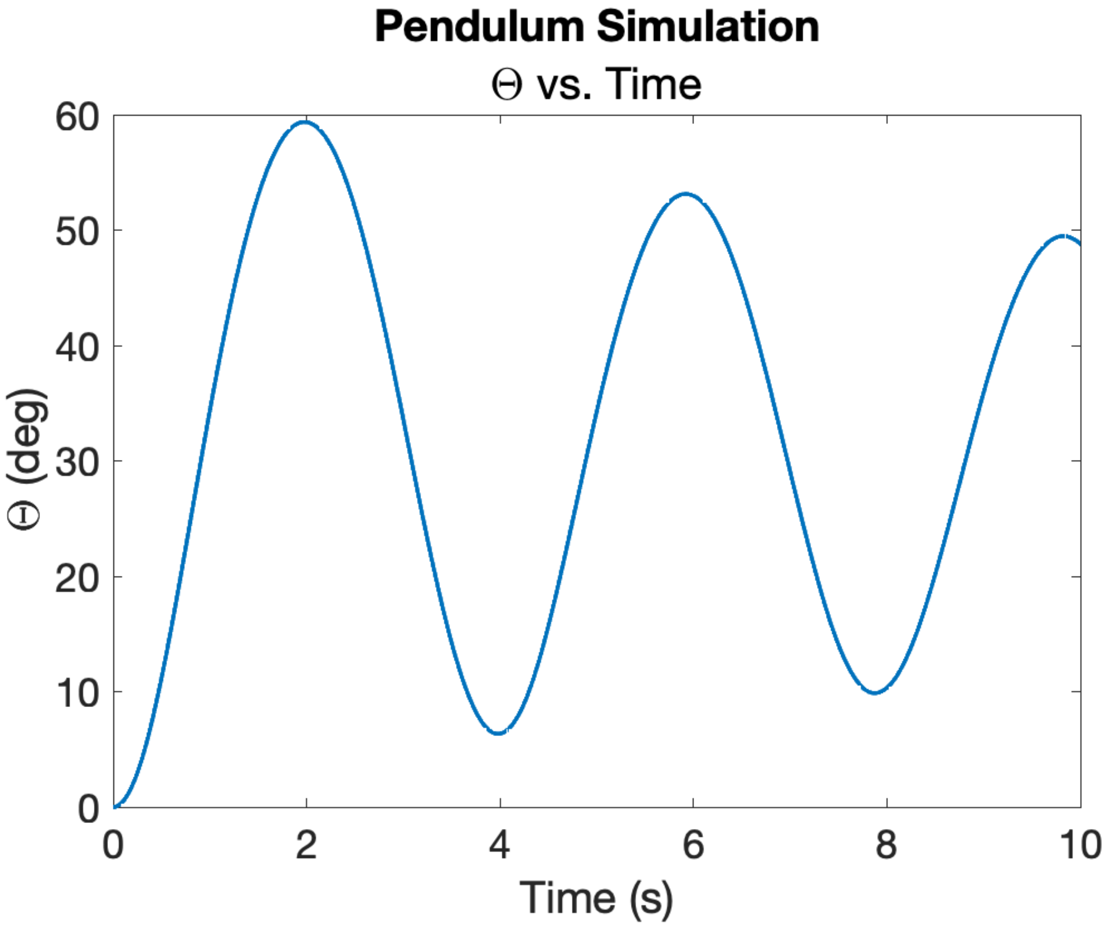
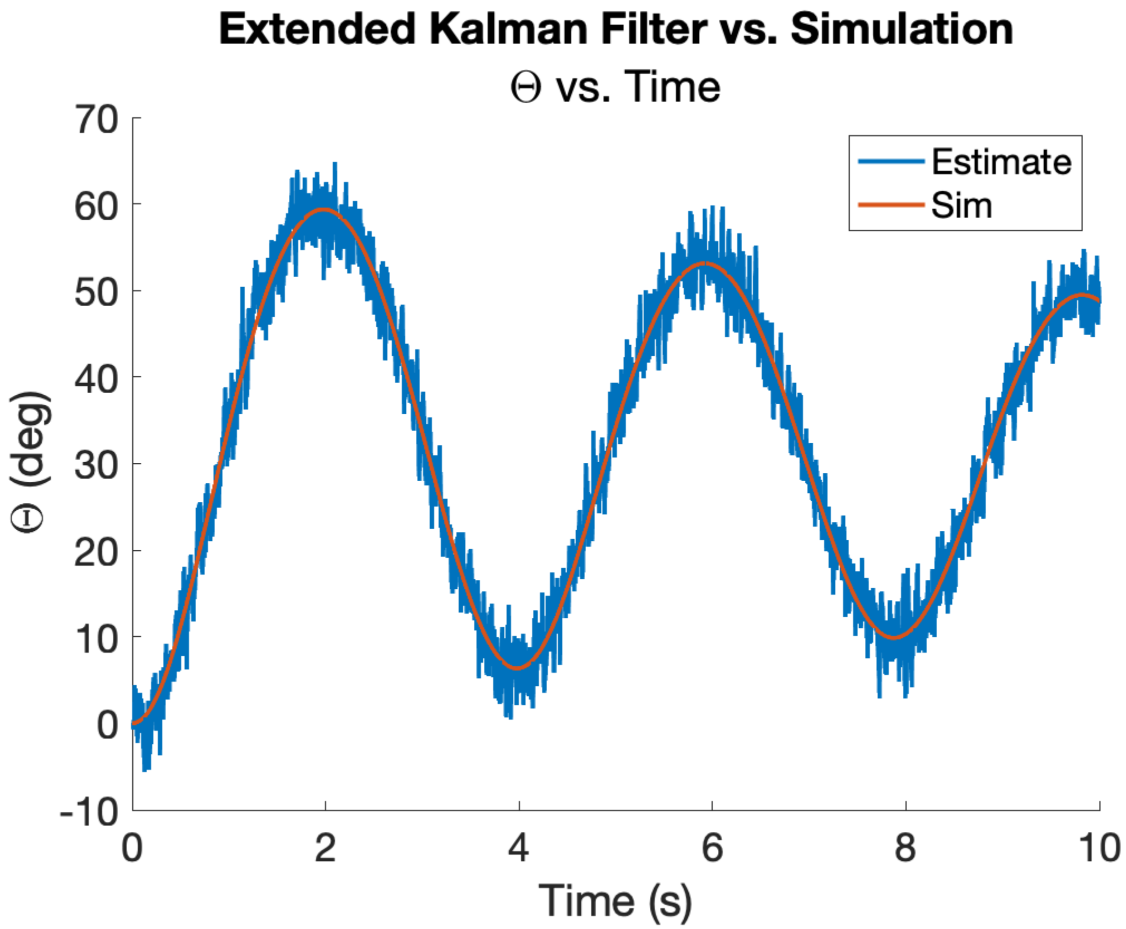
```
clear; close all; clc;

% Constants
g = -9.81;              % Gravity [m/s/s]

% System Parameters
Jp = 2.5;               % Mass Moment of Inertia @ Pivot[Nm/rad/s^2]
m = 1.6;                % Mass [kg]
l = 1;                  % Length [m]
b = 1.25;               % Damping [Nm/rad/s]
tau = 12;               % Constant Torque Input [Nm]

J = Jp + m*l^2;         % Mass Moment of Inertia [Nm/rad/s^2]

% Noise Parameters
sigmaF = sqrt(2);       % Force 1Sigma [N]

% Time
dt = 1e-3;              % Time Step [s]
time = 0:dt:10;         % Time Vector [s]
N = length(time);       % Length of Time

% Simulation
theta = zeros(1,N-1);   % Angular Position [rad]
thetad = zeros(1,N-1);  % Angular Velocity [rad/s]
thetadd = zeros(1,N-1); % Angular Acceleration [rad/s]
for k = 1:N-1
    F = 5 + sigmaF*randn(1);
    thetadd(k+1) = ((m*g*l)/J)*sin(theta(k)) - ((F*l)/J)*cos(theta(k)) - ...
        (b/J)*thetad(k)^3 + tau/J;
    thetad(k+1) = thetad(k) + thetadd(k+1)*dt;
    theta(k+1) = theta(k) + thetad(k+1)*dt;
end

figure();
plot(time, rad2deg(theta), 'LineWidth', 2);
title('Pendulum Simulation');
subtitle('\Theta vs. Time');
xlabel('Time (s)');
ylabel('\Theta (deg)');
ax = gca;
ax.FontSize = 18;
```



Extend Kalman Filter

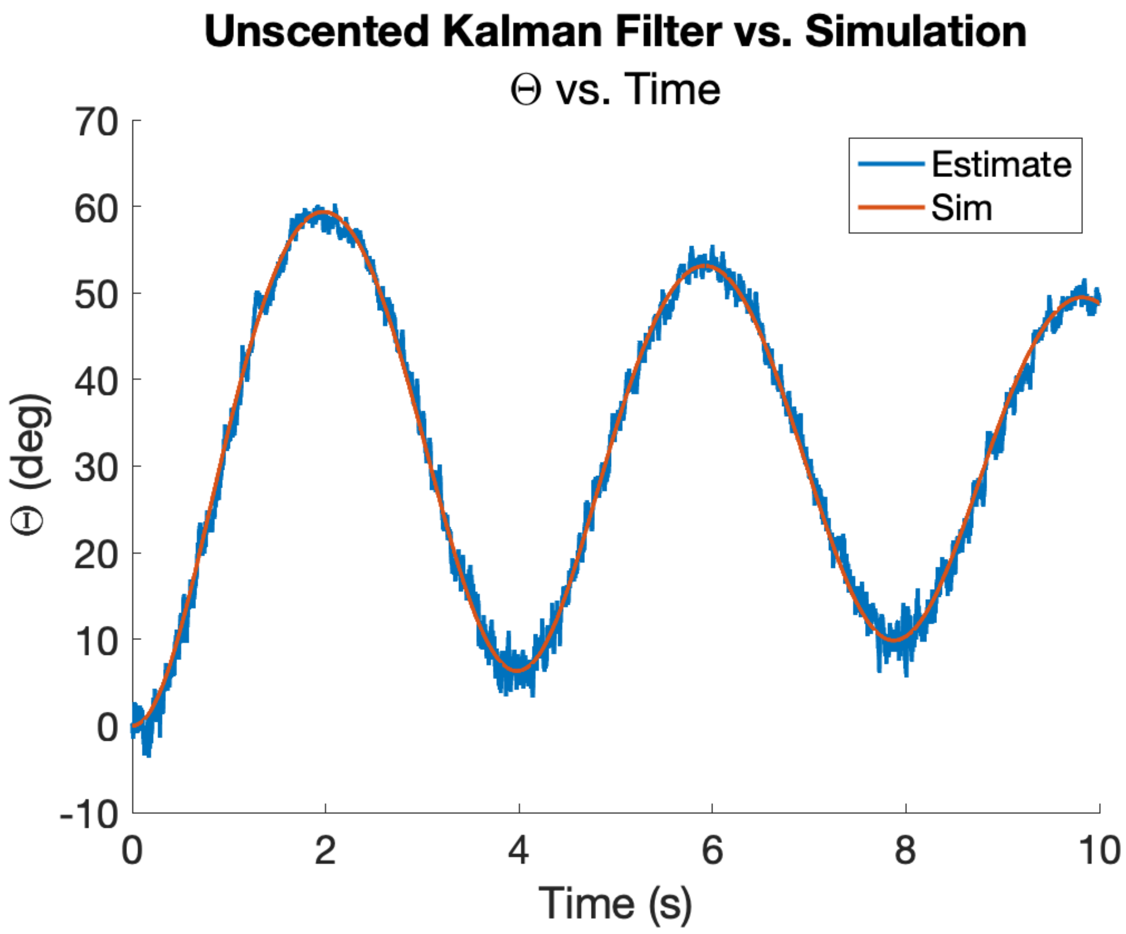
```
sigmaTheta = sqrt(deg2rad(1));          % Measurement 1Sigma [rad]

y = theta + sigmaTheta*randn(1,N);      % Measurements [rad]
A = [        0       1;
     (m*g*l)/J 3*(b/J)];                % Dynamic Matrix
Phi = expm(A*dt);
B = [  0;
     1/J];                              % Input Matrix
C = [1 0];
Bw = [0;
      1];                               % Input Noise Matrix

Qc = sigmaF^2;                          % Continuous Process Covariance
Qd = Bw*Qc*Bw';                         % Discrete Process Covariance
R = sigmaTheta^2;                       % Measurement Covariance
P = Qd;                                 % State Covariance
xEKF = zeros(2,N-1);                    % State
for k = 1:N-1
    % Time Update
    xp = [xEKF(1,k) + xEKF(2,k)*dt;
```

```
          xEKF(2,k) + (((m*g*l)/J)*sin(xEKF(1,k)) - (b/J)*xEKF(2,k)^3 + tau/
J)*dt];
    Pp = Phi*P*Phi' + Qd;
    % Kalman Gain
    K = Pp*C'/(C*Pp*C' + R);
    % Measurement Update
    xEKF(:,k+1) = xp + K*(y(k) - C*xp);
    P = (eye(2) - K*C)*Pp;
end

figure();
hold('on');
plot(time, rad2deg(xEKF(1,:)), 'LineWidth', 2);
plot(time, rad2deg(theta), 'LineWidth', 2);
title('Extended Kalman Filter vs. Simulation');
subtitle('\Theta vs. Time');
xlabel('Time (s)');
ylabel('\Theta (deg)');
legend('Estimate', 'Sim');
ax = gca;
ax.FontSize = 18;
```

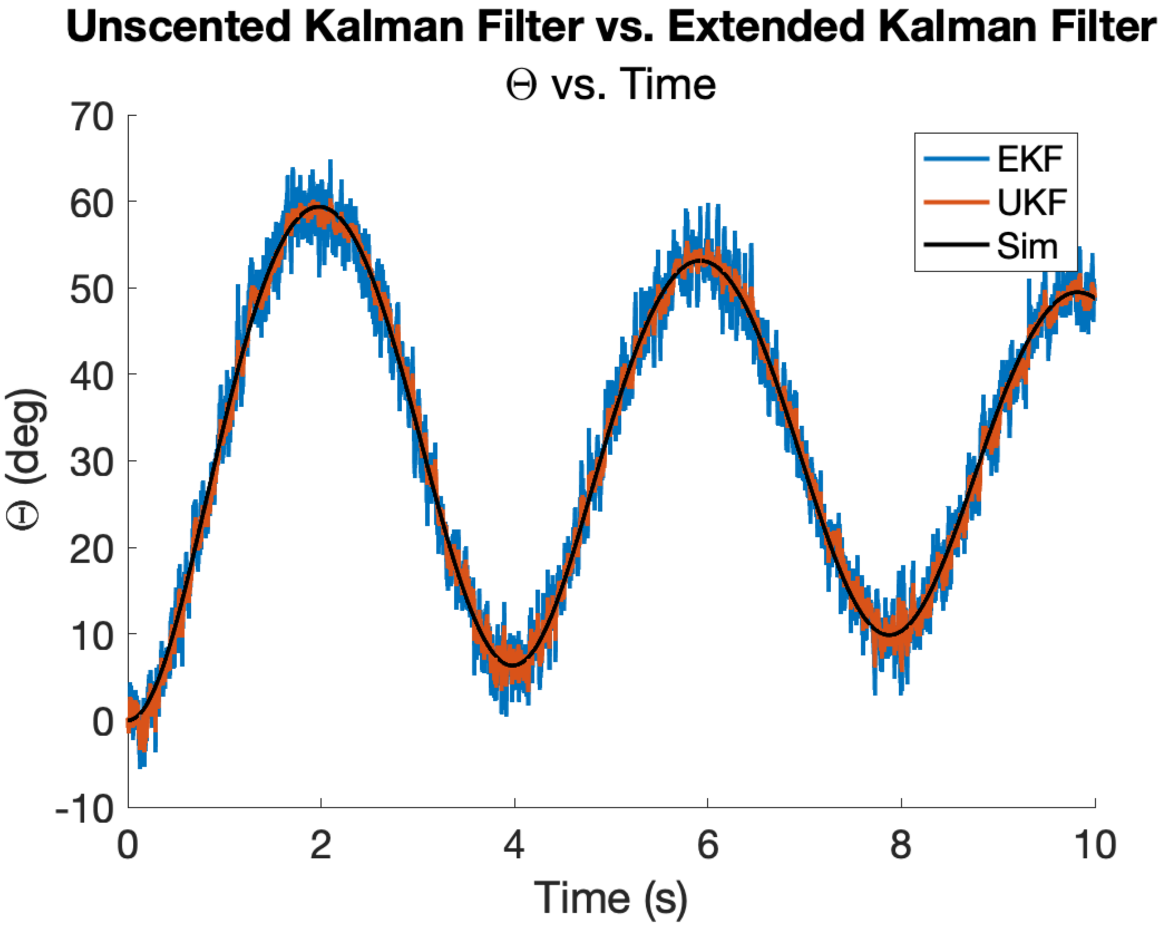


Unscented Kalman Filter

```
kappa = 1e-3;
n = 2;
na = 3;

xUKF = zeros(n, N-1);
xa = [xUKF; zeros(na-n, N-1)];
P = Qd;
Pa = blkdiag(P,Qd);

W = zeros(1,(2*na + 1));
W(1) = kappa/(na+kappa);
W(2:end) = 1/(2*(na + kappa));

X = zeros(n, 2*na + 1);
Xa = zeros(na, 2*na + 1);

for k = 2:N
    % Sigma Point Generation
    Pa = blkdiag(P,Qd);
    sigmaSpacing = sqrt((na + kappa)*Pa);

    xa(1:2,k-1) = xUKF(:,k-1);
    Xa(:,1) = xa(:,k-1);
    for i = 2:na+1
        Xa(:,i) = xa(:,k-1) + sigmaSpacing(i,i);
        Xa(:,i+na) = xa(:,k-1) - sigmaSpacing(i,i);
        X(:,i) = [Xa(1,i) + Xa(2,i)*dt;
                  Xa(2,i) + (((m*g*l)/J)*sin(Xa(1,i)) - (b/J)*Xa(2,i)^3 + tau/
J)*dt];
        X(:,i+na) = [Xa(1,i+na) + Xa(2,i+na)*dt;
                     Xa(2,i+na) + (((m*g*l)/J)*sin(Xa(1,i+na)) - (b/
J)*Xa(2,i+na)^3 + tau/J)*dt];
    end

    % Time Update
    xp = W(1)*X(:,1);
    for i = 2:2*na+1
        xp = xp + W(i)*X(:,i);
    end

    Pp = W(1)*(X(:,1) - xp)*(X(:,1) - xp)';
    for i = 2:2*na+1
        Pp = Pp + W(i)*(X(:,i) - xp)*(X(:,i) - xp)';
    end

    % Kalman Gain
    Y = X(1,:);
    yp = W(1)*Y(1);
    for i = 2:2*na+1
        yp = yp + W(i)*Y(i);
    end
    Pyy = W(1)*(Y(1) - yp)*(Y(1) - yp)';
    Pxy = W(1)*(X(1) - xp)*(Y(1) - yp)';
    for i = 2:(2*na+1)
        Pyy = Pyy + W(i)*(Y(i) - yp)*(Y(i) - yp)';
        Pxy = Pxy + W(i)*(X(i) - xp)*(Y(i) - yp)';
    end
    Pyy = Pyy + R;
    K = Pxy/Pyy;

    % Measurement Update
    xUKF(:,k) = xUKF(:,k-1) + K*(y(k) - yp);
    P = Pp - K*Pyy*K';
end

figure();
hold('on');
plot(time, rad2deg(xUKF(1,:)), 'LineWidth', 2);
plot(time, rad2deg(theta), 'LineWidth', 2);
title('Unscented Kalman Filter vs. Simulation');
subtitle('\Theta vs. Time');
xlabel('Time (s)');
ylabel('\Theta (deg)');
legend('Estimate', 'Sim');
ax = gca;
ax.FontSize = 18;

figure();
hold('on');
plot(time, rad2deg(xEKF(1,:)), 'LineWidth', 2);
plot(time, rad2deg(xUKF(1,:)), 'LineWidth', 2);
plot(time, rad2deg(theta), 'k', 'LineWidth', 2);
title('Unscented Kalman Filter vs. Extended Kalman Filter');
subtitle('\Theta vs. Time');
xlabel('Time (s)');
ylabel('\Theta (deg)');
legend('EKF', 'UKF', 'Sim');
ax = gca;
ax.FontSize = 18;
```



Unscented Kalman Filter vs. Simulation
Θ vs. Time
Estimate
Sim
Θ (deg)
70
60
50
40
30
20
10
0
-10
0
2
4
6
8
10
Time (s)



Published with MATLAB® R2023b