The Cooper Union Department of Electrical Engineering
Prof. Fred L. Fontaine
ECE416 Adaptive Filters
Project: Kalman Filters
June 23, 2019

In this project, you will be implementing and studying the performance of several forms of the discrete-time Kalman filter, specifically:

- the "standard" (i.e., covariance) Kalman filter
- the information Kalman filter
- the square-root covariance Kalman filter
- the extended Kalman filter (EKF)

You can use numerical routines for matrix analysis in MATLAB, such as *eig, qrd, svd,* etc. However, do NOT use "advanced" MATLAB functions related to Kalman filtering, such as *kalman,* except as noted below (e.g., find an appropriate function to solve the discrete-time ARE, as described below).
NOTE: All data is real in this project!!! Also, in this write-up: for a vector u, $|u|$ is the Euclidean length; and for a matrix A, $\|A\|$ is the spectral norm.
In the first three cases, the system model will be TIME-VARYING and is given by:

$$\begin{aligned} x(n+1) &= A(n)\,x(n) + v(n) \\ y(n) &= C(n)\,x(n) + w(n) \end{aligned}$$

In order to examine the effect of time-varying behavior on the system, for iterations $1 \le n \le N_1$, take $A(n) = A_1$, and for $N_1 + 1 \le n \le N_1 + N_2$, take $A(n) = A_2$ where:

$$A_1 = \begin{bmatrix} -0.9 & 1 & 0 & 0 \\ 0 & -0.9 & 0 & 0 \\ 0 & 0 & -0.5 & 0.5 \\ 1 & 0 & -0.5 & -0.5 \end{bmatrix} \qquad A_2 = \begin{bmatrix} 0.9 & 1 & 0 & 0 \\ 0 & 0.9 & 0 & 0 \\ 0 & 0 & -0.5 & 0.5 \\ 1 & 0 & -0.5 & -0.5 \end{bmatrix}$$

For all times take:

$$C = \begin{bmatrix} 1 & -1 & 1 & -1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Assume the covariance matrices Q_v and Q_w of the process noise $v(n)$ and observation noise $w(n)$, respectively, are constant and given by:

$$Q_v = \begin{bmatrix} 1/4 & 1/4 & 0 & 0 \\ 1/4 & 1/2 & 1/4 & 0 \\ 0 & 1/4 & 1/2 & 1/4 \\ 0 & 0 & 1/4 & 1/2 \end{bmatrix} \qquad Q_w = 0.1I$$

Studying the Prescribed System

Start by computing the eigenvalues of A_1 and A_2. Confirm: each is stable; each has an eigenvalue of algebraic multiplicity 2 but geometric multiplicity 1; there are also a pair of complex eigenvalues. You should see the significance of the "change" from A_1 to A_2.
Given the following model:

$$x(n+1) = Ax(n) + v(n)$$

where $v(n)$ is white with covariance matrix Q_v, and assuming A is stable, the steady-state covariance matrix of $x(n)$ satisfies the *discrete-time Lyapunov equation:*

$$K = AKA^T + Q_v$$

Do not convuse this K with the covariance matrices of the Kalman state filtered and predicted ERRORS. This $K = \lim_{n\to\infty} E\left(x(n)\, x^T(n)\right)$ does not depend on the initial condition $x(0)$ (assuming stability). There is a MATLAB function that computes the solution to the discrete-time Lyapunov equation. Find it for the matrices A_1, A_2, with the prescribed Q_v. This will give you a sense of the "size" of the $\vec{x}$ at steady-state. The eigenvalues of A can give you a sense as to how long until steady-state conditions are met; indeed, let p be the magnitude of the dominant eigenvalue in A, and let us take N to be the smallest value such that $p^N < 0.01$. We can use N to model the time until the effect of initial condition $x(0)$ dissipates. When you run the nonstationary model, take $N1$ and $N2$ to be this computed parameter N for the respective A_1, A_2 matrices.
Check that in each case (A, C) is observable (form the observability matrix and compute its rank). Combined with the fact that Q_v is full-rank, that means each matrix would be associated with a steady-state prediction error covariance matrix that solves an algebraic Riccati equation (ARE). Find the MATLAB function that solves the discrete-time ARE, and use it to compute these steady-state covariance matrices, say $K_{1,ss}$ and $K_{2,ss}$. Note that you will not only have to find the function, but find how to "call it" correctly. Compute the eigenvalues of these covariance matrices, confirming they are positive definite. Also compute $\|K_{1,ss} - K_{2,ss}\|$, where $\|\cdot\|$ denotes spectral norm. This is one way of seeing the significance of switching the matrices.
Now just do an experiment. Start with initial condition $x(0)$ the all 1's vector. Run the process equation for N time stepts using A_1, Now switch to $A2$ and run for an additional N time steps. Now plot the state vector versus time (i.e, $plot(1 : endtime,\ x)$). Observe the different behavior under $A1$ and $A2$, and the transition when the matrices switch.

Covariance Kalman Filter

Write a "core" function that runs one iteration of the Kalman filter, taking as input $\hat{x}(n|n-1)$ and $K(n, n-1)$. It should return $\hat{x}(n|n)$ and $\hat{x}(n+1|n)$ as well as $K(n, n)$, $K(n+1, n)$, and $G(n)$.
As you perform all the following, I want you to monitor (after each iteration) that $K(n, n) > 0$; I suspect it will be. However, if you ever detect a failure of this condition— continue the experiment to the full number of iterations, and see what happens; and then switch over to the Joseph form to get correct results.

First do a preliminary study. In each case, create initial $x(0)$ as a vector with iid $N(0,1)$ components. Take $K(1,0)$ as the identity. First take A_1 (constant), and repeat 100 times and find the average number of iterations it takes for $\|K(n,n-1) - K_{1,ss}\| / \|K_{1,ss}\| < 0.01$. Do the same for A_2.
So let N_1 and N_2 be the average times you found. If either is < 10, replace them with 10 (i.e., enforce that $N_1 \geq 10$ and $N_2 \geq 10$). Now run the routine with, first N_1 iterations with matrix $A(n) = A_1$ followed by an IMMEDIATE switch to A_2 and another N_2 iterations with $A(n) = A_2$. First, graph $|x(n)|$, and then do a 3-D plot of (x_1, x_2, x_3) (lines connecting the discrete points) to get a sense of the actual state trajectory. Repeat this a couple times just to get a sense that what you are showing appears to be "typical". You may want to superimpose results of 5 trial runs, for example. Maybe by using two colors, clearly indicate the point where the switch from A_1 to A_2 occurs. Superimpose graphs of $|x(n) - \hat{x}(n|n-1)|$, $|x(n) - \hat{x}(n|n)|$. Also graph $\|K(n,n-1) - K_{i,ss}\|$ where $K_{i,ss}$ is $K_{1,ss}$ for $n \leq N_1$, and $K_{2,ss}$ for $N_1+1 \leq n \leq N_1+N_2$. You should repeat this a couple times just to check the results you present are "typical". I am not saying average them- just make sure there isn't a statistical fluke.
Finally, show the following graphs: $\|K(n,n-1)\|$, $\|K(n,n)\|$ and $\|K(n,n) - K(n-1,n-1)\|$.

Information Kalman Filter

Basically repeat the above. With the inverse covariance matrix denoted P, note that for example we should have $P_{1,ss} = K_{1,ss}^{-1}$ and so forth. Now instead of graphing $\|K\|$ or $\|K - K\|$, graph $\|P\|$ or $\|P - P\|$.
Now go back. Do ONE test (not 100 and average): take $K(1,0) = 10^4$; run the covariance Kalman filter, and the information Kalman filter, and see if you notice a difference in performance. Keep in mind you may need to run more iterations just to get to the steady-state covariance matrix.

Square-Root Kalman Filter

You are going to apply the square-root covariance Kalman filter to the same time-varying system as described above. Pretty much repeat the experiment for the covariance Kalman filter. See if you notice any appreciable difference in the performance.

Extended Kalman Filter

Here we are going to examine the EKF applied to different types of nonlinearities. We will have 4 state variables and 4 measurements (i.e., $\vec{x}$ and $\vec{y}$ vectors are both in $\mathbb{R}^4$).
The process and observation equations are now:

$$\begin{aligned} x(n+1) &= f(x(n), n) + v(n) \\ y(n) &= h(x(n), n) + w(n) \end{aligned}$$

where $f(\cdot)$ and $h(\cdot)$ each apply a nonlinear operations. Assume v, w are white noise with the same $Q_v = 0.2I$ and $Q_w = 0.1I$ (note there are , Q_w as above. We are going to take two

cases: first where there is a high degree of nonlinearity, and in particular h is non-monotonic. In the second case, f is not differentiable.. So first take:

$$f(\vec{x}) = \begin{bmatrix} 10\pi \sin(x_1 \cdot x_3) \\ 10\pi \sin(x_2 \cdot x_4) \\ 10\pi \sin(x_3) \\ 10\pi \sin(x_4) \end{bmatrix}$$

and $h(\cdot)$ is the following function applied componentwise:

$$h(x) = \sin(x)$$

Note that each state variable is bounded by 10π in magnitude, but the measurement function $h(\cdot)$ "folds" the value of x (i.e., multiple values of x map to the same y).
Take as initial condition $x(0)$ the all 1's vector, and run this for 100 time steps, and plot the state and, separately, the measurements y versus time.
Compute the Jacobians $F(\vec{x})$ and $H(\vec{x})$ and run the EKF for 100 time steps. Take the filtered state estimate $\hat{x}(n|n)$, and plot $\|x(n) - \hat{x}(n|n)\|$. Use reasonable initial conditions. Repeat this experiment for the case $h(x) = \tan^{-1}(x)$ (applied componentwise), which is monotonic. Does the EKF perform any better?
Now we are going to take $f(x)$ to be nondifferentiable. Let $g(\zeta) = |\zeta|$ for scalar ζ, and for a vector we apply it componentwise. Let the process equation have the form:

$$x(n+1) = g(A_1 x(n)) + v(n)$$

In order to compute the Jacobian $F(\vec{x})$ we would need $g'(\zeta)$, which of course is not defined at 0. So replace g' in the formula with the function:

$$g^d(\zeta) = \begin{cases} 1 & \zeta > \epsilon \\ 0 & |\zeta| \leq \epsilon \\ -1 & \zeta < -\epsilon \end{cases}$$

Since we want an extreme effect, take $\epsilon = 0.1$. From this, you can obtain a function $F(\vec{x})$ to use for the EKF. Take $h(x) = \tan^{-1}(x)$, and before you start with the Kalman filter, synthesize the state vector and output over 100 time steps, with initial condition $x(0)$ the all $1's$ vector, and plot them as before. Then run the EKF for 100 time steps, with reasonable initial conditions. Plot $\|x(n) - \hat{x}(n|n)\|$.