**%% switching control law**

% u = u\_hat - k\*sgn(s)

% ------------------------------------------------------------------------

**%% setup**

% controller parameters

eta = 0.1;

phi = 0.1;

lambda = 20;

% simulation time-step

dt = 0.0001;

% ------------------------------------------------------------------------

**%% initial conditions**

% time

t(1) = 0;

% state

x(1) = 0;

x\_dot(1) = 0;

x\_dotdot(1) = 0;

a(1) = 1 + abs(sin(t(1)));

f(1) = a(1)\*x\_dot(1)^2\*cos(3\*x(1));

% desired trajectory

x\_d(1) = sin((pi/2)\*t(1));

x\_d\_dot(1) = (x\_d(1)-0)/dt;

x\_d\_dotdot(1) = (x\_d\_dot(1)-0)/dt;

% error

x\_tilda(1) = x(1) - x\_d(1);

x\_tilda\_dot(1) = (x\_tilda(1) - 0)/dt;

x\_tilda\_dotdot(1) = (x\_tilda\_dot(1) - 0)/dt;

% control input

f\_hat(1) = -1.5\*x\_dot(1)^2\*cos(3\*x(1));

v(1) = x\_d\_dotdot(1) - lambda\*x\_tilda\_dot(1);

u\_hat(1) = -f\_hat(1) + v(1);

F(1) = 0.5\*x\_dot(1)^2\*abs(cos(3\*x(1)));

k(1) = F(1) + eta;

s(1) = x\_tilda\_dot(1) + lambda\*x\_tilda(1);

u(1) = u\_hat(1) - k(1)\*sign(s(1));

% ------------------------------------------------------------------------

**%% time-loop**

for i = 2:(4/dt)

% time

t(i) = t(i-1) + dt;

% state

a(i) = 1 + abs(sin(t(i)));

f(i) = a(i)\*x\_dot(i-1)^2\*cos(3\*x(i-1));

x\_dotdot(i) = u(i-1) - f(i);

x\_dot(i) = x\_dot(i-1) + dt\*x\_dotdot(i-1);

x(i) = x(i-1) + dt\*x\_dot(i-1);

% desired trajectory

x\_d(i) = sin((pi/2)\*t(i));

x\_d\_dot(i) = (x\_d(i) - x\_d(i-1))/dt;

x\_d\_dotdot(i) = (x\_d\_dot(i) - x\_d\_dot(i-1))/dt;

% error

x\_tilda(i) = x(i) - x\_d(i);

x\_tilda\_dot(i) = (x\_tilda(i) - x\_tilda(i-1))/dt;

x\_tilda\_dotdot(i) = (x\_tilda\_dot(i) - x\_tilda\_dot(i-1))/dt;

% control input

f\_hat(i) = -1.5\*x\_dot(i)^2\*cos(3\*x(i));

v(i) = x\_d\_dotdot(i) - lambda\*x\_tilda\_dot(i);

u\_hat(i) = -f\_hat(i) + v(i);

F(i) = 0.5\*x\_dot(i)^2\*abs(cos(3\*x(i)));

k(i) = F(i) + eta;

s(i) = x\_tilda\_dot(i) + lambda\*x\_tilda(i);

u(i) = u\_hat(i) - k(i)\*sign(s(i));

end

% ------------------------------------------------------------------------

**%% Display**

figure(1)

subplot(1,2,1);

plot(t, u);

xlabel('time (s)')

ylabel('control input')

title('Part 1 - Switching')

axis([0 4 -6 6])

subplot(1,2,2);

plot(t, x\_tilda);

xlabel('time (s)')

ylabel('tracking error')

title('Part 1 - Switching')

axis([0 4 -15\*10^-6 15\*10^-6])

**%% interpolated control law**

% u = u\_hat - k\*sat(s/phi)

% ------------------------------------------------------------------------

**%% setup**

…same…

% ------------------------------------------------------------------------

**%% initial conditions**

…same…

% ------------------------------------------------------------------------

**%% time-loop**

for i = 2:(4/dt)

…

% control input

f\_hat(i) = -1.5\*x\_dot(i)^2\*cos(3\*x(i));

v(i) = x\_d\_dotdot(i) - lambda\*x\_tilda\_dot(i);

u\_hat(i) = -f\_hat(i) + v(i);

F(i) = 0.5\*x\_dot(i)^2\*abs(cos(3\*x(i)));

k(i) = F(i) + eta;

s(i) = x\_tilda\_dot(i) + lambda\*x\_tilda(i);

u(i) = u\_hat(i) - k(i)\*sat(s(i), phi);

end

% ------------------------------------------------------------------------

**%% Display**

…same…

%% ------------------------------------------------------------------------

**%% saturation function**

function [sats] = sat(s, phi)

if abs(s) < phi

sats = s/phi;

else

sats = sign(s);

end

end

%% ------------------------------------------------------------------------

%% **variable phi interpolated control law**

% u = u\_hat - k\_bar\*sat(s/phi)

% phi\_dot = k\_d - lambda\*phi;

% k\_bar = k - phi\_dot;

% ------------------------------------------------------------------------

%% setup

…same…

% ------------------------------------------------------------------------

%% initial conditions

…same…

% ------------------------------------------------------------------------

%% time-loop

for i = 2:(4/dt)

…

% control input

phi(i) = phi(i-1) + phi\_dot(i-1)\*dt;

f\_hat(i) = -1.5\*x\_dot(i)^2\*cos(3\*x(i));

v(i) = x\_d\_dotdot(i) - lambda\*x\_tilda\_dot(i);

u\_hat(i) = -f\_hat(i) + v(i);

F(i) = 0.5\*x\_dot(i)^2\*abs(cos(3\*x(i)));

k(i) = F(i) + eta;

s(i) = x\_tilda\_dot(i) + lambda\*x\_tilda(i);

F\_d(i) = 0.5\*x\_d\_dot(i)^2\*abs(cos(3\*x\_d(i)));

k\_d(i) = F\_d(i) + eta;

phi\_dot(i) = k\_d(i) - lambda\*phi(i);

k\_bar(i) = k(i) - phi\_dot(i);

u(i) = u\_hat(i) - k\_bar(i)\*sat(s(i), phi(i));

end

% -------------------------------------------------------------------------

**%% Display**

…same…





