Kalman filter - for nonlinear systems

CA9/CA3 course Nonlinear Control Systems

Torben Knudsen

Automation and Control
Department of Electronic Systems
Aalborg University
Denmark

Outline

State estimation problem for nonlinear (NL) systems

Extended Kalman filter (EKF) for the DD problem

The unscented transform (UT)

Unscented Kalman filter (UKF) for the NL DD problem

State estimation problem for nonlinear (NL) systems

- Many real systems are more or less nonlinear.
- Is it possible to estimate states in NL systems?
- Does *optimal* estimators exist?
- Does *good* estimators exist?

The discrete-discrete (DD) NL estimation problem

Assume the system:

$$x_{k+1} = f(x_k, u_k) + w_k \ , \ w_k \in \mathrm{NID}(\underline{0}, Q_k) \ ,$$
$$y_k = h(x_k, u_k) + v_k \ , \ v_k \in \mathrm{NID}(\underline{0}, R_k) \ ,$$
$$\mathrm{E}(w(k)v(l)^{\mathtt{T}}) = \underline{\underline{0}}$$

Given measurements and initial values:

$$\mathcal{Y}_k \triangleq y_0, y_1, y_2 \ldots y_k \ , \ u_0, u_1, u_2 \ldots u_k$$
$$x_0 \in \mathrm{N}(\hat{x}_0, P_0)$$

Find: $\hat{x}_k$ which minimized the mean square error

$$\mathrm{E}((x_k - \hat{x}_k)^{\mathtt{T}} M (x_k - \hat{x}_k)) \ , \ M > 0$$

The continuous-discrete (CD) NL estimation problem

The state model is in continuous time i.e. a stochastic differential equation (SDE):

$$
\begin{aligned}
dx(t) = f(x(t), u(t))dt + d\omega(t) \ , \\
\omega(t) \in \mathrm{W}(Q(t)) \ , \\
y(t_k) = h(x(t_k), u(t_k)) + v_k \ , \ v_k \in \mathrm{NID}(\underline{0}, R_k)
\end{aligned}
$$

Given measurements and initial values:

$$
\begin{aligned}
\mathcal{Y}_k \triangleq y(t_0), y(t_1), y(t_2) \dots y(t_k) \ , \ u(s), t_0 \le s \le t_k \\
x(t_0) \in \mathrm{N}(\hat{x}(t_0), P(t_0))
\end{aligned}
$$

Find: $\hat{x}(t)$ which minimized the mean square error

$$
\mathrm{E}((x(t) - \hat{x}(t))^{\mathsf{T}} M (x(t) - \hat{x}(t))) \ , \ M > 0
$$

Extended Kalman filter (EKF)
Basic idea

- The optimal estimator $\mathrm{E}(x(_k)|\mathcal{Y}_k)$ can not normally be obtained.
- The KF can be used for a linearization of the NL system.
- An obvious idea is to use the derivation for the KF and then use approximations and linearization but only where necessary.

Linearization of f and h with respect to x is needed.

$$F_k(x) \triangleq \frac{\partial f(x)}{\partial x^{\mathtt{T}}} \,,$$
$$H_k(x) \triangleq \frac{\partial h(x)}{\partial x^{\mathtt{T}}} \,.$$

The argument x to F_k and H_k should be the most recent estimate of x e.g. $\hat{x}_{k|k-1}$ or $\hat{x}_{k|k}$.
Iterations in the below EKF algorithm is also possible in order to use “even more resent” estimates of x in all linearization.

Extended Kalman filter algorithm for nonlinear DD problem

Initial conditions:

$$x_0 \in \mathrm{N}(\hat{x}_{0|-1}, P_{0|-1})$$

Measurements update after receiving y_k and u_k:

$$\begin{aligned}
\hat{y}_{k|k-1} &= h(\hat{x}_{k|k-1}, u_k) \ , \\
\tilde{y}_{k|k-1} &= y_k - \hat{y}_{k|k-1} \ , \\
K_k &\triangleq P_{k|k-1} H_k^\mathsf{T} (H_k P_{k|k-1} H_k^\mathsf{T} + R_k)^{-1} \ , \\
\hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k \tilde{y}_{k|k-1} \ , \\
P_{k|k} &= (I - K_k H_k) P_{k|k-1} (I - K_k H_k)^\mathsf{T} + K_k R_k K_k^\mathsf{T} \ .
\end{aligned}$$

Time update from k to $k+1$:

$$\begin{aligned}
\hat{x}_{k+1|k} &= f(\hat{x}_{k|k}, u_k) \ , \\
P_{k+1|k} &= F_k P_{k|k} F_k^\mathsf{T} + Q_k \ .
\end{aligned}$$

Important difficulties with the EKF

- The EKF is not optimal!
- The EKF is not guarantied to be stable!

The unscented transform (UT)
Motivation

- In the derivation of the Kalman gain in the measurement update the below covariances was needed.

$$\begin{aligned} K_k &= \mathrm{Cov}(x_k, \tilde{y}_{k|k-1})\,\mathrm{Cov}(\tilde{y}_{k|k-1})^{-1} \\ &= \mathrm{Cov}(\tilde{x}_{k|k-1}, \tilde{y}_{k|k-1})\,\mathrm{Cov}(\tilde{y}_{k|k-1})^{-1} \end{aligned}$$

and also the update of output prediction was needed

$$\hat{y}_{k|k-1} = h(\hat{x}_{k|k-1}, u_k) = \mathrm{E}\left(h(x_k, u_k) + v_k | \mathcal{Y}_{k-1}\right)$$

- In the time update the conditional mean value and covariance below was needed.

$$\hat{x}_{k+1|k} = f(\hat{x}_{k|k}, u_k) = \mathrm{E}\left(f(x_k, u_k) + w_k | \mathcal{Y}_k\right),$$

$$P_{k+1|k} = F_k P_{k|k} F_k^{\mathrm{T}} + Q_k = \mathrm{Cov}\left(f(x_k, u_k) + w_k | \mathcal{Y}_k\right).$$

Can this be approximated more precisely? Using the read equations which really is what we want?

The generic probabilistic problem

Calculate/approximate the second order statistics of x, y given the second order statistics of x and a relation f:

$$y = f(x)\ ,\ \mu_x = \mathrm{E}(x)\ ,\ C_x = \mathrm{Cov}(x)$$

For non-liner relations there is no exact solution in general.

The linearization solution

The EKF can be interpreted as using the linearization approach:

$$y = f(\mu_x) + \nabla f(\mu_x)(x - \mu_x) \Rightarrow$$
$$\mu_y = f(\mu_x)\ ,\ C_{yx} = \nabla f(\mu_x) C_x\ ,$$
$$C_y = \nabla f(\mu_x) C_x \nabla f(\mu_x)^{\mathsf{T}}\ .$$

- A general approach to approximation in probability problems is *Monte Carlo (MC) simulations*.
- The UT method is similar to MC except the realizations of the random variable are not "random" but constructed and the number used is very limited compared to MC.

Algorithm

$$\alpha = 1\ ,\ \kappa = 2\ ,\ \beta = 0\ ,\ \text{(Default parameters gives } \lambda = 2)$$

$$\lambda = \alpha^2(n+\kappa) - n\ ,\ k = \sqrt{n+\lambda}\ ,\ n = \dim(x)\ ,$$

$$u_i \triangleq \text{eigenvector } i \text{ for } C_x\ ,\ l_i \triangleq \text{eigenvalue } i \text{ for } C_x\ ,$$

$$x_i = \begin{cases} \mu_x & ,\ i = 0 \\ \mu_x + k u_i \sqrt{l_i} & ,\ 1 \le i \le n \\ \mu_x - k u_i \sqrt{l_i} & ,\ n+1 \le i \le 2n \end{cases}$$

$$w_i^t = \begin{cases} \frac{\lambda}{n+\lambda} & ,\ i = 0\ ,\ t = m \text{ (for mean)} \\ \frac{\lambda}{n+\lambda} + 1 - \alpha^2 + \beta & ,\ i = 0\ ,\ t = c \text{ (for covarians)} \\ \frac{1}{2(n+\lambda)} & ,\ 1 \le i \le 2n \end{cases}$$

$$\hat{\mu}_y = \sum_{i=0}^{2n} w_i^m f(x_i)\ ,\ \widehat{C}_{yx} = \sum_{i=0}^{2n} w_i^c (f(x_i) - \hat{\mu}_y))(x_i - \hat{\mu}_x))^{\mathtt{T}}\ ,$$

$$\widehat{C}_y = \sum_{i=0}^{2n} w_i^c (f(x_i) - \hat{\mu}_y))(f(x_i) - \hat{\mu}_y))^{\mathtt{T}}\ .$$

Important knowledge about the UT

- There are many alternative ways to do this. They are some times called *sampling* methods.
- The UT gives exact results if f is affine $f(x) = Ax + b$ which it also should.
- For NL f it is still an approximation but often gives good result.
- Especially it is superior to the linearization method and the MC method.

Example

In order to have a test where the correct result is known $y = x^{\mathsf{T}}x$ is used with x Gaussian and $\mathrm{E}(x) = [1; 1]$ and $\mathrm{Cov}(x) = [1\ 1; 1\ 2]$ (using Matlab notation).

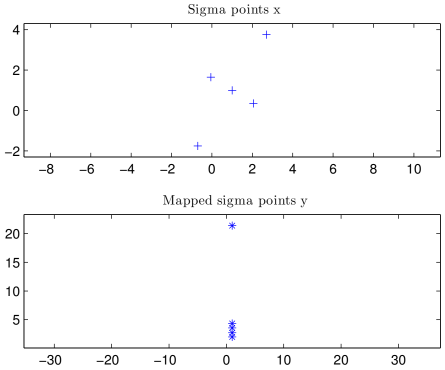

Figure: Sigma points and images for the UT.

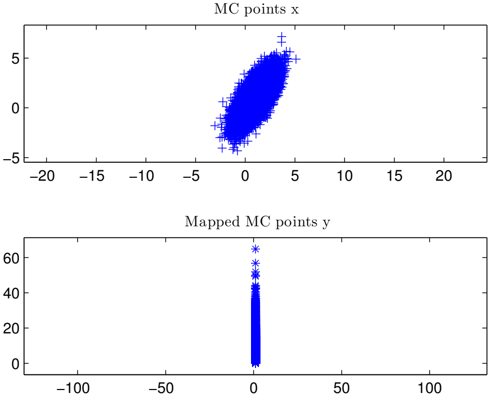

Figure: Random (sigma) points and images for the MC based method.

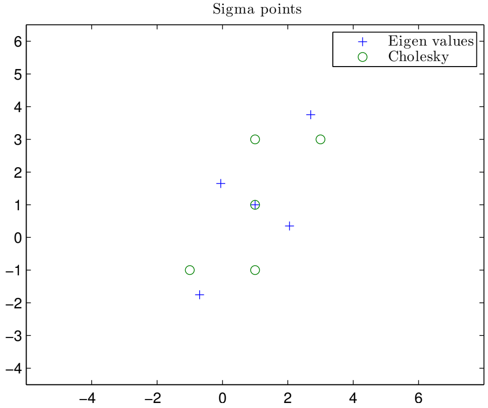

Figure: Comparision of sigma points based on eigenvalues and the Cholesky method.

Method	My	Cy	Cyx	
UTSCEst	5.000e+0	3.900e+1	4.000e+0	6.000e+0
UTJMEst	5.000e+0	3.100e+1	4.000e+0	6.000e+0
UTMCEst	5.038e+0	3.385e+1	3.988e+0	6.028e+0
THTK	5.000e+0	3.400e+1	4.000e+0	6.000e+0
UTSCRelE	0	1.471e-1	0	0
UTJMRelE	0	-8.824e-2	0	0
UTMCRelE	7.663e-3	-4.284e-3	-3.090e-3	4.670e-3

Table: Performance for the above method, the Cholesky based, the MC and the theoretical. The rows marked RelE are the errors relative to the correct results.

Assume the NL map f with input x and output y, the UT can then be represented by the function U below:

$$\begin{bmatrix} \mathrm{E}(y) \\ \mathrm{Cov}(y, x) \\ \mathrm{Cov}(y) \end{bmatrix} = U(f, \mathrm{E}(x), \mathrm{Cov}(x))$$

If the mean and covariance input is conditioned on say z then the output is also conditioned on z i.e.:

$$\begin{bmatrix} \mathrm{E}(y|z) \\ \mathrm{Cov}(y, x|z) \\ \mathrm{Cov}(y|z) \end{bmatrix} = U(f, \mathrm{E}(x|z), \mathrm{Cov}(x|z))$$

Algorithm (UKF)

Initial conditions:

$$x_0 \in \mathrm{N}(\hat{x}_{0|-1}, P_{0|-1})$$

Measurements update after receiving y_k and u_k:

$$\begin{bmatrix} \hat{y}_{k|k-1} \\ \mathrm{Cov}(\tilde{y}_{k|k-1}, \tilde{x}_{k|k-1}) \\ C_k^h \end{bmatrix} = U(h, \hat{x}_{k|k-1}, P_{k|k-1}) \ ,$$

$$\mathrm{Cov}(\tilde{y}_{k|k-1}) = C_k^h + R_k \ ,$$

$$\mathrm{Cov}(\tilde{x}_{k|k-1}, \tilde{y}_{k|k-1}) = \mathrm{Cov}(\tilde{y}_{k|k-1}, \tilde{x}_{k|k-1})^{\mathtt{T}} \ ,$$

$$\tilde{y}_{k|k-1} = y_k - \hat{y}_{k|k-1} \ ,$$

$$K_k = \mathrm{Cov}(\tilde{x}_{k|k-1}, \tilde{y}_{k|k-1}) \, \mathrm{Cov}(\tilde{y}_{k|k-1})^{-1}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \tilde{y}_{k|k-1} \ ,$$

$$P_{k|k} = P_{k|k-1} - K_k \, \mathrm{Cov}(\tilde{y}_{k|k-1}, \tilde{x}_{k|k-1}) \ .$$

Time update from k to $k+1$:

$$\begin{bmatrix} \hat{x}_{k+1|k} \\ X \\ C_k^f \end{bmatrix} = U(f, \hat{x}_{k|k}, P_{k|k}) \; ,$$

$$P_{k+1|k} = \mathrm{Cov}(\tilde{x}_{k+1|k}) = C_k^f + Q_k \; .$$