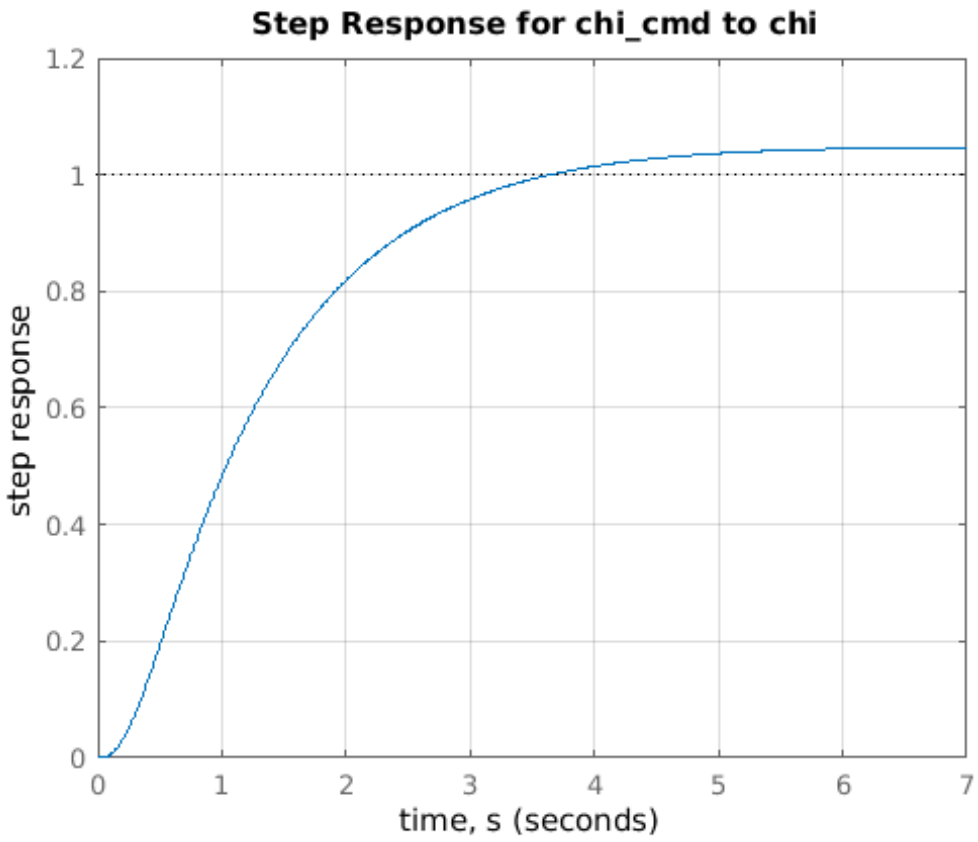
1.b)

```
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)

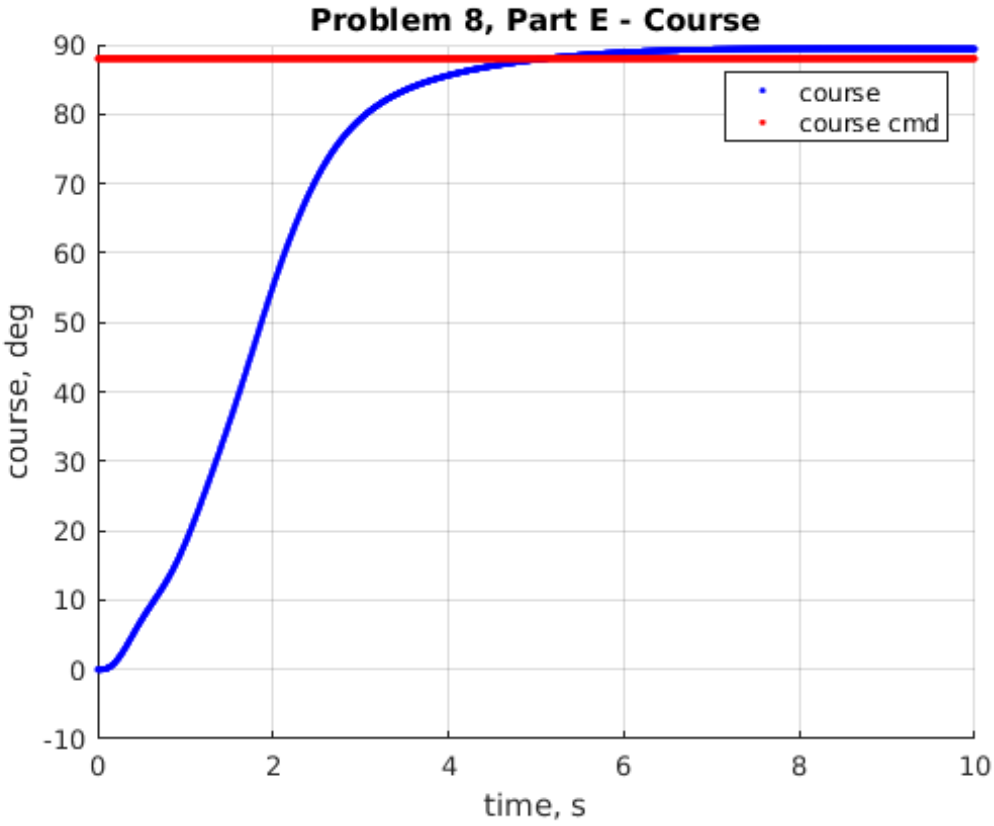


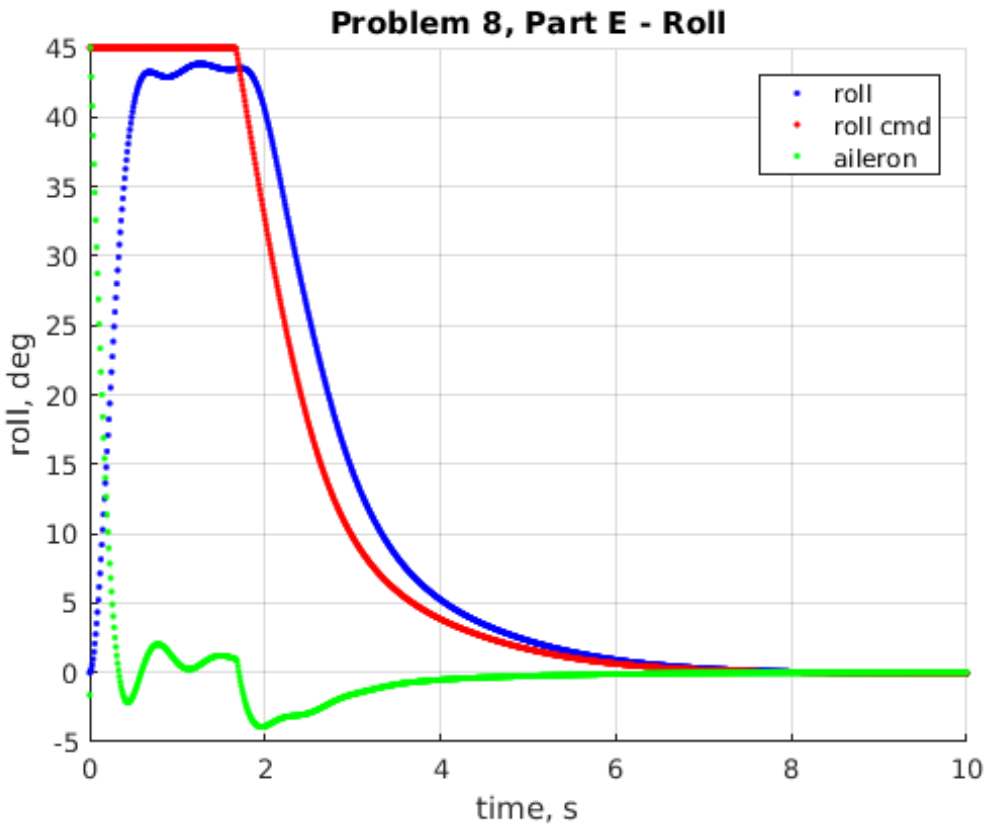
1.d)

```
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)




Problem 8, Part E - Roll
roll
roll cmd
aileron
roll, deg
time, s
45
40
35
30
25
20
15
10
5
0
-5
0
2
4
6
8
10