E}\left[z_v(k)|Z^k\right] - R_{vv}^{-1}(k)\mathbb{E}\left[w_v(k)|Z^k\right] = 0 \\
&E\left[(v(k) - \hat{v}(k))(v(k) - \hat{v}(k))^T|Z^k\right] = Q(k)
\end{aligned}$$

11.2.2 Propagation Step

The SRIF is nothing more than a way of performing a propagation step and a measurement update on the SRI equations (11.1) and (11.2). For our derivation, we'll adopt the MAP approach, in which we seek to minimize the negative natural log of the *a posteriori* conditional probability density function. This amounts to minimizing the cost function

$$\begin{aligned}
J_a\left[x(k), v(k), x(k+1), k\right] &= -\log\left\{p[x(k+1), v(k)|Z^{k+1}]\right\} \\
&= \tfrac{1}{2}[x(k) - \hat{x}(k)]^T P^{-1}(k)[\cdots] + \tfrac{1}{2}v^T(k)Q^{-1}(k)v(k) \\
&\quad + \tfrac{1}{2}[z(k+1) - H(k+1)x(k+1)]^T R^{-1}(k+1)[\cdots]
\end{aligned}$$

[2] If $R_{xx}(k)$ is not invertible then the system is not observable from the data through time k and the *a priori* info at time 0. Note that $R_{xx}(k)$ is upper triangular.

After substitution of the previously-defined square-root quantities, we have the following equivalent cost (dropping the 1/2 multipliers for notational clarity):

$$\begin{aligned} J_a\left[x(k), v(k), x(k+1), k\right] = & \underbrace{\|R_{xx}(k)x(k) - z_x(k)\|^2}_{a\ priori\ x(k)} + \underbrace{\|R_{vv}(k)v(k) - z_v(k)\|^2}_{a\ priori\ v(k)} \\ & + \underbrace{\|H_a(k+1)x(k+1) - z_a(k+1)\|^2}_{\text{measurement at k+1}} \end{aligned}$$

Notice how the first and second terms in the cost function now have the same form as the third term, the measurement cost term. In other words, $z_x(k)$ acts as another set of measurements on $x(k)$ with measurement equation (11.1), and $z_v(k)$ acts as a set of measurements on $v(k)$ with measurement equation (11.2). This is a key insight that the SRIF makes clear: prior estimates of $x(k)$ and $v(k)$ can be written in the form of measurements on $x(k)$ and $v(k)$ and can thus be conveniently packaged together with the true measurements in the cost function.

Our task is to minimize J_a subject to the dynamics model, which relates $x(k)$, $v(k)$, and $x(k+1)$. We first solve the dynamics model for $x(k)$ in terms of $v(k)$ and $x(k+1)$:

$$x(k) = F^{-1}(k)\left[x(k+1) - G(k)u(k) - \Gamma(k)v(k)\right]$$

We can then simultaneously eliminate $x(k)$ and enforce the dynamics constraint by substituting $x(k)$ into J_a. We'll call the resulting equivalent cost function J_b:

$$\begin{aligned} & J_b\left[v(k), x(k+1), k\right] \\ & = \left\| \begin{bmatrix} R_{vv}(k) & \mathbf{0} \\ -R_{xx}(k)F^{-1}(k)\Gamma(k) & R_{xx}(k)F^{-1}(k) \end{bmatrix} \begin{bmatrix} v(k) \\ x(k+1) \end{bmatrix} - \begin{bmatrix} \mathbf{0} \\ z_x(k) + R_{xx}(k)F^{-1}(k)G(k)u(k) \end{bmatrix} \right\|^2 \\ & \quad + \|H_a(k+1)x(k+1) - z_a(k+1)\|^2 \end{aligned}$$

In packaging the equation above, we invoked the identity

$$\|a\|^2 + \|b\|^2 = \left\| \begin{bmatrix} a \\ b \end{bmatrix} \right\|^2$$

Recall that $\|Tv\|^2 = \|v\|^2$ for T orthonormal. Let $T_b(k) = Q_b^T(k)$, where $Q_b(k)$ is the orthonormal matrix that results from QR factorization of the block matrix in the equation above. Multiplying the contents of the the first norm by $T_b(k)$ and rewriting yields

$$\begin{aligned} J_b\left[v(k), x(k+1), k\right] = \quad & \left\| \begin{bmatrix} \bar{R}_{vv}(k) & \bar{R}_{vx}(k+1) \\ \mathbf{0} & \bar{R}_{xx}(k+1) \end{bmatrix} \begin{bmatrix} v(k) \\ x(k+1) \end{bmatrix} - \begin{bmatrix} \bar{z}_v(k) \\ \bar{z}_x(k+1) \end{bmatrix} \right\|^2 \\ & + \|H_a(k+1)x(k+1) - z_a(k+1)\|^2 \end{aligned}$$

This was the propagation step! Why? Because we can now isolate the *a priori* SRI equation for $x(k+1)$. Just as we took a set of SRI equations and packed them into a cost function, we can take the cost function and unpack it into a set of SRI equations. Unpacking the first term into its square root information equations, we obtain

1. The SRI equation relating the state $x(k+1)$ and the process noise $v(k)$:

   $$\bar{z}_v(k) = \bar{R}_{vv}(k)v(k) + \bar{R}_{vx}(k+1)x(k+1) + \bar{w}_v(k), \quad \bar{w}_v(k) \sim (0, I)$$

   This equation is a by-product of the filtering process. We ignore it in filtering, but we will make use of it in smoothing.

2. The *a priori* SRI equation for the state at $k+1$:

   $$\bar{z}_x(k+1) = \bar{R}_{xx}(k+1)x(k+1) + \bar{w}_x(k+1), \qquad \bar{w}_x(k) \sim (0, I)$$

11.2.3 Measurement Update

As a next step, let us minimize J_b with respect to $v(k)$. The necessary condition

$$0 = \left[\frac{\partial J_b}{\partial v(k)}\right]^{\mathrm{T}} = \underbrace{\bar{R}_{vv}^T(k)}_{\text{non-singular}} \left[\bar{R}_{vv}(k)v(k) + \bar{R}_{vx}(k+1)x(k+1) - \bar{z}_v(k)\right]$$

requires

$$v(k) = \bar{R}_{vv}^{-1}(k)\left[\bar{z}_v(k) - \bar{R}_{vx}(k+1)x(k+1)\right]$$

Substituting this into the cost function and stacking the remaining terms we get

$$J_c\left[x(k+1), k+1\right] = \left\| \underbrace{\begin{bmatrix} \bar{R}_{xx}(k+1) \\ H_a(k+1) \end{bmatrix}}_{A} x(k+1) - \begin{bmatrix} \bar{z}_x(k+1) \\ z_a(k+1) \end{bmatrix} \right\|^2 \tag{11.4}$$

If A were square and non-singular we could just take its inverse to compute $\hat{x}(k+1)$ the filter's best estimate of $x(k+1)$. But because A is not square, we instead transform the contents of the norm by QR factorizing A and multiplying the contents of the norm by the transpose of the resulting orthonormal matrix (see the **Aside** below). The transformation has the effect of decoupling the cost function into a component that depends on $x(k+1)$ and one that does not:

$$J_c[x(k+1), k+1] = \left\| \begin{bmatrix} R_{xx}(k+1) \\ 0 \end{bmatrix} x(k+1) - \begin{bmatrix} z_x(k+1) \\ z_r(k+1) \end{bmatrix} \right\|^2$$

This was the update step! Why? Because we can now isolate the *a posteriori* SRI equation for $x(k+1)$. The lack of bars above the terms denotes our having gone from *a priori* to *a posteriori.* Unstack to get

$$J_c\left[x(k+1), k+1\right] = \left\|R_{xx}(k+1)x(k+1) - z_x(k+1)\right\|^2 + \left\|z_r(k+1)\right\|^2$$

Now unpack the implicit SRI equations from this cost function to get

1. The *a posteriori* SRI equation for for the state at $k+1$:

$$z_x(k+1) = R_{xx}(k+1)x(k+1) + w_x(k+1), \qquad w_x(k+1) \sim (0, I)$$

2. The residual error equation:

$$z_r(k+1) = w_r(k+1), \qquad w_r(k+1) \sim (0, I)$$

Aside: An inside view of the measurement update step
Unpack 11.4 into its implicit square root information equations:

$$\bar{z}_x(k+1) = \bar{R}_{xx}(k+1)x(k+1) + \bar{w}_x(k+1)$$

$$z_a(k+1) = H_a(k+1)x(k+1) + w_a(k+1)$$

Let $T_c(k+1) = Q_c^T(k+1)$, where $Q_c(k+1)$ is the orthonormal matrix that results from QR factorization of A. Multiplying terms by $T_c(k)$ yields

$$\begin{bmatrix} R_{xx}(k+1) \\ 0 \end{bmatrix} = T_c(k+1) \begin{bmatrix} \bar{R}_{xx}(k+1) \\ H_a(k+1) \end{bmatrix}$$
$$\begin{bmatrix} z_x(k+1) \\ z_r(k+1) \end{bmatrix} = T_c(k+1) \begin{bmatrix} \bar{z}_x(k+1) \\ z_a(k+1) \end{bmatrix}$$
$$\begin{bmatrix} w_x(k+1) \\ w_r(k+1) \end{bmatrix} = T_c(k+1) \begin{bmatrix} \bar{w}_x(k+1) \\ w_a(k+1) \end{bmatrix}$$

Because $T_c(k+1)$ is orthonormal, $w_x(k+1)$ and $w_r(k+1)$ retain the same distribution as $\bar{w}_x(k+1)$ and $w_a(k+1)$; i.e.,

$$w_x(k+1) \sim (0, I), \qquad w_r(k+1) \sim (0, I)$$

We can now minimize J_c by inspection:

$$\hat{x}(k+1) = R_{xx}^{-1}(k+1)z_x(k+1)$$

The minimum value of J_c is

$$J_c[\hat{x}(k+1), k+1] = \|z_r(k+1)\|^2$$

It can be shown that

$$\|z_r(k+1)\|^2 = z_r^T(k+1)z_r(k+1) = \nu^T(k+1)S^{-1}(k+1)\nu(k+1) \sim \chi^2_{n_z}$$

where $\nu(k+1)$ and $S(k+1)$ are the innovation and its covariance matrix, which were defined earlier.

Remarks.

1. The SRIF avoids matrix squaring. It involves only QR factorization and inversion of R matrices when necessary. This makes it numerically robust.
2. $P(k)$ is guaranteed to be symmetric and positive definite because $P(k) = R_{xx}^{-1}(k)R_{xx}^{-T}(k)$
3. For the SRIF presented here, $F(k)$ must be invertible; if not, an SRIF implementation is possible but more complicated.

11.2.4 Step-by-Step SRIF Algorithm

1. Start at $k = 0$ with

   $$R_{xx}^T(0)R_{xx}(0) = P^{-1}(0),\ R_{vv}^T(0)R_{vv}(0) = Q^{-1}(0),\ z_v(0) = 0 = R_{vv}\hat{v}(0),\ z_x(0) = R_{xx}(0)\hat{x}(0)$$

   In Matlab, the Cholesky factorization is best done as, for example,

   $$R_{vv}(k) = [\texttt{inv(chol(}Q(k)\texttt{))}]^T$$

2. Propagation Step:

   (a) QR factorize as follows:

   $$[Q_A, R_A] = \mathtt{qr}\left(\begin{bmatrix} R_{vv}(k) & 0 \\ -R_{xx}(k)F^{-1}(k)\Gamma(k) & R_{xx}(k)F^{-1}(k) \end{bmatrix}\right)$$

   (b) Perform an orthonormal transformation as

   $$\begin{bmatrix} \bar{z}_v(k) \\ \bar{z}_x(k+1) \end{bmatrix} = Q_A^T \begin{bmatrix} 0 \\ z_x(k) + R_{xx}(k)F^{-1}(k)G(k)u(k) \end{bmatrix} \tag{11.5}$$

   $$\begin{bmatrix} \bar{R}_{vv}(k) & \bar{R}_{vx}(k+1) \\ 0 & \bar{R}_{xx}(k+1) \end{bmatrix} = R_A$$

   (c) Extract $\bar{R}_{xx}(k+1)$ and $\bar{z}_x(k+1)$.

3. Measurement Update:

   (a) Cholesky factorize $R(k+1)$ as $R_a(k+1)^T R_a(k+1) = R(k+1)$:

   $$R_a(k+1) = \mathtt{chol}(R(k+1))$$

   (b) Transform $z(k+1)$ and $H(k+1)$ as

   $$z_a(k+1) = R_a^{-T}(k+1)z(k+1), \quad H_a(k+1) = R_a^{-T}(k+1)H(k+1)$$

   (c) Perform another QR factorization:

   $$[Q_B, R_B] = \mathtt{qr}\left(\begin{bmatrix} \bar{R}_{xx}(k+1) \\ H_a(k+1) \end{bmatrix}\right)$$

   (d) Transform as

   $$\begin{bmatrix} z_x(k+1) \\ z_r(k+1) \end{bmatrix} = Q_B^T \begin{bmatrix} \bar{z}_x(k+1) \\ z_a(k+1) \end{bmatrix}$$

   $$\begin{bmatrix} R_{xx}(k+1) \\ 0 \end{bmatrix} = R_B$$

   (e) Extract $R_{xx}(k+1)$ and $z_x(k+1)$.

   (f) If necessary (e.g., for a downstream control system that needs a state estimate), obtain $\hat{x}(k+1)$ as

   $$\hat{x}(k+1) = R_{xx}^{-1}(k+1)z_x(k+1)$$

4. If $k+1$ is the index of the final measurement, then stop. Otherwise, increment k and return to step 2 using the values for $z_x(k)$ and $R_{xx}(k)$ found in step 3(e).