Lesson 3 : DC Motor Lamped Parameters Identification

1 Background

DC Motor is widely used in many applications such as robot, industrial application -etc. It has been produced in great number, some is at high standard with larger documentation and specification while some are inexpensive with little to none of documentation. To be able to use the dc motor efficiently, we must know it mathematical model. In this lesson, we use a variant of a famous algorithm known as Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF) to estimate the dc motor model.

2 DC Motor Stochastic State Space Model

From Lecture 1 : DC Motor, We have a mathematical model to represent the motor:
Model with neglect the coulomb friction:

$$\boxed{\dot{\omega}(t) = -a\omega(t) + bv_a(t)}$$

Model with the coulomb friction:

$$\boxed{\dot{\omega}(t) = -a\omega(t) + bv_a(t) - csign(\omega(t))}$$

2.1 Model with neglect the coulomb friction

From the model:

$$\dot{\omega}(t) = -a\omega(t) + bv_a(t) \tag{1}$$

In control system, we have a state space model for a nonlinear model:

$$\dot{x}(t) = f(t, x(t), u(t)) + v_{noise}(t)$$

$$y(t) = h(t, x(t), u(t)) + \omega_{noise}(t)$$

Where:

- $\dot{x}(t)$ is rate of change of state
- $x(t)$ is the current state
- $u(t)$ is the input to the system
- $y(t)$ is the measurement model
- $v_{noise}(t)$ is random process noise
- $\omega_{noise}(t)$ is random measurement noise

From Equation 1, Let:

$$\begin{aligned} x_1 &= \omega \\ x_2 &= a \\ x_3 &= b \end{aligned} \tag{2}$$

We get:

$$\begin{aligned} \dot{x_1} = \dot{\omega} &= -a\omega(t) + bv_a(t) = -x_2x_1 + x_3v_a(t) \\ \dot{x_2} &= 0 \\ \dot{x_3} &= 0 \end{aligned} \tag{3}$$

In continuous nonlinear stochastic system matrix form:

$$\boxed{\begin{aligned} \dot{x}(t) = \begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \end{bmatrix}(t) &= \begin{bmatrix} -x_2x_1 + x_3v_a(t) \\ 0 \\ 0 \end{bmatrix} + \sqrt{Q_c}v_{noise}(t) \\ y(t) &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}(t) + \sqrt{R}\omega_{noise}(t) \end{aligned}} \tag{4}$$

Discretize the continuous model from Equation 4:

$$\begin{aligned} \dot{x}(t) &= \begin{bmatrix} -x_2x_1 + x_3v_a(t) \\ 0 \\ 0 \end{bmatrix} + \sqrt{Q_c}v_{noise}(t) \\ y(t) &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}(t) + \sqrt{R}\omega_{noise}(t) \end{aligned} \tag{5}$$

$$\begin{aligned} \frac{x_{k+1} - x_k}{T_s} &= \begin{bmatrix} -x_2x_1 + x_3v_a(t) \\ 0 \\ 0 \end{bmatrix} + \sqrt{Q_c}v_{noise}(t) \\ y_k &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_k + \sqrt{R}\omega_{noise}(t) \end{aligned} \tag{6}$$

We get a discretized nonlinear stochastic system in matrix form:

$$\boxed{\begin{aligned} x_{k+1} &= x_k + T_s \begin{bmatrix} -x_2x_1 + x_3v_{a,k} \\ 0 \\ 0 \end{bmatrix} + \sqrt{T_sQ_d}v_{noise,k} \\ y_k &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_k + \sqrt{R}\omega_{noise,k} \end{aligned}} \tag{7}$$

2.2 Model the coulomb friction

From the model:

$$\dot{\omega}(t) = -a\omega(t) + bv_a(t) - csign(\omega(t)) \tag{8}$$

From Equation 8, Let:

$$\begin{aligned} x_1 &= \omega \\ x_2 &= a \\ x_3 &= b \\ x_4 &= c \end{aligned} \tag{9}$$

We get:

$$\begin{aligned} \dot{x_1} = \dot{\omega} = -a\omega(t) + bv_a(t) &= -x_2x_1 + x_3v_a(t) - x_4sign(x_1) \\ \dot{x_2} &= 0 \\ \dot{x_3} &= 0 \\ \dot{x_4} &= 0 \end{aligned} \tag{10}$$

In continuous nonlinear stochastic system matrix form:

$$\boxed{\begin{aligned} \dot{x}(t) = \begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \\ \dot{x_4} \end{bmatrix}(t) &= \begin{bmatrix} -x_2x_1 + x_3v_a(t) - x_4sign(x_1) \\ 0 \\ 0 \\ 0 \end{bmatrix} + \sqrt{Q_c}v_{noise}(t) \\ y(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} &\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}(t) + \sqrt{R}\omega_{noise}(t) \end{aligned}} \tag{11}$$

We get a discretized nonlinear stochastic system in matrix form:

$$\boxed{\begin{aligned} x_{k+1} = x_k + T_s &\begin{bmatrix} -x_2x_1 + x_3v_{a,k} - x_4sign(x_1) \\ 0 \\ 0 \\ 0 \end{bmatrix} + \sqrt{T_sQ_d}v_{noise,k} \\ y_k = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} &\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}_k + \sqrt{R}\omega_{noise,k} \end{aligned}} \tag{12}$$

3 Identification using Extended Kalman Filter (EKF)

We have an EKF step below:
Initialize:
Select any

- $\hat{x}_{0|0}$ initial state estimate
- $P_{0|0}$ positive definite error covariance matrix

Time Update

$$\begin{aligned} \hat{x}_{k+1|k} &= f_d(\hat{x}_{k|k}, u_k) \\ P_{k+1|k} &= A_k P_{k|k} A_k^T + Q \end{aligned} \tag{13}$$

Measurement Update

$$\begin{aligned} \hat{y}_{k+1|k} &= h_d(\hat{x}_{k+1|k}, u_{k+1}) \\ P_{xz,k+1|k} &= P_{k+1|k} C_{k+1}^T \\ P_{zz,k+1|k} &= C_{k+1} P_{k+1|k} C_{k+1}^T + R \\ \hat{x}_{k+1|k+1} &= \hat{x}_{k+1|k} + P_{xz,k+1|k} P_{zz,k+1|k}^{-1} (y_{k+1} - \hat{y}_{k+1|k}) \\ P_{k+1|k+1} &= P_{k+1|k} - P_{xz,k+1|k} P_{zz,k|k+1}^{-1} P_{xz,k+1|k}^T \end{aligned} \tag{14}$$

3.1 Model with neglect the coulomb friction

From Equation 7, We have:

$$x_{k+1} = x_k + T_s \begin{bmatrix} -x_2 x_1 + x_3 v_{a,k} \\ 0 \\ 0 \end{bmatrix} + \sqrt{T_s Q_d} v_{noise,k}$$

$$y_k = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_k + \sqrt{R} \omega_{noise,k}$$

Applying Extended Kalman Filter

1. **Initialize a state and positive definite error covariance matrix**

$$\hat{x}_{0|0} = \begin{bmatrix} 2 \\ 13 \\ 25 \end{bmatrix} \text{ or some number randomly}$$

$$P_{0|0} = 2 * eye(3) = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix} \text{ or some number randomly}$$

2. **Compute state and covariance matrix in Time Update**
Initialize

- $T_s = 0.01$ sampling time (s) , up to user
- $Q = 0.01 \times diag(3)$ covariance, smaller is truth in process model (use for tuning)

- $R = 0.02$ covariance, smaller is truth in measurement (use for tuning)

Compute

$$\hat{x}_{k+1|k} = \hat{x}_k + T_s \begin{bmatrix} -x_2 x_1 + x_3 v_{a,k} \\ 0 \\ 0 \end{bmatrix} + \boxed{\sqrt{T_s Q_d} v_{noise,k}}$$

$\boxed{\sqrt{T_s Q_d} v_{noise,k}}$ $\leftarrow$ put this bunch if we use in simulation to simulate noise to a true system, don't put if taking real value from system because the system has noise already.

$$v_{noise,k} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Compute

$$P_{k+1|k} = A_k P_{k|k} A_k^T + Q$$

where:

$$A_k = \begin{bmatrix} -x_2 & -x_1 & v_a \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

is the jacobian matrix calculated by derivative the state model. From Equation 4

$$J_f(x,y) = \begin{bmatrix} \frac{df_1}{dx_1} & \frac{df_1}{dx_2} & \frac{df_1}{dx_3} \\ \frac{df_2}{dx_1} & \frac{df_2}{dx_2} & \frac{df_2}{dx_3} \\ \frac{df_3}{dx_1} & \frac{df_3}{dx_2} & \frac{df_3}{dx_3} \end{bmatrix} = \begin{bmatrix} \frac{-x_2 x_1 + x_3 v_a}{dx_1} & \frac{-x_2 x_1 + x_3 v_a}{dx_2} & \frac{-x_2 x_1 + x_3 v_a}{dx_3} \\ \frac{0}{dx_1} & \frac{0}{dx_2} & \frac{0}{dx_3} \\ \frac{0}{dx_1} & \frac{0}{dx_2} & \frac{0}{dx_3} \end{bmatrix}$$

3. **Compute state and covariance matrix in Measurement Update**
Compute $\hat{y}_{k+1|k} = h_d(\hat{x}_{k+1|k}, u_{k+1})$ measurement estimation. This equation is the estimation of a measurement would look like. In our case, we measure the ω directly (Equation 4) and thus we can take:

$$\hat{y}_{k+1|k} = [1 \quad 0 \quad 0]\hat{x}_{k+1|k} + D_k u_{a,k}$$

Where: $D_k = 0$
Compute

$$P_{xz,k+1|k} = P_{k+1|k} C_{k+1}^T$$
$$P_{zz,k+1|k} = C_{k+1} P_{k+1|k} C_{k+1}^T + R$$

From Equation 4, we have $C = [1 \quad 0 \quad 0]$
Compute

$$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + P_{xz,k+1|k} P_{zz,k+1|k}^{-1} (y_{k+1} - \hat{y}_{k+1|k})$$

y_{k+1} is the measurement from sensor which has noise mixed inside. Get data directly from the sensor.
Compute

$$P_{k+1|k+1} = P_{k+1|k} - P_{xz,k+1|k} P_{zz,k|k+1}^{-1} P_{xz,k+1|k}^T$$

```
function x_est = EKF(uk,y_true)
Ts=0.01;
persistent x_est_p P Qd_est R_est Qc_est;
if isempty(x_est_p)
x_est_p = [2;
13;
25;
1];

P = 2*[1 0 0 0;
0 1 0 0;
0 0 1 0;
0 0 0 1;]              %2*eye(4);

Qc_est = 1e-5*[10 0  0 0;
0 25  0 0;
0  0 25 0;
0  0  0 1;]         %1e-5*diag([10,25,25,1]);

Qd_est=Qc_est*Ts;

R_est=0.02;
end

c=[1 0 0 0];
D=0;
Ck=c;
Dk=D;


%Comput Kalman Gain and update predicted value
Wk=P*Ck'/(Ck*P*Ck'+R_est);

y_est=Ck*x_est_p+Dk*uk;

x=x_est_p+Wk.*(y_true-y_est);

%Compute prediction at next time step
x_est=x+Ts*[-x(2)*x(1)+uk*x(3)-x(4)*sign(x(1));
0          ;
0          ;
0         ];


%Update error covariance matrices
P=P-Wk*Ck*P;


%Define Ak
Ak=eye(4)+Ts*[-x(2) -x(1) uk -sign(x(1));
0   0  0   0   ;
0   0  0   0   ;
0   0  0   0   ]; % jacobian

P=Ak*P*Ak'+Qd_est;

x_est_p=x_est;
```