Posting Date: Monday Oct. 21st.

You need not hand in anything. Instead, be prepared to answer any of these problems on an upcoming take-home prelim exam. You may discuss your solutions with classmates up until the time that the prelim becomes available.

1. Suppose that one defines the following weighted least-squares cost function for purposes of developing a maximum a posteriori implementation of the Kalman filter:

$$J_a[\underline{x}(k),\underline{v}(k),\underline{x}(k+1),k] = \tfrac{1}{2}[\underline{x}(k)-\hat{\underline{x}}(k)]^{\mathrm{T}} P^{-1}(k)[\underline{x}(k)-\hat{\underline{x}}(k)] + \tfrac{1}{2}\underline{v}^{\mathrm{T}}(k)Q^{-1}(k)\underline{v}(k)$$
$$+ \tfrac{1}{2}[\underline{z}(k+1)-H(k+1)\underline{x}(k+1)]^{\mathrm{T}} R^{-1}(k+1)[\underline{z}(k+1)-H(k+1)\underline{x}(k+1)]$$

Next, suppose that one uses the dynamics model

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

in order to eliminate $\underline{x}(k)$ from this cost and thereby define:

$$J_b[\underline{v}(k),\underline{x}(k+1),k] = J_a[F^{-1}(k)\{\underline{x}(k+1)-G(k)\underline{u}(k)-\Gamma(k)\underline{v}(k)\},\underline{v}(k),\underline{x}(k+1),k]$$

Next, suppose that one partially optimizes this cost function with respect to $\underline{v}(k)$, which yields the formula

$$\underline{v}(k) = Q(k)\Gamma^{\mathrm{T}}(k)\overline{P}^{-1}(k+1)[\underline{x}(k+1)-F(k)\hat{\underline{x}}(k)-G(k)\underline{u}(k)]$$

Next, suppose that one uses this formula to eliminate $\underline{v}(k)$ from the cost function, which effectively defines a new cost function:

$$J_c[\underline{x}(k+1),k+1] = J_b[Q(k)\Gamma^{\mathrm{T}}(k)\overline{P}^{-1}(k+1)\{\underline{x}(k+1)-F(k)\hat{\underline{x}}(k)-G(k)\underline{u}(k)\},\underline{x}(k+1),k]$$

Prove that

$$J_c[\underline{x}(k+1),k+1] = \tfrac{1}{2}[\underline{x}(k+1)-\hat{\underline{x}}(k+1)]^{\mathrm{T}} P^{-1}(k+1)[\underline{x}(k+1)-\hat{\underline{x}}(k+1)]$$
$$+ J_b[\hat{\underline{v}}(k),\hat{\underline{x}}(k+1),k]$$

where

$$\hat{\underline{v}}(k) = Q(k)\Gamma^{\mathrm{T}}(k)\overline{P}^{-1}(k+1)[\hat{\underline{x}}(k+1)-F(k)\hat{\underline{x}}(k)-G(k)\underline{u}(k)]$$

Assume that all un-defined matrices and vectors in this problem formulation are the same as have been defined in lecture for the Kalman filter problem.

Hints: Both forms of $J_c[\underline{x}(k+1),k+1]$ are quadratic in the argument $\underline{x}(k+1)$. Equality of the two forms of $J_c[\underline{x}(k+1),k+1]$ can be proved by proving that the respective coefficients of the constant, linear, and quadratic $\underline{x}(k+1)$ terms are equal. You may need to try several of the alternate equivalent formulas for $P(k+1)$ in order to find the one that is most useful for purposes of this proof. Alternatively, one could prove that the two quadratic cost functions are the same by showing a) that the same value of $\underline{x}(k+1)$ minimizes both cost functions, b) that the corresponding minimum values of the two functions are identical, and c) that the two cost functions have identical Hessian matrices, i.e., that each has the same symmetric n_x-by-n_x matrix of 2nd partial derivatives with respect to the various elements of $\underline{x}(k+1)$.

2. Using the same notation and definitions as in Problem 1, prove that

$$\frac{1}{2}\underline{\nu}^{\mathrm{T}}(k+1)S^{-1}(k+1)\underline{\nu}(k+1) = J_b[\underline{\hat{v}}(k), \underline{\hat{x}}(k+1), k]$$

where $\underline{\nu}(k+1)$ is the Kalman filter innovation at time $k+1$ and $S(k+1)$ is its a priori covariance.

Hints: Re-write the terms in $J_b[\underline{\hat{v}}(k), \underline{\hat{x}}(k+1), k]$ in terms of the innovation $\underline{\nu}(k+1)$. Note that the innovation is distinct from the process noise $\underline{v}(k)$. The innovation symbol is the Greek letter "nu", and the process noise symbol is the Latin letter "vee". You should be able to reduce the cost expression to a quadratic form in $\underline{\nu}(k+1)$. It may help to use the definition of J_b in terms of J_a and to first express the terms $[\underline{\hat{x}}(k|k+1) - \underline{\hat{x}}(k)]$, $\underline{\hat{v}}(k)$, and $[\underline{z}(k+1) - H(k+1)\underline{\hat{x}}(k+1)]$ as simple linear functions of $\underline{\nu}(k+1)$. Note that $\underline{\hat{x}}(k|k+1)$ = $F^{-1}(k)\{\underline{\hat{x}}(k+1) - G(k)\underline{u}(k) - \Gamma(k)\underline{\hat{v}}(k)\}$. After doing this, it will be necessary to use matrix manipulations in order to prove that the quadratic form's weighting matrix is $S^{-1}(k+1)$. Recall that $S(k+1)$ has been defined in lecture and in Bar-Shalom. The final matrix manipulations will be easier if you first simplify your expressions for $[\underline{\hat{x}}(k|k+1) - \underline{\hat{x}}(k)]$, $\underline{\hat{v}}(k)$, and $[\underline{z}(k) - H(k+1)\underline{\hat{x}}(k+1)]$ as much as possible.

3. Implement a Kalman filter for the example problem that was presented in class. The problem matrices and the measurement data, $\underline{z}(k)$ for k = 1, ..., 50, can be loaded into your MATLAB work space by running the MATLAB script "kf_example02a.m". Hand in plots of the two elements of $\underline{\hat{x}}_k$ vs. time and of the predicted standard deviations of $\underline{\hat{x}}(k)$ vs. time, i.e., of $\sqrt{[P(k)]_{11}}$ and $\sqrt{[P(k)]_{22}}$. Plot each element of $\underline{\hat{x}}(k)$ and its corresponding standard deviation together on the same graph. Use symbols on the plot at each of the 51 points and do not connect the symbols by lines (type "help plot" in order to learn how to do this). Also, hand in numerical values for the terminal values of $\underline{\hat{x}}(50)$ and $P(50)$.

4. Use "dlqe.m" to calculate the steady-state Kalman filter gain, W_{ss}, and the steady-state *a posteriori* state estimation error covariance matrix, P_{ss}, for the system of Problem 3. Show that the time-varying *a posteriori* covariance matrix time history that you computed in Problem 3 converges to the steady-state covariance matrix that you computed in this part. Also, show that the steady-state value of the error transition matrix, $(I - W_{ss}H)F$, is stable, i.e., that all of its eigenvalues have complex magnitudes less than 1.

5. Repeat Problem 3, except use the problem matrices and measurement data that are defined by the MATLAB script "kf_example02b.m". Notice that the R and Q values are different for this problem and that there is a different measurement time history. Run your Kalman filter two additional times using the two alternate Q values that are mentioned in the comments in the file "kf_example02b.m". It is uncertain which is the correct Q value. Decide which is the

best value in the following way: Calculate $\varepsilon_\nu(k)$ for k = 1, …, 50 for each of your runs. Compute the average of these 50 values. This average times 50, i.e., $\{\varepsilon_\nu(1)+\ldots+\varepsilon_\nu(50)\}$, will be a sample of a degree 50 X^2 distribution if the filter model is correct. Develop upper and lower limits between which the average $\{\varepsilon_\nu(1)+\ldots+\varepsilon_\nu(50)\}/50$ must lie 99% of the time if the Kalman filter model is correct, and test your averages for each of the three candidate Q values. Which is the most reasonable? Look at the state estimate differences between the best filter and the other two filters. Compute the RMS value of the difference time history for each state vector element. Do the averaging over the last 40 points. Are these differences significant compared to the computed state estimation error standard deviations for the best filter?

6. Develop a truth model simulation of a stochastic linear system, and use it to test the consistency of a Kalman filter by doing Monte Carlo runs. A template for the truth model simulation is contained in the file "kf_truthmodel_template.m", which can be found on the course web site. Several key pieces of it are missing. Each missing part of the code is denoted by 4 question marks, "????". Complete the code to correctly simulate the stochastic linear system that is described in the initial comment statements.

   Use this truth model simulation and a Kalman filter to do a Monte Carlo study of the consistency of the Kalman filter. Test the Kalman filter's consistency on the Kalman filtering problem described in Problem 3. Use Monte Carlo techniques to calculate $E[\underline{\tilde{x}}(10)]$, $E[\underline{\tilde{x}}(10)\underline{\tilde{x}}^T(10)]$, $E[\underline{\tilde{x}}(35)]$, and $E[\underline{\tilde{x}}(35)\underline{\tilde{x}}^T(35)]$. Show that $E[\underline{\tilde{x}}(10)]$ and $E[\underline{\tilde{x}}(35)]$ approach zero as the number of Monte Carlo runs gets to be large. Recall that $\underline{\tilde{x}}(k) = \underline{x}(k) - \underline{\hat{x}}(k)$. Show that $E[\underline{\tilde{x}}(10)\underline{\tilde{x}}^T(10)]$ approaches $P(10)$ and that $E[\underline{\tilde{x}}(35)\underline{\tilde{x}}^T(35)]$ approaches $P(35)$ as the number of Monte Carlo runs gets to be large. Do one Monte Carlo run that uses 50 simulations and do another that uses 1000 simulations and compare your results. As part of the record of your work include your calculated values for $E[\underline{\tilde{x}}(10)]$, $E[\underline{\tilde{x}}(10)\underline{\tilde{x}}^T(10)]$, $E[\underline{\tilde{x}}(35)]$, and $E[\underline{\tilde{x}}(35)\underline{\tilde{x}}^T(35)]$ and any code that you used to perform this study.

   Hints: You will have to use the Cholesky factorization function "chol.m" in order to generate your initial truth state and the truth process and measurement noise vectors. You will also have to use the Gaussian random number generation function, "randn.m", which samples a Gaussian distribution with a mean equal to zero and covariance equal to the identity matrix. This function's outputs are uncorrelated from call to call. Do not forget to use $\underline{\hat{x}}(0)$ in your formula for the truth $\underline{x}(0)$. You can check whether you have used "chol.m" and "randn.m" to correctly define the various error and noise vectors by re-computing their covariances based on your knowledge of how "chol.m" and "randn.m" work.

7. Consider the following nonlinear least-squares estimation problem. You want to observe the sun of a planet on which you find yourself stranded in order to determine your latitude, the latitude of the sub-solar point (the point of intersection between the surface of the planet and the line that connects the center of the planet with the center of its sun), and the local azimuth direction to true north.

You will do this using the following measurements and, of course, a computer. The measurements are made using a simple device that consists of a flat board with x-y graph paper glued to it. At the origin of the x-y axes a small thin post has been erected that stands perpendicular to the x-y plane. It extends to a distance l_p above the x-y plane, and it comes to a point at its top. The board rests on the surface of the planet and has been leveled using a spirit level so that the x-y plane is horizontal. You record the position of the shadow of the tip of the post at various times during a single day. The times are spaced about an hour apart, but you forget to write down these times. Thus, the measurement vector $\underline{z}(j) = [x_p(j);y_p(j)]$, the location of the shadow point in the horizontal plane at sample time j.

You model the relationship between the things that you want to estimate and these measurements. This model assumes that the planet is spherical and that its sun does not move significantly in latitude during a single day. Suppose that the longitude difference between you and the sun's sub-solar point is $\phi_{su}(j)$ at the time of the j^{th} measurement. Suppose θ_s is the planet's sun's latitude and that θ_u is your latitude. Suppose that ψ_u is the azimuth error of your local-level coordinate system. It is defined so that $\psi_u = 0$ when your local level $+y$ axis is aligned with true north. Otherwise, ψ_u is the angle between your $+y$ axis and true north, and ψ_u is positive when true north is east of your $+y$ axis, i.e., when it is rotated towards your $+x$ axis. These quantities can be used to determine the unit direction vector from you to the sun in your local level coordinate system:

$$\begin{bmatrix} \hat{s}_{ux}(j) \\ \hat{s}_{uy}(j) \\ \hat{s}_{uz}(j) \end{bmatrix} = \underline{\hat{s}}_u(j) = \begin{bmatrix} -\sin\psi_u & \cos\psi_u & 0 \\ -\cos\psi_u & -\sin\psi_u & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \sin\theta_u & 0 & -\cos\theta_u \\ 0 & 1 & 0 \\ \cos\theta_u & 0 & \sin\theta_u \end{bmatrix} \begin{bmatrix} \cos\{\phi_{su}(j)\}\cos\theta_s \\ \sin\{\phi_{su}(j)\}\cos\theta_s \\ \sin\theta_s \end{bmatrix}$$

Note that your local level $+z$ axis points towards zenith and that parallax effects have been neglected. This vector can be used to model the location measurement of the shadow of the post's tip:

$$\begin{bmatrix} x_p(j) \\ y_p(j) \end{bmatrix} = \begin{bmatrix} \left(\dfrac{-\hat{s}_{ux}(j) l_p}{\hat{s}_{uz}(j)} \right) \\ \left(\dfrac{-\hat{s}_{uy}(j) l_p}{\hat{s}_{uz}(j)} \right) \end{bmatrix} + \underline{w}(j)$$

In this equation $\underline{w}(j)$ is the measurement noise. It is modeled as a Gaussian with statistics $E[\underline{w}(j)] = 0$ and $E[\underline{w}(j)\underline{w}^T(k)] = \delta_{jk}\, I\sigma_{xy}^2$, where σ_{xy} is the standard deviation of the measurement noise for each axis.

Suppose that you do this experiment and get the following input parameters and data:

l_p = 0.25 m
σ_{xy} = 0.0025 m
$[x_p(1);y_p(1)]$ = [-0.9550; 0.5407]
$[x_p(2);y_p(2)]$ = [-0.6314; 0.4909]
$[x_p(3);y_p(3)]$ = [-0.4526; 0.4914]

$[x_p(4);y_p(4)]$ = [-0.3276; 0.5030]
$[x_p(5);y_p(5)]$ = [-0.2349; 0.5435]
$[x_p(6);y_p(6)]$ = [-0.1415; 0.6044]
$[x_p(7);y_p(7)]$ = [-0.0449; 0.7020]
$[x_p(8);y_p(8)]$ = [+0.0777; 0.8759]

Write MATLAB code to solve this estimation problem using the Gauss-Newton method and solve it. Document your code, your estimate, and your computed estimation error covariance matrix.

Is it possible that you are on the Earth? Explain your reasoning based on your estimate, its covariance, and known facts about the Earth and its Sun.

Help and Hints: It is not obvious how to generate a reasonable first guess for input to the Gauss-Newton method. You may use the function "fstgessunlat.m", which is available on the course web site, to generate a rough first guess for input to your Gauss-Newton code. This first guess routine assumes that the planet's sun's latitude is 0. It fits a straight line to the data in the x-y plane and estimates your latitude and the azimuth of true north based on the line's point of closest approach to the post at the origin of the x-y coordinate system. Next, it computes $\hat{\underline{s}}_u(j)$ from the measurement data by normalizing the vector $[-x_p(j);-y_p(j);\, l_p]$, and it uses this vector in the model of $\hat{\underline{s}}_u(j)$ to compute $\phi_{su}(j)$. Your estimation vector will be $\underline{x} = [\phi_{su}(1);\, \phi_{su}(2);\, \phi_{su}(3);\, \phi_{su}(4);\, \phi_{su}(5);\, \phi_{su}(6);\, \phi_{su}(7);\, \phi_{su}(8);\, \theta_s;\, \theta_u;\, \psi_u]$. It may be convenient to write a function that computes $[x_p(j);y_p(j)] = \underline{h}_{short}[\underline{x}_{short}(j)]$ where $\underline{x}_{short}(j) = [\phi_{su}(j);\, \theta_s;\, \theta_u;\, \psi_u]$ and that computes $H_{short}(j) = \partial\underline{h}_{short}/\partial\underline{x}_{short}$. You can use the 2×4 matrix $H_{short}(j)$ to compute the 2×11 matrix $H(j)$. The j^{th} column of $H(j)$ equals the 1st column of $H_{short}(j)$, and the last three columns of $H(j)$ equal the last 3 columns of $H_{short}(j)$. All of the remaining columns of $H(j)$ are zero. Afterwards, it is necessary to stack the 8 different vectors $\underline{h}_{short}[\underline{x}_{short}(j)]$ for $j = 1, \ldots, 8$ in order to form the $\underline{h}[\underline{x}]$ function of the entire estimation problem and to stack the 8 different matrices $H(j)$ for $j = 1, \ldots, 8$ in order to form the H matrix of the entire estimation problem.

Extra credit option: Do the actual experiment and hand in your data and estimates. This can be done for extra credit regardless of whether this problem gets assigned on an exam. Check your θ_s estimate against the known true value for the day on which you record the data and check your θ_u estimate against the known true latitude of the place where you did the experiment. Also, try to check your ψ_u estimate by leaving the board on the ground until a starry night so that you can see the north star.

Also do the following problems from Bar Shalom:

5-1.1 (Hint: first calculate the steady-state a priori state error covariance), 5-5.1, 5-10, 5-11, 5-12 (Consider observability for part 2. A significant difference between problems 5-11 and 5-12 is that the bias in the measurements is assumed to be known in 5-11, but in 5-12 it is unknown and must be estimated.), and 5-16.