1.b)

zeta\_course = 2.1;

W\_course = 30;

w\_course = omega\_n\_phi/W\_course;

Vg = P.Va0;

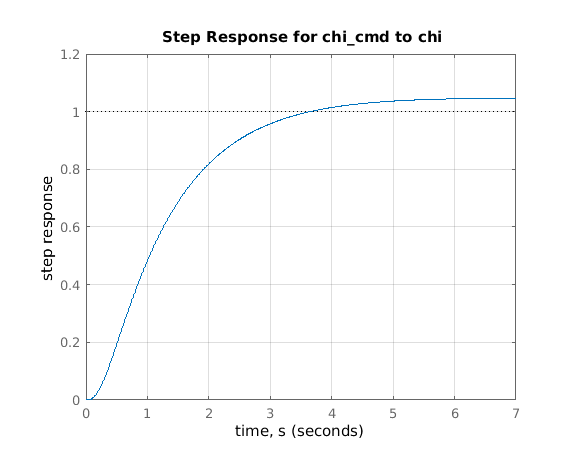
gravity = P.gravity;

P.course\_kp = 2\*zeta\_course\*w\_course\*Vg/gravity;

P.course\_ki = w\_course\*w\_course\*Vg/gravity;

P.course\_kd = 0.0;

1.c)



1.d)

function u = PIR\_course\_hold(chi\_c, chi\_hat, not\_used, init\_flag, P)

% Set up PI with rate feedback

y\_c = chi\_c; % Command

y = chi\_hat; % Feedback

y\_dot = not\_used; % Rate feedback % This was zero

kp = P.course\_kp;

ki = P.course\_ki;

kd = P.course\_kd;

u\_lower\_limit = -P.phi\_max;

u\_upper\_limit = +P.phi\_max;

% Initialize integrator (e.g. when t==0)

persistent error\_int;

if( init\_flag )

error\_int = 0;

end

% Perform "PI with rate feedback"

error = mod(y\_c - y + pi, 2\*pi) - pi; % Error between command and response

error\_int = error\_int + P.Ts\*error; % Update integrator

u = kp\*error + ki\*error\_int - kd\*y\_dot;

% Output saturation & integrator clamping

% - Limit u to u\_upper\_limit & u\_lower\_limit

% - Clamp if error is driving u past limit

if u > u\_upper\_limit

u = u\_upper\_limit;

if ki\*error>0

error\_int = error\_int - P.Ts\*error;

end

elseif u < u\_lower\_limit

u = u\_lower\_limit;

if ki\*error<0

error\_int = error\_int - P.Ts\*error;

end

end

end

1.e)

