Maintained Zones in CSTR with Recycle

Eskild Schroll-Fleischer

September 27, 2020

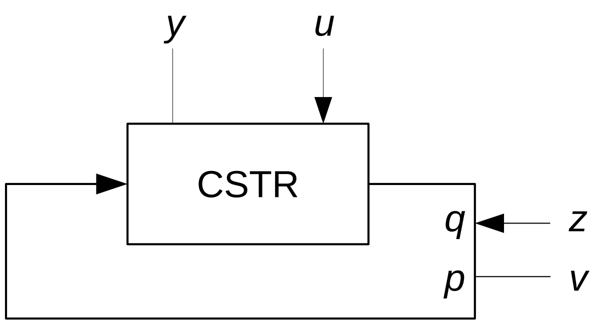

Figure 1: CSTR with PFR recycle. Two measurements y and v and two manipulated variables u and z.

Consider a CSTR of volume V with a recycle through a PFR of length L and cross-sectional area A at constant superficial velocity v, see figure 1. Measurement points are placed inside the CSTR and at the beginning of the PFR. Inlets are placed in the CSTR and just before the measurement point in the PFR. The objective is to simultaneously maintain one measurement value y in the CSTR by manipulating u and another measurement value v in the PFR by manipulating z. The measurement could for example be a concentration of a component or a temperature.

Point v is placed at axial position $p\,L$ relative to the outlet to the PFR and correspondingly z is placed at axial position $q\,L$. In this terminology $0 < p < q < 1$. The residence time in the CSTR is $\tau_c = V\,(v\,A)^{-1}$ and the residence time in the PFR is $\tau_p = L\,v^{-1}$.

The sensor in the PFR behaves as a first order system even though plug flow conditions prevail in the PFR. The time constant of this first order behaviour is τ_s.

DDE Model

The system is modeled by equations (1) to (4) where one unit of time t corresponds to one residence time τ_p in the PFR.

$$\frac{\mathrm{d}x_1(t)}{\mathrm{d}t} = \tau_c^{-1}\left(u(t) + x_2(t-p) - x_1(t)\right), \text{ for } t > 0,\ x_1(t \le 0) = 0 \tag{1}$$

$$\frac{\mathrm{d}x_2(t)}{\mathrm{d}t} = \tau_s^{-1}\left(z(t-(q-p)) + x_1(t-(1-p)) - x_2(t)\right), \text{ for } t > 0,\ x_2(t \le 0) = 0 \tag{2}$$

The measurements are then $y = x_1$ and $v = x_2$. Both u and z are PI-controllers:

$$u(t) = K_{p,\,u}\left(-x_1(t) + \tau_{I,\,u}^{-1} I_u\right), \;\; \frac{\mathrm{d}I_u(t)}{\mathrm{d}t} = -x_1(t), \text{ for } t > 0, \; I_u(t \le 0) = 0 \tag{3}$$

$$z(t) = K_{p,\,z}\left((x_{\mathrm{PFR}} - x_2(t)) + \tau_{I,\,z}^{-1} I_z\right), \;\; \frac{\mathrm{d}I_z(t)}{\mathrm{d}t} = x_{\mathrm{PFR}} - x_2(t), \text{ for } t > 0, \; I_z(t \le 0) = 0 \tag{4}$$

It is desired to move the time dependence of z from equation (2) to equation (4) to ease implementation of the model in MATLAB® R2018a. This is accomplished as follows:

$$\frac{\mathrm{d}x_2(t)}{\mathrm{d}t} = \tau_s^{-1}\left(z(t) + x_1(t-(1-p)) - x_2(t)\right), \text{ for } t > 0, \; x_2(t \le 0) = 0 \tag{5}$$

$$z(t) = K_{p,\,z}\left((x_{\mathrm{PFR}} - x_2(t-(q-p))) + \tau_{I,\,z}^{-1} I_z\right), \;\; \frac{\mathrm{d}I_z(t)}{\mathrm{d}t} = x_{\mathrm{PFR}} - x_2(t-(q-p)) \tag{6}$$

The points in history which are required are then: $t-p$, $t-(q-p)$ and $t-(1-p)$.

Controller tuning

The PI-controllers are tuned according to the Skogestad SIMC rules which dictate that for a unit gain process with time constant τ and time delay θ:

$$K_p = \frac{\tau}{\theta + \tau_c} \text{ and } \tau_I = \min\left(\tau, 4(\theta + \tau_c)\right) \text{ for } \tau_c = \frac{\tau}{5} + \theta \tag{7}$$

The time constant in the CSTR is τ_c and for the PFR τ_s. There is a time delay in the PFR and this amounts to $(q-p)$ which is the temporal distance between manipulation and measurement.

Simulation

The DDE model in equations (1), (3), (5) and (6) is implemented in MATLAB® R2018a, see appendix A.

A simulation of a closed loop experiment is presented in figure 2. The simulation shows that the two zones are maintained.

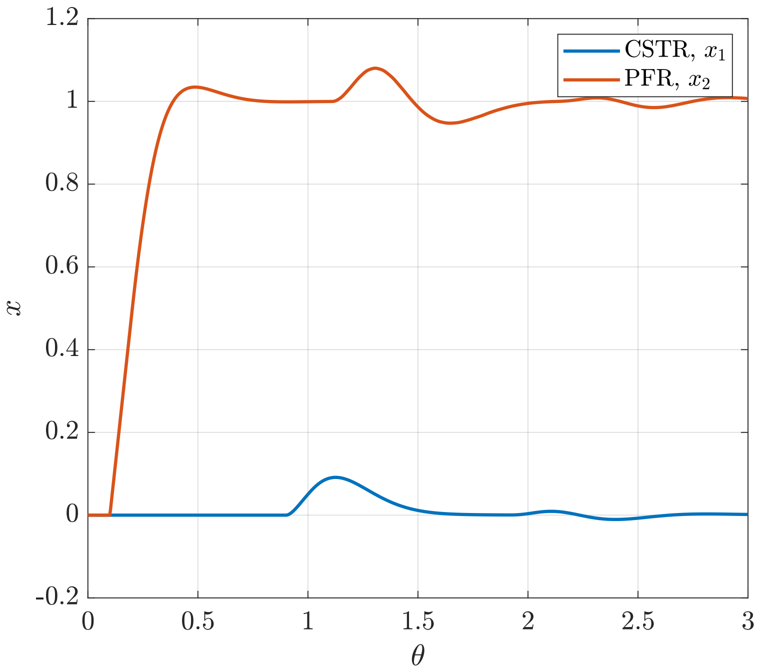

Figure 2: Simulation for $\tau_c = 0.2$, $\tau_s = 0.02$, $q = 0.9$, $p = 0.8$ and initial history $x_1(t \leq 0) = x_2(t \leq 0) = 0$.

A maintained_zones.m

```
set(0,'defaultlinelinewidth',1.7,'DefaultAxesFontSize',14,'DefaultAxesXGrid','on','DefaultAxesYGrid','
     on','defaultTextInterpreter','latex', 'DefaultFigurePosition', [0 0 600 500])
set(groot,'defaultAxesTickLabelInterpreter','latex','defaultLegendInterpreter','latex');
clear;clc;close all;
%% Parameters
P.tau_cstr = 0.2; % [theta], time constant of CSTR
P.tau_pfr = 0.02; % [theta], time constant of sensor in PFR
P.p = 0.8; % Distance from end of PFR to measured variable
P.q = 0.9; % Distance from end of PFR to manipulated variable
tau_c_cstr = P.tau_cstr/5;
P.Kpu = P.tau_cstr/tau_c_cstr; % CSTR controller gain
P.tauIu = min(P.tau_cstr,4*tau_c_cstr); % [theta], CSTR controller integral time
tau_c_pfr = P.tau_pfr/5+(P.q-P.p);
P.Kpz = P.tau_pfr/((P.q-P.p)+tau_c_pfr); % PFR controller gain
P.tauIz = min(P.tau_pfr,4*((P.q-P.p)+tau_c_pfr)); % [theta], PFR controller integral time
n_theta = 3; % Simulation duration [theta]
%% Solve DDE model
sol = dde23(@model_dde,[P.p, (1-P.p), (P.q-P.p)],@model_dde_history,[0, n_theta],[],P);
% Extract solution
theta = linspace(0,n_theta,1000);
y = deval(sol, theta);
x1 = y(1,:);
x2 = y(2,:);
%% Plot of solutions side by side
figure
plot(theta,x1)
hold on
plot(theta,x2)
ylabel('$x$')
xlabel('$\theta$')
legend('CSTR, $x_1$', 'PFR, $x_2$')
%print('plot','-dpdf')
%% DDE history function
function [x] = model_dde_history(t,P)
    x = zeros(4,1);
end
%% DDE model function
function [dx] = model_dde(t,x,xh,P)
    % Pick relevant states
    x1 = x(1);
    x2 = x(2);
    Iu = x(3);
    Iz = x(4);
    x2ptheta = xh(2,1); % x2(t-p)
    x11ptheta = xh(1,2); % x1(t-(1-p))
    x2qptheta = xh(2,3); % x2(t-(q-p))
    % Integral states are calculated:
    dIudt = -x1;
    if t > (P.q-P.p)
        dIzdt = 1-x2qptheta;
    else
        dIzdt = 0;
    end
    % Control action
    u = P.Kpu*(dIudt + Iu/P.tauIu);
    z = P.Kpz*(dIzdt + Iz/P.tauIz);
    % DDEs
    dx1dt = (u+x2ptheta-x1)/P.tau_cstr;
    dx2dt = (z+x11ptheta-x2)/P.tau_pfr;

    dx = [dx1dt; dx2dt; dIudt; dIzdt];
end
```