# LAB-5 EXTENDED KALMAN FILTER

Madhumita Krishnan

10/23/2018

## 1 Introduction

The report deals with the design and implementation of Extended Kalman filter to track a sinusoidal model.

The Extended Kalman filter is one of the most extensively used filters for solving non-linear problems.It is used in real-time applications.It solves the non-linear estimation problem by linearizing state and measurement equations and applying the standard Kalman filter formula to the resulting linear estimation problem.

The report begins with developing the Extended Kalman filter equations. The filter results for three different values of dynamic noise and measurement noise are shown and the results are discussed.

## 2 Method

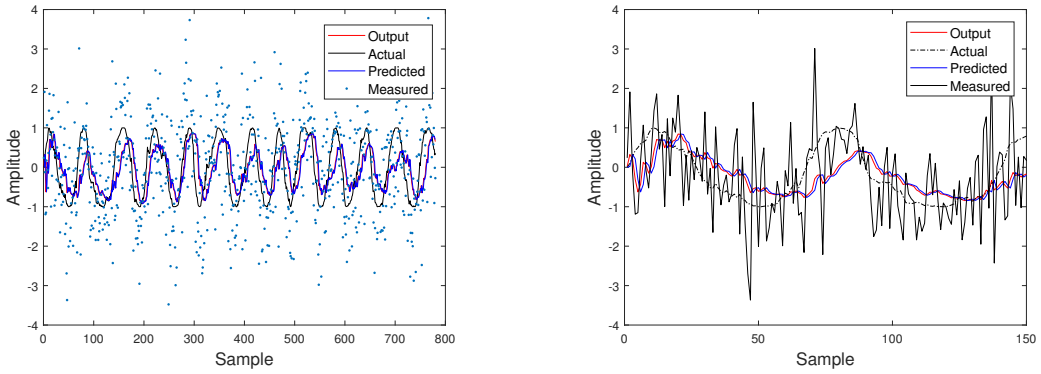
The Extended Kalman filter is a continuous cycle of predict-update. Following are the equations for the process

1. Predict next state
$$X_{t,t-1} = f(X_{t-1,t-1}, 0) \tag{1}$$
where $f(X_{t-1,t-1}, 0)$ is the approximate state of $\tilde{x_t}$.

2. Predict next state covariance
$$S_{t,t-1} = \left(\frac{\partial f}{\partial x}\right) S_{t-1,t-1} \left(\frac{\partial f}{\partial x}\right)^T + \left(\frac{\partial f}{\partial a}\right) Q \left(\frac{\partial f}{\partial a}\right)^T \tag{2}$$
where $\left(\frac{\partial f}{\partial x}\right)$ and $\left(\frac{\partial f}{\partial a}\right)$ are the Jacobians of the state transition equations.

3. Kalman gain $K_t$
$$K_t = S_{t,t-1} \left(\frac{\partial g}{\partial x}\right)^T \left[\left(\frac{\partial g}{\partial x}\right) S_{t,t-1} \left(\frac{\partial g}{\partial x}\right)^T + \left(\frac{\partial g}{\partial n}\right) R \left(\frac{\partial g}{\partial n}\right)^T\right]^{-1} \tag{3}$$
where $\left(\frac{\partial g}{\partial x}\right)$ and $\left(\frac{\partial g}{\partial n}\right)$ are the Jacobians of the measurement equations.

4. Update state
$$X_{t,t} = X_{t,t-1} + K_t[Y_t - g(\tilde{x}_t, 0)] \tag{4}$$
where $g(\tilde{x}_t, 0)$ is the ideal noiseless measurement of the approximate state.




(a) Complete plot of the tracking

(b) A closer image of the same

Figure 1: 1D Tracking of EKF

5. Update state covariance

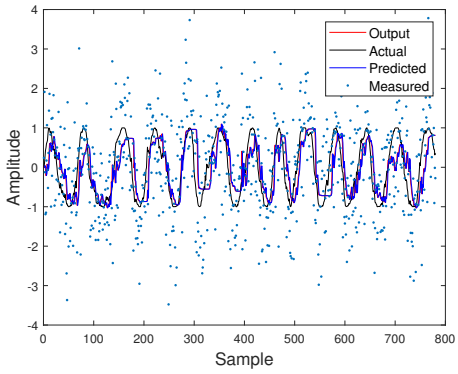
$$S_{t,t} = \left[ I - K_t \left( \frac{\partial g}{\partial x} \right) \right] S_{t,t-1} \tag{5}$$

Consider the tracking of an object in one dimension using a constant velocity model. The state variable is $X_t = \begin{bmatrix} x_t \\ \dot{x_t} \\ h_t \end{bmatrix}$ and the state transition equations for this model are $f(x_t, a_t) = \begin{bmatrix} x_{t+1} = x_t + T\dot{x_t} + 0 \\ \dot{x_{t+1}} = \dot{x_t} + a_t \\ h_{t+1} = sin\left(\frac{x_t}{10}\right) \end{bmatrix}$ where $x_t$ provides the position and $\dot{x_t}$, the velocity at time t. The parameter $h_t$ models the movement of the sine wave. Equations for the Extended Kalman filter are constructed to track the position of the particle.

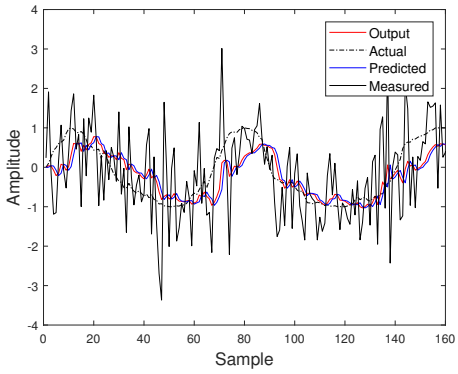
# 3 Results

The plot for the output of Extended Kalman filter is seen in Figures 1(a) and 1(b). Here the covariance of dynamic noise is given by $Q = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.00001 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ and the covariance of measurement noise is given by $R = \begin{bmatrix} 0.01 \end{bmatrix}$. Further filter outputs for three different ratios of dynamic noise to measurement noise are estimated.

Initially an Extended Kalman filter is built with the covariance of dynamic

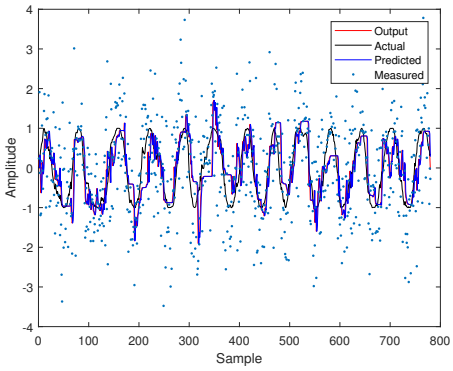



(a) Complete plot of the tracking

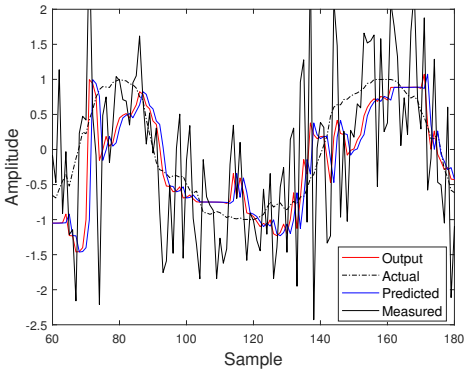



(b) A closer image of the same

Figure 2: Q=0.001 and R=0.1




(a) Complete plot of the tracking

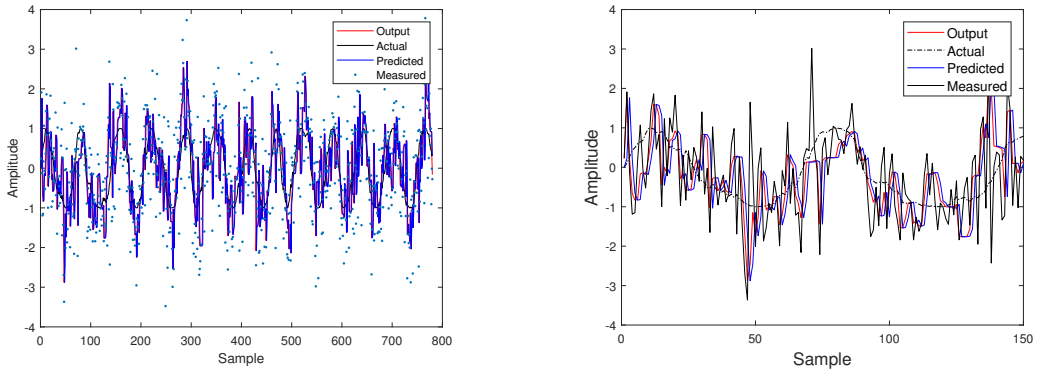



(b) A closer image of the same

Figure 3: Q=0.001 and R=0.01

noise $Q = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.001 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ and covariance of measurement noise $R = \begin{bmatrix} 0.1 \end{bmatrix}$. It is seen in Figure 2(a) and 2(b) that the output of the Extended Kalman Filter tries to follow the predicted value instead of the measurements.

For the second example,the dynamic noise and measurement noise covariance matrices are $Q = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.001 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ and $R = \begin{bmatrix} 0.01 \end{bmatrix}$ respectively.From the Figures 3(a) and 3(b) it can be inferred that the filter starts to follow the measured values.




(a) Complete plot of the tracking

(b) A closer image of the same

Figure 4: Q=0.001 and R=0.0001

For the final example the covariance matrix of dynamic noise is $Q = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.001 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ and covariance matrix of measurement noise is $R = \begin{bmatrix} 0.0001 \end{bmatrix}$. It is seen in Figures 4(a) and 4(b) that the output of the filter follows the measured values more closely which is because of the decrease in the value of R.

# 4 Conclusion

Although the Extended Kalman Filter is straightforward and simple it suffers from instability due to linearization of state and measurement equations. This linearization yields to approximation errors which the filter does not take into account. Results of the filter are reliable only if the time step intervals are sufficiently small. Also calculation of Jacobians increases the computational cost. On the other hand filters like Unscented Kalman filter and Cubature Kalman filter have higher accuracy and lower computational cost when compared to Extended Kalman filter.

# 5 Appendix

**Code for 1D Tracking:**

```
X=zeros(3,1);
Xt=zeros(3,1);
S=[.1 .0001 .0001;.0001 .1 .0001;.0001 .0001 .1];
```

```
St=zeros(3,3);
Kt=zeros(3,1);
T=1;
Q=[0 0 0;0 .00001 0;0 0 0];
R=[0.01];
t=1;
i=eye(3);
dfda =[0 0 0 ;0 1 0;0 0 0];
dgdx=[0 0 1];
dgdn=[1];
while t<=780

    Xt=X;               %predict-update cycle
    g=Xt(3,1);
    m=0.1*(cos(Xt(1,1)/10));
    dfdx = [1 T 0;0 1 0;m 0 0];
    St=(dfdx*S*transpose(dfdx))+(dfda*Q*transpose(dfda));
    yt=[sindata{t,2}];
    yt1=[sindata{t,1}];
    y1(t,1)=yt1(1,1);
    y(t,1)=yt(1,1);
    Kt=St*transpose(dgdx)*inv(dgdx*St*(transpose(dgdx))+dgdn*R*transpose(dgdn));
    X=Xt+Kt*(yt-(g));
    S=(i-(Kt*dgdx))*St;
    h(t,1)=X(3,1);
    h1(t,1)=Xt(3,1);
    t1(t,1)=t;
    t=t+1;

end
    plot(t1,h,'r',t1,y1,'k',t1,h1,'b',t1,y,'.');
    xlabel('Sample')
    ylabel('Amplitude')
    legend('Output','Actual','Predicted','Measured')
    hold on
hold off
```