Posting Date: Monday Nov. 18th.

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

1. Consider the nonlinear discrete-time system model.

$$\begin{bmatrix} x_\phi(j+1) \\ x_\omega(j+1) \\ x_\alpha(j+1) \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & \dfrac{\Delta t^2}{2} \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_\phi(j) \\ x_\omega(j) \\ x_\alpha(j) \end{bmatrix} + \underline{v}(j)$$

$$\underline{z}(j) = \begin{bmatrix} A\cos\{x_\phi(j)\} \\ A\sin\{x_\phi(j)\} \end{bmatrix} + \underline{w}(j)$$

where

$$E[\underline{v}(j)] = 0,\; E[\underline{v}(j)\underline{v}^{\mathrm{T}}(k)] = \delta_{jk}\, q_C \begin{bmatrix} \Delta t^5/20 & \Delta t^4/8 & \Delta t^3/6 \\ \Delta t^4/8 & \Delta t^3/3 & \Delta t^2/2 \\ \Delta t^3/6 & \Delta t^2/2 & \Delta t \end{bmatrix}$$

$$E[\underline{w}(j)] = 0,\; E[\underline{w}(j)\underline{w}^{\mathrm{T}}(k)] = \delta_{jk} \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix},\; E[\underline{v}(j)\underline{w}^{\mathrm{T}}(k)] = 0$$

This is a simplified form of a system model that can be used to design a phase-lock loop for a radio receiver by using Kalman filtering techniques. Phase-lock loops are important for data demodulation in communications systems and for the measurement of carrier phase navigation data by a GPS receiver. The problem quantities in the model are defined as follows:

x_ϕ: Carrier phase perturbation from nominal in radians
x_ω: Doppler shift in rad/sec
x_α: Rate of change of Doppler shift in rad/sec^2
Δt: Sample period in sec
A: Signal amplitude in receiver output amplitude units
q_C: Carrier phase jerk intensity in rad^2/sec^5
σ: Measurement noise in receiver output amplitude units

The signal-to-noise ratio of this system is $A^2/(2\sigma^2)$.

If the signal-to-noise ratio is high, then it is possible to use the 2-argument arctangent function to synthesize a phase angle measurement:

$$z_\phi(j) = atan2\left[z_2(j), z_1(j)\right] = x_\phi(j) + 2\pi n(j) + w_\phi(j)$$

where $n(j)$ is an integer that "unwinds" the 2π phase ambiguity of the 2-argument arctangent function and where

$$E[\,w_\phi(j)\,] \;=\; 0,\;\; E[\,w_\phi(j)\; w_\phi(k)\,] \;\cong\; \delta_{jk}\,\frac{\sigma^2}{A^2}$$

If one can work out a way to estimate $n(j)$, then one can implement a linear Kalman filter for the high signal-to-noise ratio case. A method for estimating $n(j)$ that works well is to compute:

$$n(j) \;=\; \text{round}[\frac{z_\phi(j) - \bar{x}_\phi(j)}{2\pi}]$$

where $\bar{x}_\phi(j)$ is the Kalman filter's a priori carrier phase estimate at sample j.

If the signal-to-noise ratio is low, then it is better to work directly with the original measurement equation. One needs an estimate of the amplitude A in order to do this. This estimate can be formed by augmenting the Kalman filter state vector to include amplitude dynamics. A random-walk model will suffice for many applications. The resulting augmented system model takes the form:

$$\begin{bmatrix} x_\phi(j{+}1) \\ x_\omega(j{+}1) \\ x_\alpha(j{+}1) \\ A(j{+}1) \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} & 0 \\ 0 & 1 & \Delta t & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_\phi(j) \\ x_\omega(j) \\ x_\alpha(j) \\ A(j) \end{bmatrix} + \begin{bmatrix} \underline{v}(j) \\ v_A(j) \end{bmatrix}$$

$$\underline{z}(j) \;=\; \begin{bmatrix} A(j)\,cos\{x_\phi(j)\} \\ A(j)\,sin\{x_\phi(j)\} \end{bmatrix} + \underline{w}(j)$$

where

$$E[v_A(j)] \;=\; 0,\;\; E[v_A(j)v_A(k)] \;=\; \delta_{jk}\,q_A\,\Delta t,\;\; E[\underline{v}(j)v_A(k)] \;=\; 0$$

and where q_A is the continuous-time white noise intensity that drives the amplitude's random walk. It is given in receiver output amplitude units squared per second.

<u>Tasks to complete</u>:

a) Develop a linear SRIF for the high signal-to-noise ratio case. Encode it by completing the template file "pll_lkf_temp.m", which is available on the course web site. You will have to replace the sections denoted by ???? with appropriate code.

b) Apply your SRIF from Part a) to the two cases whose data are contained in the files "case01_data.mat" and "case02_data.mat", which are available on the course web site. The first case has a high signal-to-noise ratio, and the second case has a much lower signal-to-noise ratio. The second case also has two different values for $\hat{\underline{x}}(0)$. Try both of them. Compute the means of the innovation statistic time histories, $\varepsilon_\nu(1)$, $\varepsilon_\nu(2)$,

$\varepsilon_V(3)$, … for each case. Compare these to the expected mean for the appropriate Chi-squared distribution. Compute reasonable lower and upper bounds for these means and check whether the actual means respect these bounds. Does the filter seem reasonable for all cases? Are you able to use the innovation statistic's mean to determine the best $\hat{\underline{x}}(0)$ for the data in "case02_data.mat"? Does the estimated state time history vary significantly depending on which initial state estimate you used for the 2nd case?

c) Repeat Part a), except develop an iterated extended SRIF for the 4-state dynamic model that is recommended for use when the signal-to-noise ratio is not very high. Do this by completing the template file "pll_iekf_temp.m", which is available on the course web site. You will have to replace the sections denoted by ???? with appropriate code.

d) Repeat Part b), except use your iterated extended SRIF from Part c) to filter the data. Try the case of one iteration, i.e., a standard extended Kalman filter, and try the case of 4 iterations. Does the iterated extended filter work better than the linear filter for the data in "case02_data.mat", i.e., for the data that correspond to a low signal-to-noise ratio? Does the iterated filter with 4 iterations work better than the normal extended Kalman filter that iterates just once? Answer these questions by considering the means of the innovation statistic time histories.

<u>Hints</u>: Be sure to properly include $n(j)$ in the measurement update of the linear SRIF. In order to compute the $\varepsilon_V(j)$ statistic using SRIF calculations, you might want to refer to your lecture notes, to a previous assignment, or to both. The iterated SRIF measurement update is particularly tricky. You must be careful to take proper account of the state vector value about which you linearize when you compute the updated state information vector. Your measurement update must re-use the a priori square-root information matrix and vector only in the appropriate places. Do not use the first element of the innovation statistic time history, $\varepsilon_V(0)$, in computing the time history's mean because $\varepsilon_V(0)$ will be arbitrarily set to zero due to the fact that there is no measurement update at the filter initialization time.

2. Solve the nonlinear Kalman filtering problem of Problem 1, Part c using a UKF. Compare its results to those of your extended SRIF. In order to complete this part, you will need to develop a sensible $\varepsilon_V(k)$ statistic for your UKF.

3. Solve the particle filtering example problem that was presented in lecture. Use the basic particle filtering method that was presented in lecture, in which $q[\underline{x}(k+1)|\underline{x}(k),\underline{z}(k+1)] = p[\underline{x}(k+1)|\underline{x}(k)]$. Use one of the first two re-sampling schemes that was presented in lecture and use the natural logarithm-based weighting update method that was mentioned in lecture in order to avoid underflow problems. Solve the estimation problem using the measurement data that is given in the file measdata_pfexample.mat, which is posted on the course web site.

4. Re-do Problem 3 using an extended Kalman filter, and compare your results to those of Problem 3.

5. Re-do Problem 3 using a UKF, and compare your results to those of Problems 3 and 4. Use $\alpha = 1$, $\kappa = 1$, and $\beta = 2$ in your UKF.

6. Prove that the UKF approaches the 1st-order EKF in the limit of very small UKF α. Assume for purposes of this proof that one can ignore the effects of numerical round-off error on the accuracy of finite-difference derivatives. Also assume that the point about which the EKF linearizes its dynamics function $\underline{f}[k,\underline{x}(k),\underline{u}(k),\underline{v}(k)]$, the point $\underline{x}(k) = \hat{\underline{x}}(k)$ and $\underline{v}(k) = 0$, is a point where the second derivatives of that function with respect to $\underline{x}(k)$ and $\underline{v}(k)$ are all zero. Similarly, assume that the point about which the EKF linearizes its measurement model function $\underline{h}[k+1,\underline{x}(k+1)]$, the point $\underline{x}(k+1) = \bar{\underline{x}}_{EKF}(k+1) = \underline{f}[k,\hat{\underline{x}}(k),\underline{u}(k),0]$, is a point where the second derivative of that function with respect to $\underline{x}(k+1)$ is zero.

   Hints: You will have to consider 3rd derivative effects on the UKF's $\bar{\underline{x}}(k+1)$ and $\bar{\underline{z}}(k+1)$ calculations in order to complete this proof in a mathematically rigorous manner. You will also need to deal with the fact that all of the UKF weights include terms that scale as $1/\alpha^2$. Therefore, these weight terms go to infinity as α approaches zero.

7. Do fixed-interval particle smoothing on the particle filtering problem of Problem 3, i.e., the one that was presented in lecture. Do this by using the first particle filtering algorithm that was presented in lecture. Recall that this method generates each "particle" as being an open-loop simulation of the entire trajectory of 100 samples that uses a random-number generator in a truth-model-like simulation. The weight of each of these "particles" is proportional to the exponential of negative one half the sum of the weighted squares of the measurement errors, summed over the entire trajectory. How many of these trajectory "particles" are needed in order to get a good smoothed estimate of the state? How many are needed in order for a significant number of them to have non-negligible weights?

8. A UKF should have comparable performance to a KF on a linear problem. Re-do Problem 3 of Assignment 5 using a UKF. Compare your results to the optimal results that you achieved with a Kalman filter.

9. A good particle filter should have comparable performance to a KF on a linear problem. Re-do Problem 3 of Assignment 5 using a particle filter. Use one of the first two re-sampling schemes that was presented in lecture and use the natural logarithm-based weighting update method that was mentioned in lecture in order to avoid underflow problems. Compare your results to the optimal results that you achieved with a Kalman filter. How many particles did you need in order to achieve results that closely approximate those of the KF?

Note that the following useful software is available on the course web site for help with doing Problems 10 and 11:

> epanechnikovsample01.m: Draws samples from the Epanechnikov kernel distribution $K_{opt}(\underline{x})$ that is presented in Eq. (76) of the Feb. 2002 IEEE tutorial paper by Arulampalam, Maskell, Gordon, and Clapp.
>
> unithypervolume01.m: Computes the hypervolume of the unit hypersphere in n_x space, c_{nx} as used in Eqs. (76) and (78) of the Feb. 2002 IEEE tutorial paper by Arulampalam, Maskell, Gordon, and Clapp.
>
> epanechnikovmagsolve01.m: Function needed by epanechnikovsample01.m.

Note, also, that Eq. (77) of the Feb. 2002 IEEE tutorial paper by Arulampalam, Maskell, Gordon, and Clapp appears to have a sign error. The corrected form of the equation, as per Eq. (3.51) of the 2004 book by Ristic, Arulampalam, and Gordon, appears to be:

$$h_{opt} = \frac{A}{N_s^{1/(n_x+4)}}$$

10. Re-do Problem 3 using the regularized particle filter that is denoted as Algorithm 6 in the Feb. 2002 IEEE tutorial paper by Arulampalam, Maskell, Gordon, and Clapp. Are there any noticeable beneficial or adverse effects of the regularized filter's modified re-sampling scheme?

11. Re-do Problem 9 using the regularized particle filter that is denoted as Algorithm 6 in the Feb. 2002 IEEE tutorial paper by Arulampalam, Maskell, Gordon, and Clapp. Are there any noticeable beneficial or adverse effects of the regularized filter's modified re-sampling scheme?

12. Suppose that the first element of a particular system's discrete-time dynamics model function is $f_1(\underline{x}) = a + \underline{b}^{\mathrm{T}}\underline{x} + 0.5\underline{x}^{\mathrm{T}}C\underline{x}$, where a is a known scalar constant, $\underline{b}$ is a known n_x-by-1 vector constant, and C is a known symmetric n_x-by-n_x matrix constant, Suppose, also, that the probability density function for $\underline{x}$ is a Gaussian with mean $\underline{\hat{x}}$ and covariance $P = S_x S_x^{\mathrm{T}}$. It is easy to show that the mean value of $f_1(\underline{x})$ is

$$\bar{f}_1 = E\{f_1(\underline{x})\} = a + \underline{b}^T \underline{\hat{x}} + 0.5\underline{\hat{x}}^T C \underline{\hat{x}} + 0.5\,Trace(CP)$$

    Prove that the UKF mean propagation calculation gives exactly this same value for $\bar{f}_1$ regardless of its choice of α or κ.

    Hint: It will probably be helpful to recall that $Trace(AB) = Trace(BA)$ if both matrix multiplications make sense.

13. Most of the software and an example data set for solving an advanced batch nonlinear least-squares estimation problem for missile tracking is contained in the .zip file missilenl3dpacket.zip. This is a 3-dimensional problem with a single radar that measures the missile's range, elevation, and azimuth. The missile dynamics are modeled as being a ballistic trajectory. Instead of characterizing the missile's motion using a simple flat-Earth approximation, this formulation models the missile's ballistic trajectory over an oblate Earth that has $1/r^2$ and J_2 gravity effects as well as Coriolis and centrifugal effects due to the Earth's rotation. This model necessitates the use of numerical integration in order to develop the relationship between the estimated initial state of the missile at time t_0, which is

$$\underline{x}_0 = \begin{bmatrix} \vec{r}_{ven0} \\ \vec{v}_{ven0} \end{bmatrix}$$

and its state at any given radar measurement time. The vector $\vec{r}_{ven0}$ is the missile's initial Cartesian position vector in the Vertical/East/North local-level coordinate system that is centered at the radar, and $\vec{v}_{ven0} = \dot{\vec{r}}_{ven0}$ is the missile's initial velocity vector measured relative to this coordinate system.

The batch measurement model for this system takes the form:

$$\underline{z}_{hist} = \underline{h}_{batch}(t_0, \underline{x}_0, \underline{t}_{meas}) + \underline{w}_{hist} = \begin{bmatrix} \underline{h}\{t_{m1}, \underline{x}(t_0, \underline{x}_0, t_{m1})\} \\ \underline{h}\{t_{m2}, \underline{x}(t_0, \underline{x}_0, t_{m2})\} \\ \underline{h}\{t_{m3}, \underline{x}(t_0, \underline{x}_0, t_{m3})\} \\ \vdots \\ \underline{h}\{t_{mN}, \underline{x}(t_0, \underline{x}_0, t_{mN})\} \end{bmatrix} + \begin{bmatrix} \underline{w}_1 \\ \underline{w}_2 \\ \underline{w}_3 \\ \vdots \\ \underline{w}_N \end{bmatrix}$$

where $\underline{x}(t_0, \underline{x}_0, t)$ is the solution to the following initial value problem of the state dynamics differentiation equation:

$$\dot{\underline{x}}(t) = \underline{f}[t, \underline{x}(t)]$$

$$\underline{x}(t_0) = \underline{x}_0$$

Note that the dynamics model function $\underline{f}[t, \underline{x}(t)]$ contains the $1/r^2$ and J_2 gravity models and the Coriolis and centrifugal acceleration models.

The function $\underline{h}\{t, \underline{x}(t)\}$ is the 3-by-1 vector model for an individual missile range-elevation-azimuth measurement vector when the missile is at the position and velocity in $\underline{x}(t)$. The vector

$$\underline{t}_{meas} = \begin{bmatrix} t_{m1} \\ t_{m2} \\ t_{m3} \\ \vdots \\ t_{mN} \end{bmatrix}$$

contains the sample times of the N individual radar measurement vectors. The vector

$$\underline{w}_{hist} = \begin{bmatrix} \underline{w}_1 \\ \underline{w}_2 \\ \underline{w}_3 \\ \vdots \\ \underline{w}_N \end{bmatrix}$$

is the measurement noise vector. Its elements are assumed to be samples from zero-mean Gaussian distributions that are uncorrelated with each other. The measurement vector $\underline{z}_{hist}$ contains the stacked time history of individual sets of radar range, elevation, and azimuth measurements at the sample times in $\underline{t}_{meas}$. The vectors $\underline{z}_{hist}$ and $\underline{w}_{hist}$ and the measurement model function $\underline{h}_{batch}(t_0, \underline{x}_0, \underline{t}_{meas})$ all have dimension $(3N) \times 1$.

This is the course's only batch nonlinear least-squares estimation problem where the full nonlinear measurement model function depends on an unknown initial condition vector for which numerical integration is required to determine the state at other times in order to properly model all of the measurements. The challenge of this problem is to develop the batch measurement model function $\underline{h}_{batch}(t_0, \underline{x}_0, \underline{t}_{meas})$ by using the function "c2dnonlinear.m" that you developed in Problem 1 of Assignment 7. You must use this function in order to do the numerical integration of the nonlinear system dynamics model that must occur as part of the calculations of $\underline{h}_{batch}(t_0, \underline{x}_0, \underline{t}_{meas})$. Recall that "c2dnonlinear.m" was originally developed for use in a nonlinear Kalman filter rather than a nonlinear batch filter. Nevertheless, it can be used in a batch filter too.

The assignment tasks are to edit the function template "hbatchmissile3dnl_temp.m" that is found in the file "missilenl3dpacket.zip". Fill in the places where ???? appears in order to develop valid software for computing $\underline{h}_{batch}(t_0, \underline{x}_0, \underline{t}_{meas})$ and its first partial derivative with respect to $\underline{x}_0$. Rename your edited function to be "hbatchmissile3dnl.m". After you have completed this task, use your completed "c2dnonlinear.m" function from Problem 1 of Assignment 7 and the other functions in "missilenl3dpacket.zip" to solve the batch estimation problem whose data are contained in the included data file "gnestmissile3dnl_iodata01.mat". This will require you to run the Gauss-Newton solution function "gnestmissile3dnl.m." You can compare your results to the outputs that have been generated using valid versions of "c2dnonlinear.m" and "hbatchmissile3dnl.m". These results are contained in the data file in the variables named "x0est_out" , "Jopt_out" , "P_out", and "itermflag_out". If you like, you can also compare your solution to the "truth" value that is contained in the variable "x0true". It might be interesting to re-solve the problem using the noiseless measurement data in the vector "zhist_true" instead of using the noisy data in the vector "zhist". What solution would you expect to get in this situation? What is the lowest elevation angle of the radar measurements?

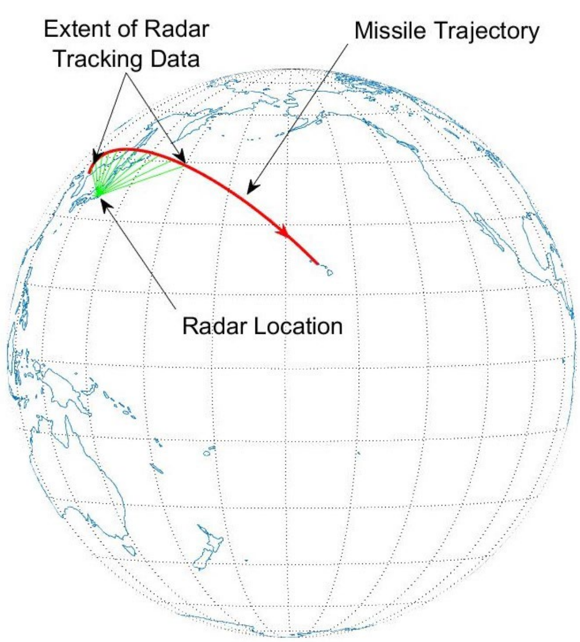

Fig. 1. "Truth" Missile Trajectory and Radar Tracking Geometry for Problem 13.

14. Develop a two-filters version of the smoother. Do this for the original linear smoothing problem

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

$$\underline{z}(k+1) = H(k+1)\underline{x}(k+1) + \underline{w}(k+1)$$

for $k = 0, \ldots, N-1$. Assume that the process noise and measurement noise statistics are

$$E\{\underline{v}(k)\} = 0$$

$$E\{\underline{v}(j)\underline{v}^T(k)\} = \delta_{jk}Q(k)$$

$$E\{w(k)\} = 0$$

$$E\{\underline{w}(j)\underline{w}^T(k)\} = \delta_{jk}I_{n_z \times n_z}$$

$$E\{\underline{w}(j)\underline{v}^T(k)\} = 0$$

Thus, the measurement model has already been transformed to one in which the measurement noise error covariance equals the identity matrix. This will be helpful because your assignment is to use SRIF techniques to develop the two-filters version of the smoother.

Assume that the following initial *a posteriori* square-root information equation is available

$$\underline{\mathfrak{z}}_x(0) = \mathcal{R}_{xx}(0)\underline{x}(0) + \underline{w}_x(0)$$

where $\underline{w}_x(0)$ is Gaussian random noise with a mean of zero and a covariance equal to the identity matrix and where $\underline{w}_x(0)$ is not correlated with any of the measurement noise vectors or process noise vectors.

The strategy for developing the two-filters SRIF smoother is to run a complete SRIF forward filter pass through the entire data interval from $k = 0, \ldots, N$. The results that must be retained from this filter are the $\bar{\underline{\mathfrak{z}}}_x(k)$ vector and the $\bar{\mathcal{R}}_{xx}(k)$ matrix from the *a priori* square-root information equation

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

for all $k = 1, \ldots, N$. A special backward SRIF filter pass will also be developed and implemented. It will produce square-root information equations of the form

$$\underline{\mathfrak{z}}_x^{bf}(k) = \mathcal{R}_{xx}^{bf}(k)\underline{x}(k) + \underline{w}_x^{bf}(k)$$

for $k = 0, \ldots, N$, where this equation contains all of the information about the state conditioned on the measurements from $\underline{z}(k)$ to $\underline{z}(N)$, but not including data from $\underline{z}(0)$ to $\underline{z}(k-1)$. The noise vector $\underline{w}_x^{bf}(k)$ in this equation is also assumed to be Gaussian. It also has a mean of zero and a covariance equal to the identity matrix, and it is uncorrelated with $\underline{\bar{w}}_x(k)$. Thus, the preceding two square-root information equations will together contain all of the information that the smoother needs to use to form the smoothed square-root information equation.

$$\underline{\mathfrak{z}}_x^*(k) = \mathcal{R}_{xx}^*(k)\underline{x}(k) + \underline{w}_x^*(k)$$

Your <u>first task</u> in this problem is to define the calculations that determine $\mathcal{R}_{xx}^*(k)$ from $\bar{\mathcal{R}}_{xx}(k)$ and $\mathcal{R}_{xx}^{bf}(k)$ and that determine $\underline{\mathfrak{z}}_x^*(k)$ from $\underline{\bar{\mathfrak{z}}}_x(k)$ and $\underline{\mathfrak{z}}_x^{bf}(k)$.

<u>Hint</u>: This operation will look somewhat like the measurement update operation of a standard forward pass SRIF.

There will be no need to perform this data fusion operation at sample $k = 0$ because $\mathcal{R}_{xx}^*(0) = \mathcal{R}_{xx}^{bf}(0)$ and $\underline{\mathfrak{z}}_x^*(0) = \underline{\mathfrak{z}}_x^{bf}(0)$ will hold true if the backward SRIF filter pass treats the initial square-root information equation

$$\underline{\mathfrak{z}}_x(0) = \mathcal{R}_{xx}(0)\underline{x}(0) + \underline{w}_x(0)$$

as though it were a measurement equation at sample $k = 0$. That is, as though it were an equation of the form:

$$\underline{z}(0) = H(0)\underline{x}(0) + \underline{w}(0)$$

It must process this "measurement equation" in the same way that processes measurements at other samples. Under this scenario, the sample 0 measurement model uses $\underline{z}(0) = \underline{\mathfrak{z}}_x(0)$, $H(0) = \mathcal{R}_{xx}(0)$, and $\underline{w}(0) = \underline{w}_x(0)$.

You must develop the needed SRIF backward filter. Your backward SRIF will also need a kind of *a priori* square-root information so that it can alternate between dynamic propagation and measurement updates. Its *a priori* square-root information will take the form

$$\underline{\bar{\mathfrak{z}}}_x^{bf}(k) = \bar{\mathcal{R}}_{xx}^{bf}(k)\underline{x}(k) + \underline{\bar{w}}_x^{bf}(k)$$

It will contain the information about $\underline{x}(k)$ that comes from measurements $\underline{z}(k+1)$ to $\underline{z}(N)$.

The backward-pass SRIF filter will be initialized with

$$\underline{\bar{\mathfrak{z}}}_x^{bf}(N) = 0_{n_x \times 1}$$

$$\bar{\mathcal{R}}_{xx}^{bf}(N) = 0_{n_x \times n_x}$$

because there is no information available after $\underline{z}(N)$.

The backward-pass SRIF starts with a measurement update at sample $k = N$ followed by a backward dynamic propagation from sample N to sample $N - 1$, followed by a measurement update at sample $N-1$, followed by a backward dynamic propagation from sample $N-1$ to sample $N-2$, etc.

The measurement update of the backwards SRIF needs to process the equations

$$\underline{\bar{\mathfrak{z}}}_x^{bf}(k) = \bar{\mathcal{R}}_{xx}^{bf}(k)\underline{x}(k) + \underline{\bar{w}}_x^{bf}(k)$$

and

$$\underline{z}(k) = H(k)\underline{x}(k) + \underline{w}(k)$$

to produce the equation

$$\underline{\mathfrak{z}}_x^{bf}(k) = \mathcal{R}_{xx}^{bf}(k)\underline{x}(k) + \underline{w}_x^{bf}(k)$$

Your <u>second task</u> is to develop the needed calculations for this measurement update.

<u>Hint</u>: They will be similar to the measurement update calculations for the standard forward pass SRIF.

Your <u>third task</u> (which is the last) is to develop the dynamic propagation of the backward-pass SRIF. It needs to process the equations

$$0 = \mathcal{R}_{vv}(k)\underline{v}(k) + \underline{w}_v(k)$$

$$\underline{\mathfrak{z}}_x^{bf}(k+1) = \mathcal{R}_{xx}^{bf}(k+1)\underline{x}(k+1) + \underline{w}_x^{bf}(k+1)$$

and

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

in order to produce the equation

$$\underline{\bar{\mathfrak{z}}}_x^{bf}(k) = \bar{\mathcal{R}}_{xx}^{bf}(k)\underline{x}(k) + \underline{\bar{w}}_x^{bf}(k)$$

Develop the needed calculations.

<u>Hints</u>: This can be done in a way that is similar to the dynamic propagation calculations for the standard forward-pass SRIF. The difference is that the regular dynamics model will be used to eliminate $\underline{x}(k+1)$ from its square-root information equation in order to arrive at an equation which involves the unknowns $\underline{x}(k)$ and $\underline{v}(k)$ rather than the unknown $\underline{x}(k+1)$. This is done prior to forming a large block matrix that must get QR-factorized as part of the dynamic propagation. In the forward-pass SRIF, on the other hand, an inverted form of the dynamics model was used to eliminate the unknown $\underline{x}(k)$ from its square-root information equation in order to arrive at an equation which involved the unknowns $\underline{x}(k+1)$ and $\underline{v}(k)$ rather than the unknown $\underline{x}(k)$.