```
%% Problem 4: Cruise Control

%% Part A: Equilibrium for open and closed loop controllers

% beta = a/a_hat: use as domain from 0:2
beta = 0:0.05:2

% let Vss/Vref = V
V_ff = 1./beta
V_fb = 10./(beta + 10)

figure(1)
clf
hold on
plot(beta, V_ff)r', 'latex')
hold off

saveas(1, "ES155P0_4a_steadystatevsbeta.png")

%% Part C: PI controller Modelling

omega_0 = [0.01 0.1 1 10];

% Part i
err = compute_PI_err([1 0], 2, omega_0);
saveas(gcf, "ES155P1_4ci_output.png")
ploterr([1 0], omega_0, err, 3);
saveas(gcf, "ES155P1_4ci_error.png")

% Part ii
err = compute_PI_err([1 1], 4, omega_0);
saveas(gcf, "ES155P1_4cii_output.png")
ploterr([1 1], omega_0, err, 5);
saveas(gcf, "ES155P1_4cii_error.png")


% Part iii
err = compute_PI_err([1 10], 6, omega_0);
saveas(gcf, "ES155P1_4ciii_output.png")
ploterr([1 10], omega_0, err, 7);
saveas(gcf, "ES155P1_4ciii_error.png")



function ploterr(gains, omega_0, err_vals, fignum)
    figure(fignum); clf;
    loglog(omega_0, err_vals);
    xlabel('$\omega_0$, log scale', 'interpreter', 'latex')
    ylabel('Error Amplitude in 10th Period, log scale')
    title(sprintf('Error vs Input Frequency, $k_p =$ %d, $k_i =$ %d', gains(1), gains(2)),
'interpreter', 'latex')
end

function err_pp_vals = compute_PI_err(gains, fignum, omega_0)
    % setup
    num_freqs = length(omega_0);

    % save err_pp in vector
    err_pp_vals = zeros(size(omega_0));

    % prepare plotting
    figure(fignum); clf;

    titlestring = {'PI Cruise Control Output and Error', sprintf('$k_p =$ %d, $k_i =$ %d',
```

```
gains(1), gains(2))};
    suptitle = annotation('textbox', [.2 .9 .6 .1], 'String', titlestring, 'interpreter',
'latex')
    suptitle.HorizontalAlignment = 'center';
    suptitle.LineStyle = 'none';

    % compute and plot
    for i = 1:num_freqs
        [t, y_D, y, err, err_pp_vals(i)] = PIcontrol_err(omega_0(i), gains);

        cur_omega_0 = omega_0(i);

        subplot(num_freqs, 2, 2*i - 1)
        plot(t,y)
        hold on
        plot(t, y_D)
        hold off
        title(sprintf('Output, $\\omega_0 =$ %.2f', cur_omega_0), 'interpreter', 'latex')

        subplot(num_freqs, 2, 2*i)
        plot(t, err)
        title(sprintf('Error, $\\omega_0 =$ %.2f', cur_omega_0), 'interpreter', 'latex')
    end

    err_pp_vals



end

function [t, y_D, y, err, err_pp] = PIcontrol_err(omega_0, gains)

    % calculate timespan required for 10 periods given an omega_0
    % (T = 2pi/omega)
    T = round(2*pi/omega_0);
    tspan = [0 10*T];

    % compute output
    [t, y] = ode45(@(t,y) PIcontroller(t, y, omega_0, gains), tspan, [2; 0]);
    y = y(:,1); % choose first column

    % compute error
    y_D = sin(omega_0*t);
    err = y - y_D;

    last_T_err = err(end-T:end);

    max_err = max(last_T_err);
    min_err = min(last_T_err);

    % compute error peak-peak amplitude over last period.
    err_pp = max_err - min_err;
end
function dydt = PIcontroller(t, y, omega_0, gains)
    m = 1;
    a = 0.1;
    omega = 0;
    y_D = sin(omega_0 * t);
    kp = gains(1);
    ki = gains(2);

    dydt=[1/m*(kp*(y_D-y(1))+ki*y(2)); y_D-y(1)];
end
```