Kalman Filtering with Known Forcing Function

Consider the following system:

$$x_{k+1} = F_k x_k + G_k w_k + d_{k+1} \quad \textbf{(1a)}$$

$$z_k = H_k x_k + v_k + r_k \quad \textbf{(1b)}$$

where the state and measurement processes involve *known* input functions d_k and r_k, respectively.

Recall, that the Kalman filter utilizes the error

$$e_k = z_k - \hat{z}_k \quad \textbf{(2a)}$$

multiplied by the Kalman gain, K, to recover an estimate of the state, x_k where, in the case of (1),

$$\hat{z}_k = H_k \hat{x}_k + r_k. \quad \textbf{(2b)}$$

The current and one-step ahead state estimates are

$$\hat{x}_k = \hat{x}_k^- + K_k e_k \quad \textbf{(3a)}$$

and

$$\hat{x}_{k+1}^- = F_k \hat{x}_k + d_{k+1}. \quad \textbf{(3b)}$$

Now, let's see how these modified measurement (2b) and state (3b) predictor equations are incorporated into the Kalman filter algorithm.

***Step 1*: For k=0 choose a value for $\hat{x}_0^-$ and compute the prediction error variance:**

$$P_0^- = E[(x_0 - \hat{x}_0^-)(x_0 - \hat{x}_0^-)^{tr}].$$

If we choose $\hat{x}_0^- = d_0$, then we have $P_0^- = E[\tilde{x}_0 \tilde{x}_0^{tr}]$ where $\tilde{x}_0 \overset{\Delta}{=} x_0 - d_0$

***Step 2*: For k=0: Compute the Kalman gain:** $K_k = P_k^- H_k^{tr} (H_k P_k^- H_k^{tr} + R)^{-1}$.

***Step 3*: Compute the update estimate:** $\hat{x}_k = \hat{x}_k^- + K_k [z_k - (H_k \hat{x}_k^- + r_k)]$.

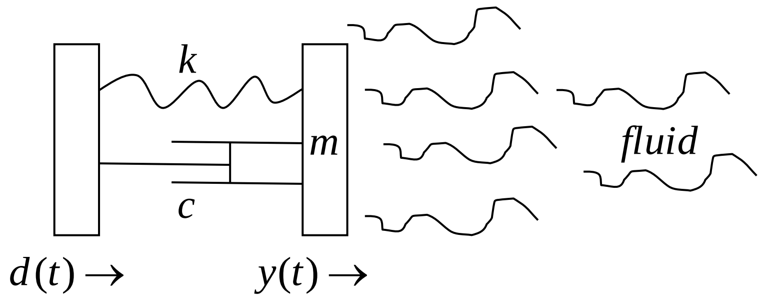
***Step 3*: Compute the update error variance:** $P_k = (I - K_k H_k) P_k^-$.

***Step 4*: Compute 1-step prediction and associated error variance:**

$$\hat{x}_{k+1}^- = F_k \hat{x}_k + d_{k+1} \quad ; \quad P_{k+1}^- = F_k P_k F_k^{tr} + G_k G_k^{tr}$$

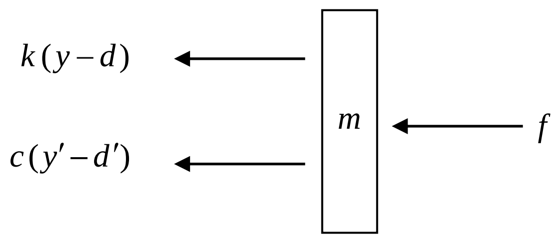
***Step 5*: Go to *Step 2* and increment k by one.** □

***Example*: Prediction of the 1-D position and velocity of a flexible robot arm:**



The fluid has a mean viscosity that is included as a part of the viscous damping coefficient *c*. However, the fluid also has turbulence that results in zero-mean fluctuating loading of the mass.

To develop the equation for the motion of *m*, construct a free body diagram:



$\sum forces = m\ddot{y} = -c(\dot{y}-\dot{d}) - k(y-d) - f$. **Hence,** $m\ddot{y} + c\dot{y} + ky = c\dot{d} + kd - f$. **Assume the system is *underdamped*, and define** $\omega_n = \sqrt{\dfrac{k}{m}}$, $\zeta = \dfrac{c}{2\sqrt{km}}$, $\omega_d = \omega_n\sqrt{1-\zeta^2}$ **. Then we have**

$$\ddot{y} + 2\zeta\omega_n\dot{y} + \omega_n^2 y = 2\zeta\omega_n\dot{d} + \omega_n^2 d - (1/m)f \; . \qquad \textbf{(1)}$$

The system (1) is a 2-input / single output system. We will use the *impulse-invariant* method of obtaining the discrete-time version of (1). To this end, we will require the following transforms:

$\dfrac{b}{(s+a)^2+b^2}$	$e^{-at}\sin(bt)$	$\dfrac{ze^{-aT}\sin(bT)}{z^2-2ze^{-aT}\cos(bT)+e^{-2aT}}$
$\dfrac{s+a}{(s+a)^2+b^2}$	$e^{-at}\cos(bt)$	$\dfrac{z^2-ze^{-aT}\cos(bT)}{z^2-2ze^{-aT}\cos(bT)+e^{-2aT}}$

The transfer function between *d*(*t*) and *y*(*t*) is:

$$\frac{Y_d(s)}{D(s)} = H_1(s) = \frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} = \frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2\zeta\omega_n s + (\zeta\omega_n)^2 + \omega_n^2 - (\zeta\omega_n)^2} = \frac{2\zeta\omega_n s + \omega_n^2}{(s+\zeta\omega_n)^2 + \omega_d^2} .$$

$$H_1(s) = 2\zeta\omega_n \bullet \frac{s + (\omega_n/2\zeta) + \zeta\omega_n - \zeta\omega_n}{(s+\zeta\omega_n)^2 + \omega_d^2} = 2\zeta\omega_n\left[\frac{s+\zeta\omega_n}{(s+\zeta\omega_n)^2+\omega_d^2} + \frac{(\omega_n/2\zeta) - \zeta\omega_n}{(s+\zeta\omega_n)^2+\omega_d^2}\right]$$

$$H_1(s) = 2\zeta\omega_n \left[\frac{s+\zeta\omega_n}{(s+\zeta\omega_n)^2+\omega_d^2} + \frac{[(\omega_n/2\zeta)-\zeta\omega_n](\omega_d/\omega_d)}{(s+\zeta\omega_n)^2+\omega_d^2} \right].$$

Define $c = [(\omega_n/2\zeta)-\zeta\omega_n]/\omega_d$ **. We then have**

$$H_1(s) = 2\zeta\omega_n \left[\frac{s+\zeta\omega_n}{(s+\zeta\omega_n)^2+\omega_d^2} + c \bullet \frac{\omega_d}{(s+\zeta\omega_n)^2+\omega_d^2} \right] \qquad (2a)$$

The discrete transfer function associate with (2a) is therefore

$$H_1(z) = 2\zeta\omega_n \left[\frac{1-[e^{-\zeta\omega_n T}\cos(\omega_d T)]z^{-1}}{1-2[e^{-\zeta\omega_n T}\cos(\omega_d T)]z^{-1}+e^{-2\zeta\omega_n T}z^{-2}} + c \bullet \frac{[e^{-\zeta\omega_n T}\sin(\omega_d T)]z^{-1}}{1-2[e^{-\zeta\omega_n T}\cos(\omega_d T)]z^{-1}+e^{-2\zeta\omega_n T}z^{-2}} \right]$$

$$H_1(z) = 2\zeta\omega_n \left[\frac{1+e^{-\zeta\omega_n T}[c \bullet \sin(\omega_d T)-\cos(\omega_d T)]z^{-1}}{1-2[e^{-\zeta\omega_n T}\cos(\omega_d T)]z^{-1}+e^{-2\zeta\omega_n T}z^{-2}} \right]. \qquad \textbf{(2b)}$$

The transfer function between *f*(*t*) and *y*(*t*) is:

$$\frac{Y_f(s)}{F(s)} = H_2(s) = (1/m\omega_d) \bullet \frac{\omega_d}{(s+\zeta\omega_n)^2+\omega_d^2}. \qquad \textbf{(3a)}$$

The discrete transfer function associate with (3a) is therefore

$$H_2(z) = (1/m\omega_d) \bullet \frac{[e^{-\zeta\omega_n T}\sin(\omega_d T)]z^{-1}}{1-2[e^{-\zeta\omega_n T}\cos(\omega_d T)]z^{-1}+e^{-2\zeta\omega_n T}z^{-2}}. \qquad \textbf{(3b)}$$

Hence, we have

$$Y(z) = Y_d(z) + Y_f(z) = H_1(z)D(z) + H_2(z)F(z).$$

Specifically,

$$[1-2[e^{-\zeta\omega_n T}\cos(\omega_d T)]z^{-1}+e^{-2\zeta\omega_n T}z^{-2}]Y(z) = 2\zeta\omega_n\{1+e^{-\zeta\omega_n T}[c \bullet \sin(\omega_d T)-\cos(\omega_d T)]z^{-1}\}D(z) + [(1/m\omega_d)e^{-\zeta\omega_n T}\sin(\omega_d T)]z^{-1}F(z) \qquad \textbf{(4a)}$$

Define the parameters:

$$[a_1 = 2[e^{-\zeta\omega_n T}\cos(\omega_d T)] \quad ; \quad a_2 = -e^{-2\zeta\omega_n T} \; ;$$
$$b_1 = 2\zeta\omega_n \quad ; \quad b_2 = 2\zeta\omega_n e^{-\zeta\omega_n T}[c \bullet \sin(\omega_d T)-\cos(\omega_d T)] \quad ; \quad b_3 = -(1/m\omega_d)e^{-\zeta\omega_n T}\sin(\omega_d T).$$

Then the difference equation corresponding to (4a) is:

$$y_k = a_1 y_{k-1} + a_2 y_{k-2} + b_1 d_k + b_2 d_{k-1} + b_3 f_{k-1}. \qquad \textbf{(4b)}$$

We will now assume that the fluid force adheres to an AR(2) model, so that spectral peaking associated with vortex shedding, etc. may be incorporated.

$$f_k = c_1 f_{k-1} + c_2 f_{k-2} + u_k . \qquad \textbf{(5)}$$

Finally, we will assume that the mass incorporates a position sensor, and that there is sensor measurement noise. Then the measurement model is

$$z_k = y_k + v_k . \qquad \textbf{(6)}$$

We are now in a position to formulate the state and measurement equations associated with the Kalman filter parameters. To this end, define the known <u>deterministic</u> input $\partial_k = b_1 d_k + b_2 d_{k-1}$. **Then (4b) becomes**

$$y_k = a_1 y_{k-1} + a_2 y_{k-2} + \partial_k + b_3 f_{k-1} . \qquad \textbf{(7)}$$

Now, define the state vector $x_k = [y_k \; y_{k-1} \; f_k \; f_{k-1}]^{tr}$ **. From (5) and (7) we have**

$$\begin{bmatrix} y_{k+1} \\ y_k \\ f_{k+1} \\ f_k \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & b_3 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & c_1 & c_2 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} y_k \\ y_{k-1} \\ f_k \\ f_{k-1} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ u_{k+1} \\ 0 \end{bmatrix} + \begin{bmatrix} \partial_k \\ 0 \\ 0 \\ 0 \end{bmatrix} \Rightarrow x_{k+1} = F x_k + \vec{u}_k + \vec{\partial}_k . \qquad \textbf{(8a)}$$

$$z_k = y_k + v_k = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} x_k + v_k \quad \Rightarrow \quad z_k = H x_k + v_k . \qquad \textbf{(8b)}$$

From (8a) we have $Q = diag\{0 \; 0 \; \sigma_u^2 \; 0\}$, and from (8b) $R = \sigma_v^2$. For the state initial condition $\hat{x}_0^- = [y_0 \; y_{-1} \; f_0 \; f_{-1}]' + [\partial_0 \, 0 \, 0 \, 0]' = [0 \, 0 \, 0 \, 0]' + [\partial_0 \, 0 \, 0 \, 0]' = [\partial_0 \, 0 \, 0 \, 0]'$, the initial prediction error covariance is

$$P_0^- = Cov(x_0 - \hat{x}_0^-) = Cov(y_0 - \partial_0 , y_{-1} \, , f_0 . f_{-1}) . \qquad (9)$$

Since ∂_0 is deterministic, it does not influence any covariances. Hence, (9) becomes:

$$P_0^- == Cov(y_0 , y_{-1} \, , f_0 . f_{-1}) \begin{bmatrix} R_y(0) & R_y(1) & R_{yf}(0) & R_{yf}(-1) \\ R_y(1) & R_y(0) & R_{yf}(1) & R_{yf}(0) \\ R_{yf}(0) & R_{yf}(1) & R_f(0) & R_f(1) \\ R_{yf}(-1) & R_{yf}(0) & R_f(1) & R_f(0) \end{bmatrix} \qquad (10)$$

Now recall that the state equation 8(a) for k=0:

$$x_0 = F x_{-1} + \vec{u}_{-1} \qquad (12)$$

Then, $P_0^- = R_x(0)$ This gives:

$$E(x_0 x_0^{tr}) = FE(x_{-1} x_0^{tr}) + E(\vec{u}_{-1} x_0^{tr}) \Rightarrow R_x(0) = FR_x(1) + E(\vec{u}_{-1} x_0^{tr}) \qquad (13a)$$

$$E(x_0 x_{-1}^{tr}) = FE(x_{-1} x_{-1}^{tr}) + E(\vec{u}_{-1} x_{-1}^{tr}) \Rightarrow R_x(1)^{tr} = FR_x(0) + E(\vec{u}_{-1} x_{-1}^{tr}) \qquad (13b)$$

Now:

$$E(\vec{u}_{-1}x_0^{tr})=E(\begin{bmatrix}0\\0\\u_0\\0\end{bmatrix}\begin{bmatrix}y_0 & y_{-1} & f_0 & f_{-1}\end{bmatrix})=\begin{bmatrix}0&0&0&0\\0&0&0&0\\E(y_0u_0)&0&\sigma_u^2&0\\0&0&0&0\end{bmatrix}=\begin{bmatrix}0&0&0&0\\0&0&0&0\\\sigma_u^2&0&\sigma_u^2&0\\0&0&0&0\end{bmatrix}\overset{\Delta}{=}C_u\,.$$

Also: $E(\vec{u}_{-1}x_{-1}^{tr})=E(\begin{bmatrix}0\\0\\u_0\\0\end{bmatrix}\begin{bmatrix}y_{-1} & y_{-2} & f_{-1} & f_{-2}\end{bmatrix})=\mathbf{0}$

Hence: $$R_x(0)=FR_x(1)+C_u \tag{14a}$$

$$R_x(1)^{tr}=FR_x(0)\quad\Rightarrow R_x(1)=R_x(0)^{tr}F^{tr}=R_x(0)F^{tr} \tag{14b}$$

Substituting (14b) into (14a): $$R_x(0)=FR_x(0)F^{tr}+C_u\,. \tag{15}$$

Equation (15) can be written as: $FR_x(0)F^{tr}-R_x(0)+C_u=0$. This is called the discrete-time *Lyapunov Equation* for the variable $R_x(0)$. The Matlab code 'dlyap(F,Cu) will solve it.

As simple as the Matlab solution is, it lacks insight into the nature of the elements of $P_0^-=R_x(0)$. From (10) it is clear that those elements are auto- and cross-correlations at lags zero and one. So, one might ask: How can we arrive at these correlations via psd information. We now present a method that recovers all auto/cross-correlations based on psds.

Recall: $Y_f(z)=H_2(z)F(z)$ and $f_k=c_1f_{k-1}+c_2f_{k-2}+u_k$ gives $F(z)=\dfrac{1}{1-c_1z^{-1}-c_2z^{-2}}U(z)=H_{fu}(z)U(z)$

Hence, $Y_f(z)=H_2(z)H_{fu}(z)U(z)=\breve{H}(z)U(z)$. From the Wiener-Kinchine Theorem we therefore have:

(i): $S_f(z)=E\left(\left|H_{fu}(z)U(z)\right|^2\right)=\left|H_{fu}(z)\right|^2\sigma_u^2$

(ii): $S_{yf}(z)=E\left(Y_f(z)\bar{F}(z)\right)=E\left[H_2(z)F(z)\bar{F}(z)\right]=H_2(z)\left|H_{fu}(z)\right|^2\sigma_u^2$

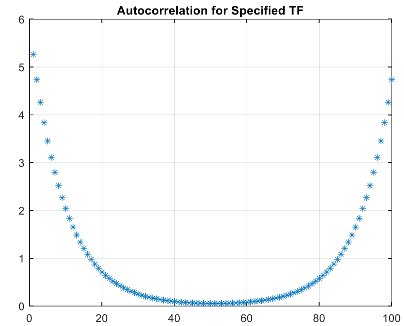
(iii): $S_y(z)=E\left(Y_f(z)\bar{Y}_f(z)\right)=E\left[H_2(z)F(z)\bar{H}_2(z)\bar{F}(z)\right]=\left|H_2(z)H_{fu}(z)\right|^2\sigma_u^2$

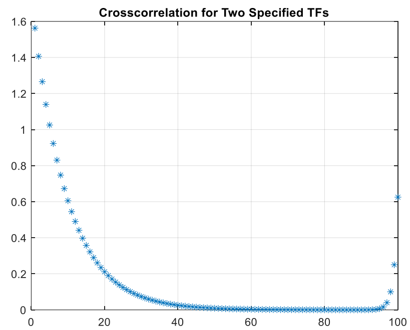
For numerical values, we can compute the quantities and then recover the associated correlation functions. Note that this is not necessary in relation to (i), as we have a direct method of solving for the autocorrelation for an AR(2) process.

NOTE: To recover the coefficient of a transfer function H, type: [num,den]=tfdata(H,'v').

The code below recovers correlation functions from psds specified via transfer functions.

```
%PROGRAM NAME: psd2xcorr.m
%This code recovers Rxy from Sxy in dicrete time domain
%============================================
%Examples:
m=100;
%TF1:
H1=tf([1 0],[1,-.9],1); %Transfer Function
[H1n,H1d]=tfdata(H,'v');
H1nw=fft(H1n,m); H1dw=fft(H1d,m);
H1w=H1nw./H1dw;
%TF2:
%H2=H1; %For Autocorrelation
H2=tf([1 0],[1,-.4],1); %For Crosscorrelation
[H2n,H2d]=tfdata(H2,'v');
H2nw=fft(H2n,m); H2dw=fft(H2d,m);
H2w=H2nw./H2dw;
%Cross-Spectrum % Cross-Correlation
S12=H1w.*conj(H2w);
R12=real(ifft(S12));
figure(1)
plot(R12,'*')
title('Crosscorrelation for Two Specified TFs')
grid
```





Comment: I checked to make sure that the imaginary parts of the ifft's were zero. They were. ☺