Engineer's Guide to Debugging an Indirect Extended Kalman Filter

Dr. Randall Christensen

September 1, 2020

1 Introduction

This document is an attempt at capturing my methods of debugging an indirect, extended Kalman filter (EKF). My objective is to provide a procedure that systematically verifies correct implementation of each piece of the filter, so that when you glue everything together, there are no surprises. Two overriding principles are "divide and conquer" and "never swing for the fence".

2 Indirect Extended Kalman Filter Equations

The "truth model" of the system is characterized by a typically large vector of numbers called the truth state ($\underline{x}_t$). The truth states evolve according to a continuous nonlinear differential equation and are observed by a discrete measurement

$$\dot{\underline{x}}_t = \underline{f}_t\left(\underline{x}_t, \underline{u}_t\right) + B_t \underline{w}_t \tag{1}$$

$$\tilde{\underline{z}}_t = \underline{h}_t\left(\underline{x}_t\right) + G_t \underline{\nu}_t \tag{2}$$

In the design of a Kalman filter, however, the number of states considered are typically less than truth, and potentially of a different form. Similar to the form of the truth omel, the "design model" for the extended Kalman filter (i.e. the model you base the design on) consists of a nonlinear model of the *continuous* dynamics and *discrete* measurements.

$$\dot{\underline{x}} = \underline{f}\left(\underline{x}, \underline{u}\right) + B \underline{w} \tag{3}$$

$$\tilde{\underline{z}} = \underline{h}\left(\underline{x}\right) + G \underline{\nu} \tag{4}$$

where dependence on time is implied.

In the extended Kalman filter, the nonlinear design models are linearized about the current state estimate to develop linear perturbation models of the state dynamics and measurements

$$\delta\dot{\underline{x}} = F\left(\hat{\underline{x}}\right) \delta\underline{x} + B \underline{w} \tag{5}$$

$$\delta\tilde{\underline{z}} = H\left(\hat{\underline{x}}\right) \delta\underline{x} + G \underline{\nu} \tag{6}$$

where the process noise $\underline{w}$ and the measurement noise $\underline{\nu}$ are zero mean, white noise sources with power spectral density and variance defined by the following

$$E\left[\underline{w}\left(t\right) \underline{w}^T\left(t'\right)\right] = Q\delta\left(t - t'\right) \tag{7}$$

$$E\left[\underline{\nu}_l \underline{\nu}_m^T\right] = R\delta\left[l - m\right] \tag{8}$$

The equations used for propagation of the estimated state $\hat{\underline{x}}$ and the error state covariance P are

$$\dot{\hat{\underline{x}}} = \underline{f}\left(\hat{\underline{x}}, \underline{u}\right) \tag{9}$$

$$\dot{P} = F\left(\hat{\underline{x}}\right) P + PF^T\left(\hat{\underline{x}}\right) + BQB^T \tag{10}$$

The updated error state vector $\delta\underline{x}$ and the associated covariance are

$$\delta\hat{\underline{x}}^+ = K\left[\tilde{\underline{z}}_t - \underline{h}\left(\hat{\underline{x}}^-\right)\right] \tag{11}$$

$$P^+ = \left[I - KH\left(\underline{\hat{x}}^-\right)\right] P^- \left[I - KH\left(\underline{\hat{x}}^-\right)\right]^T + KGRG^T K^T \tag{12}$$

where the Kalman gain is defined as

$$K = P^- H^T\left(\underline{\hat{x}}^-\right)\left[H\left(\underline{\hat{x}}^-\right) P^- H^T\left(\underline{\hat{x}}^-\right) + GRG^T\right]^{-1} \tag{13}$$

In addition to the preceding equations which define the Kalman filter, four useful mappings can be defined between the true navigation state $\underline{x}$, the estimated state $\underline{\hat{x}}$, the truth state $\underline{x}_t$, and the error state $\delta\underline{x}$.

$$\underline{x} = \underline{c}\left(\underline{\hat{x}}, \delta\underline{x}\right) \tag{14}$$

$$\underline{\hat{x}} = \underline{i}\left(\underline{x}, \delta\underline{x}\right) \tag{15}$$

$$\delta\underline{x} = \underline{e}\left(\underline{\hat{x}}, \underline{x}\right) \tag{16}$$

$$\underline{x} = \underline{m}\left(\underline{x}_t\right) \tag{17}$$

Equation 14 provides the fundamental relationship between the true navigation state ($\underline{x}_n$), the navigation errors ($\delta\underline{x}$), and the estimated states ($\underline{\hat{x}}$). It is used in the linearization of the design model when using perturbation techniques to obtain eqns 5 and 6. It is also used to correct the navigation state given the estimated error state $\delta\underline{\hat{x}}^+$. Equation 15 is derived from equation 14 and can be used to insert errors into the state estimates in a way that is consistent with equation 14. Equation 16 is also derived from equation 14 and can be used to calculate the estimation errors consistent with equation 14. Finally, in applications where the truth state vector is different or larger than the navigation states. Equation 17 provides the relationship between the truth state ($\underline{x}_t$) and the corresponding true navigation state.

3 Verification Steps

This section lists several steps which verify separate components of the Kalman filter. It is assumed that a Monte Carlo simulation has been completely implemented (i.e. initialization, propagation, Kalman update, data saving, etc). If you are implementing your EKF while using this debugging guide, you can ignore references to quantities that you have not implemented yet.

3.1 Error State Vector

Equations 14 to 17 are important tools in the following verification steps, and thus need to be verified themselves. For lack of a better method, the following procedure tests that they are consistent with each other, but does not necessarily verify correct definition of errors. For this verification step, do the following:

1. Calculate the navigation state vector without errors using equation 17.
2. Define an error state vector $\delta\underline{x}$.
3. Inject estimation errors into the state estimates using equation 15.
4. Compute the estimation errors using equation 16 and verify that it is equal to the defined error state vector $\delta\underline{x}$.
5. Use the computed estimation errors from the previous step to correct the state estimates using equation 14.
6. Verify that the errors have be removed.

3.2 Nonlinear State Propagation and Nonlinear Measurement Modeling

In the absence of process noise and Kalman updates, the state propagation equation 9 should produce the same results as the truth state propagation equation 1. Furthermore, in the absence of measurement noise ν, the residual $\underline{\tilde{z}} - \underline{h}\left(\underline{\hat{x}}^-\right)$ in equation 11 should be zero. Thus for this verification step, do the following:

1. Set the true process noise $\underline{w}_t$, and the true measurement noise $\underline{\nu}_t$ to zero. Note that the matrices Q and R are not set to zero, because this could cause numerical problems in the Kalman filter.
2. Set the Kalman gain K to zero to prevent state or covariance updates.

3. Set the initial estimation errors to zero.
4. Run a single simulation for a "sufficient" amount of time.
5. Compute estimation errors using equation 16.
6. Plot estimation errors vs. time.
7. Plot residuals vs. time.

This step verifies that the state propagation in equation 1 matches the navigation state propagation in equation 9. This step also verifies that the model used to synthesize measurements $\underline{\tilde{z}}_t$ from truth states in equation 2 is consistent with the model, $\underline{\mathrm{h}}\left(\underline{\hat{\mathrm{x}}}^-\right)$, used to predict the measurements using the navigation states as in equation 11.

3.3 Linear Error State Modeling

The perturbation model in equation 5 is a linear approximation to the difference between the models in equations 3 and 9. Thus, to first order, errors propagating through equations 3 and 9 should approximately agree with errors propagating through equation 5. For the EKF to work properly, the agreement should exist at least over the time duration of a single Kalman cycle. Thus, for this verification step, do the following:

1. Set the true process noise $\underline{\mathrm{w}}_t$, and the measurement noise $\underline{\nu}_t$ to zero. Note that the matrices Q and R are not set to zero, because this could cause numerical problems in the Kalman filter.
2. Set the Kalman gain K to zero to prevent state or covariance updates.
3. Set the time constant of first-order Gauss-Markov (FOGM) processes to 1/10 of the simulation time (for higher order systems, set the bandwidth $= 10/T_{sim}$). This is done so the FOGM process changes significantly during the simulation.
4. At the beginning of the simulation, define an error state vector $\delta\underline{\mathrm{x}}$.
5. Inject estimation errors into the state estimates using equation 15.
6. Over one Kalman cycle, propagate the error state vector using equation 5, the estimated state using equation 9, and the truth state using equation 1.
7. Compute the estimation error at the beginning of the Kalman update using equation 16 and compare with the propagated error state vector. The difference should be "small".

This step verifies that the F matrix used to propagate the covariance in equation 10 is implemented correctly.

3.4 Linear Measurement Modeling

Similar to the previous step, equation 6 is the linear approximation to the residual $\underline{\tilde{z}}_t - \underline{\mathrm{h}}\left(\underline{\hat{\mathrm{x}}}^-\right)$ used in equation 11. Thus error injection can be used to validate the H matrix used in the calculation of the Kalman gain in equation 13. Thus for this verification step, do the following:

1. Set the true process noise $\underline{\mathrm{w}}_t$, and the true measurement noise $\underline{\nu}_t$ to zero. Note that the matrices Q and R are not set to zero, because this could cause numerical problems in the Kalman filter.
2. Set the Kalman gain K to zero to prevent state or covariance updates.
3. Run the simulation to the first Kalman update.
4. Inject estimation errors into the state estimates using equation 15.
5. Compute the residual $\underline{\tilde{z}} - \underline{\mathrm{h}}\left(\underline{\hat{\mathrm{x}}}^-\right)$ and compare it to the linear approximation of the same quantity using equation 6. The difference should be "small".

This step verifies correction implementation of the H matrix.

3.5 Covariance Propagation

So far, you have verified several components of the Kalman filter. This has been done with all noise sources turned off. In this step, you will test the propagation of process noise. For small errors, the propagation of the error covariance using equation 10 should match the ensemble statistics of errors propagated through the nonlinear system in equations 1 and 9. Thus for this verification step, do the following:

1. Set the true measurement noise $\underline{\nu}_t$ to zero. Note that the R matrix is not set to zero, because this could cause numerical problems in the Kalman filter.
2. Set the Kalman gain K to zero to prevent state or covariance updates.
3. Set the time constant of first-order Gauss-Markov (FOGM) processes to 1/10 of the simulation time (for higher order systems, set the bandwidth $= 10/T_{sim}$). This is done so the FOGM process changes significantly during the simulation.
4. Set the initial estimation errors to be consistent with the initial covariance P.
5. Run a Monte Carlo simulation of ˜200 samples for a "sufficient" amount of time.
6. Create a hair plot of the estimation errors for ALL states and verify that the ensemble statistics are consistent with the propagated covariance and zero mean.

This step verifies correct implementation of the covariance propagation in equation 10. It also verifies correct implementation of the process noise coupling matrix B and that you have synthesized process noise $\underline{\text{w}}_t$ consistent with the Q matrix in equation 10.

3.6 Estimation Capability

At this point, you have done enough verification to trust nearly all the terms involved in the Kalman update equations. For a properly-tuned Kalman filter with "small" estimation errors, the error covariance P should accurately represent the ensemble statistics of the state estimation errors (which should be zero mean). In addition, statistics of the residual computed in equation 11 should be zero mean with a covariance of $H\left(\hat{\underline{\text{x}}}^-\right) P^- H^T \left(\hat{\underline{\text{x}}}^-\right) + GRG^T$ from equation 13. Thus for this verification step, do the following:

1. Set the initial estimation errors to be consistent with the initial covariance P.
2. Set all parameters to be representative of your system (e.g. time constants, noise values, etc)
3. Run a Monte Carlo simulation of ˜200 samples for a "sufficient" amount of time and with a state trajectory that is representative of your application.
4. Create a hair plots of the estimation errors for ALL states and verify that the ensemble covariance is consistent with the EKF covariance matrix P.
5. Create residual plots for ALL measurements and verify that the they are zero mean, and have a covariance that is consistent with the residual covariance $H\left(\hat{\underline{\text{x}}}^-\right) P^- H^T \left(\hat{\underline{\text{x}}}^-\right) + GRG^T$.

This steps verifies correct implementation of equations 11 to 13. It also verifies correct implementation of the measurement noise coupling matrix G and that you have synthesized measurement noise $\underline{\nu}_t$ consistent with the R matrix in equations 12 and 13.

4 Additional tips

- Overly-frequent Kalman updates can mask bugs in the propagation of the covariance.
- Over-large covariances may violate the assumptions of linearization.
- Others? Please add your "lessons learned" to this document and pass your findings on to me for future students.