Melvin Cordon

Robot System ID

Homework 7

#3

The Kalman filter models the data a lot better than the OLS with a lot less noise and. Kalman filter is also doing this with one less parameter compared to the OLS (U\_dot\_dot was not included in the Kalman filter)



Figure 2: EKF of Thrust data



Figure 3: OLS of Thrust data

EKF FOR PROBLEM 3

%%% efk %%%

close all

clear all

clc

%data imported

Actual\_Thrust = csvread("Jan\_26\_19\_19-0.csv");

AActual\_Thrust = csvread("ManData.csv");

AAActual\_Thrust = csvread("StepInput.csv");

t = Actual\_Thrust(:,1)\*1e-6;

y = Actual\_Thrust(:,3);

u =Actual\_Thrust(:,1);

uu = deriv(u,.01);

uuu = deriv(uu,.01);

g= -9.81;

in = Actual\_Thrust(:,1);

%% Setting up the model matrices %%

dt = 0.01;

k = 1;

% b = 0.00001; % Damping coefficient

% grav = 9.81; % acceleration due to gravity guess

u(k) = 0; % Guess no acceleration to start

uu(k) = 0; % Guess no velocity to start

uuu(k) = 0

N = 4; % number of states

aa(k) = 0;

bb(k) = 0;

cc(k) = 0;

% Initial guess for parameters (initial cond. come from first data points)

Xk = [y(k);aa(k);bb(k);cc(k)];

% State update matrix (if nonlinear this may not be a matrix)

% Note: For pendulum, there is sin(angle), therefore not linear

% Input matrix - No input for pendulum

B = [0;0;0;0;0;0;0];

% Measurement matrix (what do we have estimates of).

H = [1 0 0 0]; % Only measuring angle and velocity

P = eye(N)\*.05; % Initial covariance matrix, update number as needed

Q = eye(N)\*.005;

R = eye(1)\*.1; %Size of different measurements (2 measurements here)

for k = 2:length(y)

F = [0 Xk(2) Xk(3) 1;

0 1 0 0;

0 0 1 0;

0 0 0 1];

%X\_pred = A\*Xk + B\*0 %+ [0; 0; -22.78]; % No Input at the moment

%X\_pred = A\*Xk + B\*0;% [0; 0; -22.78]; % No Input at the moment

X\_pred(1) = Xk(2)\*u(k) + Xk(3)\*uu(k) + Xk(4);

X\_pred(2) = Xk(2) ;

X\_pred(3) = Xk(3) ;

X\_pred(4) = Xk(4) ;

% X\_pred = X\_pred;

P\_pred = F\*P\*F' + Q;

Z = [y(k)]; % New measurements/data

yk = Z - H\*X\_pred';

Sk = H\*P\_pred\*H' + R;

Kk = P\_pred\*H'\*Sk^-1;

Xk = X\_pred' + Kk\*yk ;

P = (eye(N) - Kk\*H)\*P\_pred;

y\_kal(k) = Xk(1);

yd\_kal(k) = Xk(2);

ydd\_kal(k) = Xk(3);

end

yd(1) = 0;

yd(2:k) = diff(y)/dt;

[b,a] = butter(2,20/50);

yd(2:k) = filter(b,a,yd(2:k));

figure(1)

title('Pred')

plot(t, y, t, y\_kal)

xlabel('Time [sec]')

ylabel('Pos [m]')

legend('Meas','Kalman')

% figure(2)

% title('Velocity')

% plot(t, yd\_kal, t, yd)

% xlabel('Time [sec]')

% ylabel('Velocity [m/s]')

% legend('Kalman','Mesured')

%

% figure(3)

% title('Kalman Velocity')

% plot(t, yd\_kal)

% xlabel('Time [sec]')

% ylabel('Velocity [m/s]')

% legend('Kalman','Mesured')

%

% figure(4)

% title('Accleration')

% plot(t, ydd, t, ydd\_kal)

% xlabel('Time [sec]')

% ylabel('Acceleration [m/s^2]')

% legend('Data','Kalman')

# 1

From the graph we can see the parameters change over time. It can be noted that the acceleration parameter tends to vary a lot more then position and velocity parameter. This could possibly be due to

