A Design Study Approach to Classical Control

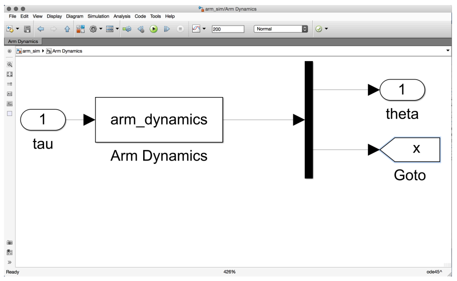
Randal W. Beard Timothy W. McLain
Brigham Young University

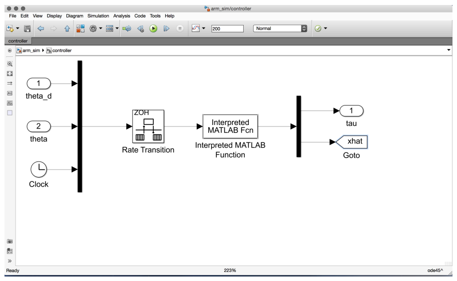
Updated: May 26, 2016

Homework A.13

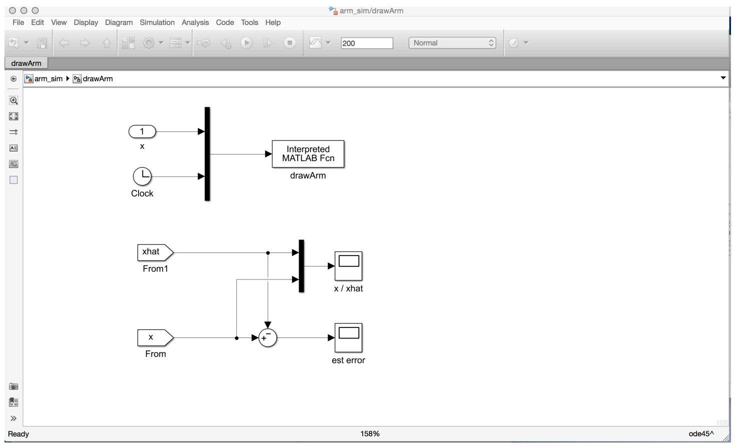
The objective of this problem is to design an observer that estimates the state of the system and uses the estimated state in the controller designed in Homework A.12.

(a) The first step is to modify the Simulink diagram so that we can observe the performance of the observer. Modify the Simulink diagram from Homework A.112 so that the output of the s-function defining the dynamics is both y and x as shown in Figure . In addition, modify the Simulink diagram so that the controller outputs both u and $\hat{x}$, as also shown in Figure .





The animation block can also be modified to include plots of x and $\hat{x}$ and also the estimation error $x - \hat{x}$, as shown in Figure .



The s-function defining the system will need to be modified so that the output of the block is both the normal output, as well as the actual state:

```
sizes.NumOutputs      = 1+2; % in mdlInitializeSizes

function sys=mdlOutputs(t,x,u,P)
  theta     = x(1);
  thetadot  = x(2);
  tau       = u(1);
  sys = [theta; x];
```

Similarly, the output of the control block will need to be modified so that the output of the function is both the control output u as well as the state estimate $\hat{x}$:

```
function out=arm_ctrl(in,P)
    % control code goes here
    out = [u; xhat];
end
```

(b) For the sake of understanding the function of the observer, for this problem we will use exact parameters, without an input disturbance. Modify `arm_dynamics.m` so that the parameters known to the controller are the actual plant parameters (uncertainty parameter $\alpha = 0$).

(c) Verify that the state space system is observable by checking that $\text{rank}(\mathcal{O}_{A,C}) = n$.

(d) In the control block, add an observer to estimate the state $\hat{x}$, and use the estimate of the state in your feedback controller. Tune the poles of the controller and observer to obtain good performance.

(e) As motivation for the next chapter, add an input disturbance to the system of 0.01 and observe that there is steady state error in the response even though there is an integrator. This is caused by a steady state error in the observation error. In the next chapter we will show how to remove the steady state error in the observation error.

Solution

Matlab code used to design the observer based controller is shown below:

```
clear all

% initial conditions
P.theta0 = 0;
P.thetadot0 = 0;

% system parameters known to controller
P.m = 0.5;    % kg
P.ell = 0.3; % m
P.b = 0.01;  % N m s
P.g = 9.8;    % m/s^2

% sample rate
P.Ts = 0.01;

% equalibrium torque
P.theta_e = 0*pi/180;
P.tau_e   = P.m*P.g*P.ell/2*cos(P.theta_e);

% saturation constraint
P.tau_max = 1;
tau_max = P.tau_max-P.tau_e;

%---------------------
```

```
% state space design
P.A = [...
    0, 1;...
    0, -3*P.b/P.m/(P.ell^2);...
    ];
P.B = [0; 3/P.m/(P.ell^2) ];
P.C = [...
    1, 0;...
    ];

% form augmented system
A1 = [P.A, zeros(2,1); -P.C, 0];
B1 = [P.B; 0];

% tuning parameters for control
wn = 5.5;
zeta = 0.707;
integrator_pole = -5;

% compute gains
des_char_poly = conv([1,2*zeta*wn,wn^2],poly(integrator_pole));
des_poles = roots(des_char_poly);
if rank(ctrb(A1,B1))≠3,
    disp('System Not Controllable');
else % if so, compute gains
    K1   = place(A1,B1,des_poles);
    P.K  = K1(1:2);
    P.ki = K1(3);
end

% observer design
% tuning parameters for observer
wn_obs = 10;
zeta_obs = 0.707;

% desired observer poles
des_obsv_char_poly = [1,2*zeta*wn,wn^2];
des_obsv_poles = roots(des_obsv_char_poly);

% is the system observable?
if rank(obsv(P.A,P.C))≠2,
    disp('System Not Observable');
else % if so, compute gains
    P.L = place(P.A',P.C',des_obsv_poles)';
end
```

Lines 1–53 are identical to the state feedback design from Homework A.12 except that we are using ω_n as a tuning parameter instead of t_r. This is to highlight the bandwidth separation between the controller and the observer. The poles of the observation error are selected in Lines 55–69 in a form that allows $\omega_{n_{obs}}$ to be tuned for performance. The observability check is in Line 64–66, and the observer gains are computed using the `place` command in Line 68. Note the transposes on this line.

Matlab code for the observer based control is shown below:

```matlab
1  function out=arm_ctrl(in,P)
2      theta_r = in(1);
3      theta_m = in(2);
4      t       = in(3);
5
6      % implement observer
7      persistent xhat       % estimated state (for observer)
8      persistent tau         % delayed input (for observer)
9      if t<P.Ts,
10         xhat   = [0; 0];
11         tau    = 0;
12     end
13     N = 10;
14     % compute equilibrium torque at thetahat
15     tau_e = P.m*P.g*(P.ell/2)*cos(xhat(1));
16     for i=1:N,
17         xhat = xhat + ...
18             P.Ts/N*(P.A*xhat+P.B*(tau-tau_e)...
19                     +P.L*(theta_m-P.C*xhat));
20     end
21     thetahat = xhat(1);
22
23     % implement integrator
24     error = theta_r - thetahat;
25     persistent integrator
26     persistent error_d1
27     % reset persistent variables at start of simulation
28     if t<P.Ts==1,
29         integrator  = 0;
30         error_d1    = 0;
31     end
32     integrator = integrator + (P.Ts/2)*(error+error_d1);
33     error_d1 = error;
34
```

```
35      % compute control torque
36      tau_e = P.m*P.g*(P.ell/2)*cos(thetahat);
37      tau_unsat = tau_e - P.K*xhat - P.ki*integrator;
38      tau = sat( tau_unsat, P.tau_max);
39
40      % integrator anti-windup
41      if P.ki≠0,
42         integrator = integrator + P.Ts/P.ki*(tau-tau_unsat);
43      end
44
45      out = [tau; xhat];
46  end
47
48  function out = sat(in,limit)
49      if      in > limit,       out = limit;
50      elseif in < -limit,      out = -limit;
51      else                      out = in;
52      end
53  end
```

The observer is implemented in Lines 6–21. Note that the control signal on line 37 uses $\hat{x}$ instead of x, and that $\hat{\theta}$ is used instead of θ on lines 24 and 36. See the wiki for the complete solution.