Specialized course *Process Control* at Norwegian University of Life Sciences (NMBU), Spring 2018

Compulsory exercise for Lecture 6

1. **Kalman Filter**:
   a. Implement, in Matlab, a simulator of the liquid tank (including random process disturbances and measurement noise) and the Kalman Filter for the (simulated) tank presented in example 18.2 in the text-book. (Here is a SimView simulator of this system.) Implement the native algorithm of the Kalman gain (i.e. do not use any inbuilt Matlab-function for calculating K). Run some illustrative simulations (e.g. try various values of the Q matrix, and observe the effect on the estimates).
   b. Assume that the outflow is through a valve which is placed h0 [m] below the outlet of the tank, with the following relation between Fout and the liquid level:

      Fout = Kv*f(z)*sqrt[p/G]

      where Kv is valve constant, f is valve function (known), z is valve opening [%] (known), p [Pa] is pressure drop across the valve which assumed being equal to the hydrostatic pressure due to liquid level above the valve, rho is liquid density, g [m/s^2] is the gravity, and G [1] is specific density. Fout is not measured (and shall not be estimated either). Formulate the mathematical model (a continuous-time state-space model) which can be used to estimate Kv with a Kalman filter. Is the state-space model linear of nonlinear? (You are not required to implement this Kalman Filter.)

2. **Moving Horizon Estimation (MHE):** In Example 1.10 in the lecture notes about optimization, the process disturbance *d* is estimated. Modify the Matlab script presented in the example (the script is also available from the course home page) so that the gain *K* is estimated in stead of the *d*. (You can give *d* any fixed value you want.) Is your implementation successful?

3. **Model-predictive Control (MPC)**: Here is a Matlab script implementing MPC for a simulated air heater. (The model of the air heater, from control signal acting on the heater to measured tube outlet temperature, is basically time constant with time delay.) The script includes an Observer for estimating an “input disturbance”, which is a disturbance added to the control signal. One purpose of including such an estimated disturbance in MPC is to ensure zero steady-state control error despite model errors in the underlying model of the MPC. Note that in the present script, the Observer uses the control signal *after* the time-delay as control signal (or model input signal). In this way, the time-delay is not a part of the Oberver, which simplifies the implementation of the Observer.
   a. Open the script in Matlab, and try to understand it. Draw a block diagram which shows how the various parts (simulator; Observer; MPC) of the system implemented in the script, are interconnected.
   b. Run the script. Does it seem to work?
   c. Play with some of the settings (you should decide which), and observe the impact of the changes on the behaviour of the MPC control system.
   d. Implement a model error in the gain of the model. Check by simulations if using the Observer ensures steady state error despite this model error.

e. Voluntary: Replace the Observer in the script by a Kalman Filter which you should implement from scratch (that is, without using an inbuilt function in Matlab to calculate the Kalman gain). Is the system still working?

Updated 29 April 2018 by Finn Aakre Haugen, course teacher.