Extended and Unscented Kalman Filter Algorithms for Online State Estimation

You can use discrete-time extended and unscented Kalman filter algorithms for online state estimation of discrete-time nonlinear systems. If you have a system with severe nonlinearities, the unscented Kalman filter algorithm may give better estimation results. You can perform the state estimation in Simulink® and at the command line. To perform the state estimation, you first create the nonlinear state transition function and measurement function for your system.

At the command line, you use the functions to construct the `extendedKalmanFilter` or `unscentedKalmanFilter` object for desired algorithm, and specify whether the process and measurement noise terms in the functions are additive or non-additive. After you create the object, you use the `predict` and `correct` commands to estimate the states using real-time data. For information about the order in which to execute these commands, see the `predict` and `correct` reference pages.

In Simulink, you specify these function in the Extended Kalman Filter and Unscented Kalman Filter blocks. You also specify whether the process and measurement noise terms in the functions are additive or non-additive. In the blocks, the software decides the order in which prediction and correction of state estimates is done.

Extended Kalman Filter Algorithm

The `extendedKalmanFilter` command and Extended Kalman Filter block implement the first-order discrete-time Kalman filter algorithm. Assume that the state transition and measurement equations for a discrete-time nonlinear system have non-additive process and measurement noise terms with zero mean and covariance matrices Q and R, respectively:

$$
\begin{aligned}
x[k+1] &= f(x[k], w[k], u_s[k]) \\
y[k] &= h(x[k], v[k], u_m[k]) \\
w[k] &\sim (0, Q[k]) \\
v[k] &\sim (0, R[k])
\end{aligned}
$$

Here f is a nonlinear state transition function that describes the evolution of states `x` from one time step to the next. The nonlinear measurement function h relates `x` to the measurements `y` at time step `k`. These functions can also have additional input arguments that are denoted by u_s and u_m. The process and measurement noise are `w` and `v`, respectively. You provide Q and R.

In the block, the software decides the order of prediction and correction of state estimates. At the command line, you decide the order. For information about the order in which to execute these commands, see the `predict` and `correct` reference pages. Assuming that you implement the `correct` command before `predict`, the software implements the algorithm as follows:

1. Initialize the filter object with initial values of the state, `x[0]`, and state estimation error covariance matrix, P.

   $$
   \begin{aligned}
   \hat{x}[0|-1] &= E(x[0]) \\
   P[0|-1] &= E(x[0] - \hat{x}[0|-1])(x[0] - \hat{x}[0|-1])^T
   \end{aligned}
   $$

   Here $\hat{x}$ is the state estimate and $\hat{x}[k_a|k_b]$ denotes the state estimate at time step k_a using measurements at time steps `0,1,...,`k_b. So $\hat{x}[0|-1]$ is the best guess of the state value before you make any measurements. You specify this value when you construct the filter.

2. For time steps k = 0,1,2,3,..., perform the following:

   a. Compute the Jacobian of the measurement function, and update the state and state estimation error covariance using the measured data, `y[k]`. At the command line, the `correct` command performs this update.

   $$C[k] = \frac{\partial h}{\partial x}\Big|_{\hat{x}[k \mid k-1]}$$

   $$S[k] = \frac{\partial h}{\partial v}\Big|_{\hat{x}[k \mid k-1]}$$

   The software calculates these Jacobian matrices numerically unless you specify the analytical Jacobian.

   $$K[k] = P[k \mid k-1]C[k]^T(C[k]P[k \mid k-1]C[k]^T + S[k]R[k]S[k]^T)^{-1}$$
   $$\hat{x}[k \mid k] = \hat{x}[k \mid k-1] + K[k](y[k] - h(\hat{x}[k \mid k-1], 0, u_m[k])$$
   $$\hat{P}[k \mid k] = P[k \mid k-1] - K[k]\,C[k]\,P[k \mid k-1]$$

   Here `K` is the Kalman gain.

   b. Compute the Jacobian of the state transition function, and predict the state and state estimation error covariance at the next time step. In the software, the `predict` command performs this prediction.

   $$A[k] = \frac{\partial f}{\partial x}\Big|_{\hat{x}[k \mid k]}$$

   $$G[k] = \frac{\partial f}{\partial w}\Big|_{\hat{x}[k \mid k]}$$

   The software calculates these Jacobian matrices numerically unless you specify the analytical Jacobian. This numerical computation may increase processing time and numerical inaccuracy of the state estimation.

   $$P[k+1 \mid k] = A[k]P[k \mid k]A[k]^T + G[k]Q[k]G[k]^T$$
   $$\hat{x}[k+1 \mid k] = f(\hat{x}[k \mid k], 0, u_s[k])$$

   These values are used by the `correct` command in the next time step.

The Extended Kalman Filter block supports multiple measurement functions. These measurements can have different sample times as long as their sample time is an integer multiple of the state transition sample time. In this case, a separate correction step is performed corresponding to measurements from each measurement function.

The algorithm steps described previously assume that you have non-additive noise terms in the state transition and measurement functions. If you have additive noise terms in the functions, the changes in the algorithm are:

- If the process noise `w` is additive, that is the state transition equation has the form $x[k] = f(x[k-1], u_s[k-1]) + w[k-1]$, then the Jacobian matrix `G[k]` is an identity matrix.
- If the measurement noise `v` is additive, that is the measurement equation has the form $y[k] = h(x[k], u_m[k]) + v[k]$, then the Jacobian matrix `S[k]` is an identity matrix.

Additive noise terms in the state and transition functions reduce the processing time.

The first-order extended Kalman filter uses linear approximations to nonlinear state transition and measurement functions. As a result, the algorithm may not be reliable if the nonlinearities in your system are severe. The unscented Kalman filter algorithm may yield better results in this case.

Unscented Kalman Filter Algorithm

The unscented Kalman filter algorithm and Unscented Kalman Filter block use the unscented transformation to capture the propagation of the statistical properties of state estimates through nonlinear functions. The algorithm first generates a set of state values called sigma points. These sigma points capture the mean and covariance of the state estimates. The algorithm uses each of the sigma points as an input to the state transition and measurement functions to get a new set of transformed state points. The mean and covariance of the transformed points is then used to obtain state estimates and state estimation error covariance.

Assume that the state transition and measurement equations for an M-state discrete-time nonlinear system have additive process and measurement noise terms with zero mean and covariances Q and R, respectively:

$$
\begin{aligned}
x[k+1] &= f(x[k], u_s[k]) + w[k] \\
y[k] &= h(x[k], u_m[k]) + v[k] \\
w[k] &\sim (0, Q[k]) \\
v[k] &\sim (0, R[k])
\end{aligned}
$$

You provide the initial values of Q and R in the `ProcessNoise` and `MeasurementNoise` properties of the unscented Kalman filter object.

In the block, the software decides the order of prediction and correction of state estimates. At the command line, you decide the order. For information about the order in which to execute these commands, see the `predict` and `correct` reference pages. Assuming that you implement the `correct` command before `predict`, the software implements the algorithm as follows:

1. Initialize the filter object with initial values of the state, `x[0]`, and state estimation error covariance, P.

   $$
   \begin{aligned}
   \hat{x}[0 \mid -1] &= E(x[0]) \\
   P[0 \mid -1] &= E(x[0] - \hat{x}[0 \mid -1])(x[0] - \hat{x}[0 \mid -1])^T
   \end{aligned}
   $$

   Here $\hat{x}$ is the state estimate and $\hat{x}[k_a \mid k_b]$ denotes the state estimate at time step k_a using measurements at time steps `0,1,...,`k_b. So $\hat{x}[0 \mid -1]$ is the best guess of the state value before you make any measurements. You specify this value when you construct the filter.

2. For each time step k, update the state and state estimation error covariance using the measured data, `y[k]`. In the software, the `correct` command performs this update.

   a. Choose the sigma points $\hat{x}^{(i)}[k \mid k-1]$ at time step k.

   $$
   \begin{aligned}
   \hat{x}^{(0)}[k \mid k-1] &= \hat{x}[k \mid k-1] \\
   \hat{x}^{(i)}[k \mid k-1] &= \hat{x}[k \mid k-1] + \Delta x^{(i)} \qquad i = 1, \ldots, 2M \\
   \Delta x^{(i)} &= (\sqrt{cP[k \mid k-1]})_i \quad i = 1, \ldots, M \\
   \Delta x^{(M+i)} &= -(\sqrt{cP[k \mid k-1]})_i \quad i = 1, \ldots, M
   \end{aligned}
   $$

   Where $c = \alpha^2(M + \kappa)$ is a scaling factor based on number of states M, and the parameters α and κ. For more information about the parameters, see Effect of Alpha, Beta, and Kappa Parameters. $\sqrt{cP}$ is the matrix square root of cP such that $\sqrt{cP}\ (\sqrt{cP})^T = cP$ and $(\sqrt{cP})_i$ is the ith column of $\sqrt{cP}$.

   b. Use the nonlinear measurement function to compute the predicted measurements for each of the

sigma points.

$$\hat{y}^{(i)}[k \mid k-1] = h(\hat{x}^{(i)}[k \mid k-1], u_m[\mathrm{k}]) \qquad i = 0, 1, ..., 2M$$

c. Combine the predicted measurements to obtain the predicted measurement at time k.

$$\hat{y}[k] = \sum_{i=0}^{2M} W_M^{(i)} \hat{y}^{(i)}[k \mid k-1]$$

$$W_M^{(0)} = 1 - \frac{M}{\alpha^2(M+\kappa)}$$

$$W_M^{i} = \frac{1}{2\alpha^2(M+\kappa)} \qquad i = 1, 2, ..., 2M$$

d. Estimate the covariance of the predicted measurement. Add `R[k]` to account for the additive measurement noise.

$$P_y = \sum_{i=0}^{2M} W_c^{(i)} (\hat{y}^{(i)}[k \mid k-1] - \hat{y}[k])(\hat{y}^{(i)}[k \mid k-1] - \hat{y}[k])^T + R[k]$$

$$W_c^{(0)} = (2 - \alpha^2 + \beta) - \frac{M}{\alpha^2(M+\kappa)}$$

$$W_c^{i} = 1/(2\alpha^2(M+\kappa)) \qquad i = 1, 2, ..., 2M$$

For information about β parameter, see Effect of Alpha, Beta, and Kappa Parameters.

e. Estimate the cross-covariance between $\hat{x}[k \mid k-1]$ and $\hat{y}[k]$.

$$P_{xy} = \frac{1}{2\alpha^2(m+\kappa)} \sum_{i=1}^{2M} (\hat{x}^{(i)}[k \mid k-1] - \hat{x}[k \mid k-1])(\hat{y}^{(i)}[k \mid k-1] - \hat{y}[k])^T$$

The summation starts from i = 1 because $\hat{x}^{(0)}[k \mid k-1] - \hat{x}[k \mid k-1] = 0$.

f. Obtain the estimated state and state estimation error covariance at time step k.

$$K = P_{xy} P_y^{-1}$$

$$\hat{x}[k \mid k] = \hat{x}[k \mid k-1] + K(y[k] - \hat{y}[k])$$

$$P[k \mid k] = P[k \mid k-1] - K P_y K_k^T$$

Here `K` is the Kalman gain.

3. Predict the state and state estimation error covariance at the next time step. In the software, the `predict` command performs this prediction.

a. Choose the sigma points $\hat{x}^{(i)}[k \mid k]$ at time step k.

$$\hat{x}^{(0)}[k \mid k] = \hat{x}[k \mid k]$$

$$\hat{x}^{(i)}[k \mid k] = \hat{x}[k \mid k] + \Delta x^{(i)} \qquad i = 1, ..., 2M$$

$$\Delta x^{(i)} = (\sqrt{cP[k \mid k]})_i \quad i = 1, ..., M$$

$$\Delta x^{(M+i)} = -(\sqrt{cP[k \mid k]})_i \quad i = 1, ..., M$$

b. Use the nonlinear state transition function to compute the predicted states for each of the sigma points.

$$\hat{x}^{(i)}[k+1 \mid k] = f(\hat{x}^{(i)}[k \mid k], u_s[\mathrm{k}])$$

c. Combine the predicted states to obtain the predicted states at time $k+1$. These values are used by the `correct` command in the next time step.

$$\hat{x}[k+1 \mid k] = \sum_{i=0}^{2M} W_M^{(i)} \hat{x}^{(i)}[k+1 \mid k]$$

$$W_M^{(0)} = 1 - \frac{M}{\alpha^2(M+\kappa)}$$

$$W_M^i = \frac{1}{2\alpha^2(M+\kappa)} \quad i = 1, 2, ..., 2M$$

d. Compute the covariance of the predicted state. Add `Q[k]` to account for the additive process noise. These values are used by the `correct` command in the next time step.

$$P[k+1 \mid k] = \sum_{i=0}^{2M} W_c^{(i)} (\hat{x}^{(i)}[k+1 \mid k] - \hat{x}[k+1 \mid k])(\hat{x}^{(i)}[k+1 \mid k] - \hat{x}[k+1 \mid k])^T + Q[k]$$

$$W_c^{(0)} = (2 - \alpha^2 + \beta) - \frac{M}{\alpha^2(M+\kappa)}$$

$$W_c^i = 1/(2\alpha^2(M+\kappa)) \quad i = 1, 2, ..., 2M$$

The Unscented Kalman Filter block supports multiple measurement functions. These measurements can have different sample times as long as their sample time is an integer multiple of the state transition sample time. In this case, a separate correction step is performed corresponding to measurements from each measurement function.

The previous algorithm is implemented assuming additive noise terms in the state transition and measurement equations. If the noise terms are non-additive, the main changes to the algorithm are:

- The `correct` command generates `2*(M+V)+1` sigma points using `P[k|k-1]` and `R[k]`, where `V` is the number of elements in measurement noise `v[k]`. The `R[k]` term is no longer added in the algorithm step 2(d) because the extra sigma points capture the impact of measurement noise on `Py`.
- The `predict` command generates `2*(M+W)+1` sigma points using `P[k|k]` and `Q[k]`, where `W` is the number of elements in process noise `w[k]`. The `Q[k]` term is no longer added in the algorithm step 3(d) because the extra sigma points capture the impact of process noise on `P[k+1|k]`.

Effect of Alpha, Beta, and Kappa Parameters

To compute the state and its statistical properties at the next time step, the unscented Kalman filter algorithm generates a set of state values distributed around the mean state value. The algorithm uses each sigma points as an input to the state transition and measurement functions to get a new set of transformed state points. The mean and covariance of the transformed points is then used to obtain state estimates and state estimation error covariance.

The spread of the sigma points around the mean state value is controlled by two parameters α and κ. A third parameter, β, impacts the weights of the transformed points during state and measurement covariance calculations.

- α — Determines the spread of the sigma points around the mean state value. It is usually a small positive value. The spread of sigma points is proportional to α. Smaller values correspond to sigma points closer to the mean state.
- κ — A second scaling parameter that is usually set to 0. Smaller values correspond to sigma points closer to the mean state. The spread is proportional to the square-root of κ.
- β — Incorporates prior knowledge of the distribution of the state. For Gaussian distributions, β = 2 is optimal.

You specify these parameters in the `Alpha`, `Kappa`, and `Beta` properties of the unscented Kalman filter. If you know the distribution of state and state covariance, you can adjust these parameters to capture the transformation of higher-order moments of the distribution. The algorithm can track only a single peak in the probability distribution of the state. If there are multiple peaks in the state distribution of your system, you can adjust these parameters so that the sigma points stay around a single peak. For example, choose a small `Alpha` to generate sigma points close to the mean state value.

References

[1] Simon, D. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. John Wiley and Sons Inc., 2006.

See Also

Functions

`extendedKalmanFilter` | `unscentedKalmanFilter`

Blocks

Extended Kalman Filter | Unscented Kalman Filter