Task 2

Analysis of piloted airplaine stability.

Contents

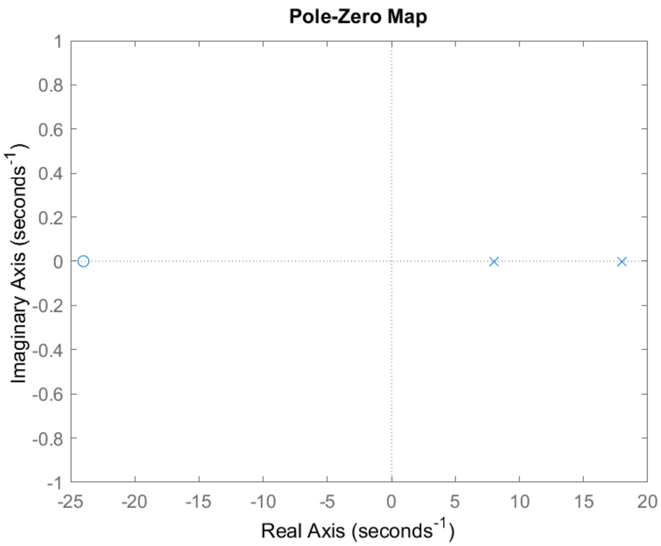
Transfer function

the transfer function, with canard deflection as input and pitch altitude as output, is given as:

$$\frac{\theta}{\delta_c} = \frac{s+24}{(s-8)(s-18)}$$

This system is clearly unstable, as both poles are positive real numbers (8 and 18). We can verify this by using pzplot and looking at the poles:

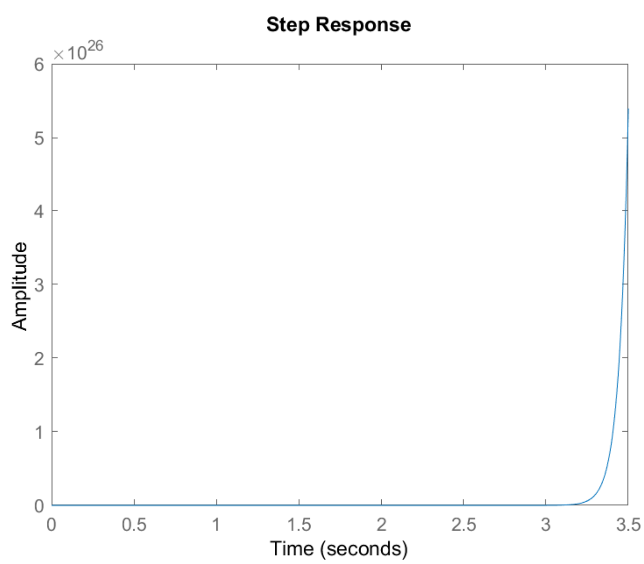
```
s = tf('s');
sys = (s+24)/((s-8)*(s-18));
pzplot(sys);
```



As we can see, both poles have positive real part, so the system must be unstable

Step response and stability

```
close;
step(sys);
```

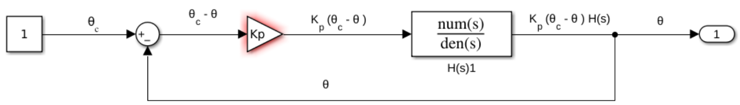


The output of our system blows up to infinity, so it is clearly unstable.

The open-loop system does not satisfy BIBO, and requires closed loop control to become stable.

Propotional control transfer function

We add a proportional control and find an equivalent transfer function for the whole system. I use the simulink block diagram to help find the expression for the new system:



$$\theta = K_p\theta_c H(s) - K_p\theta H(s)$$

$$\theta(K_p H(s) + 1) = \theta_c K_p H(s)$$

$$\frac{\theta}{\theta_c} = \frac{K_p H(s)}{K_p H(s) + 1}$$

I insert the plant model $H(s)$ to obtain the full closed loop transfer function:

$$\frac{\theta}{\theta_c} = \frac{K_p \frac{s+24}{s^2-26+144}}{K_p \frac{s+24}{s^2-26+144} + 1}$$

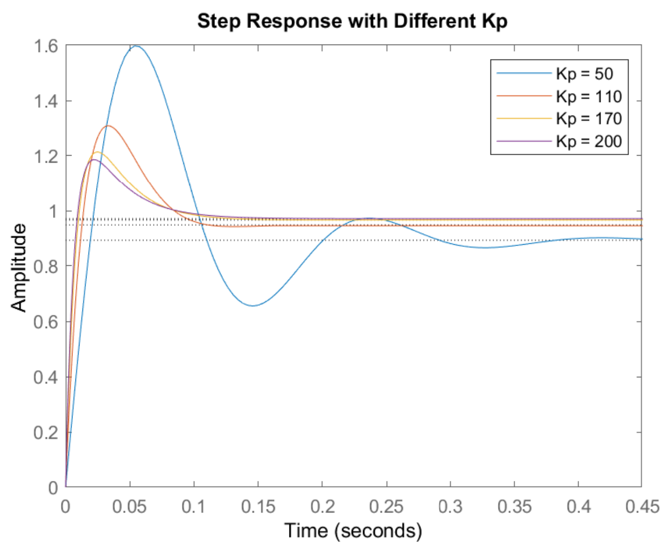
Finally, simplifying the fraction gives:

$$\frac{\theta}{\theta_c} = \frac{K_p s + K_p 24}{s^2 + (K_p - 26)s + (144 + 24K_p)}$$

Now that we have obtained the transfer function for our controlled system, We can plot the step responses for varying Kp's (proportional gains):

```
figure;
hold on;
for Kp = [50 110 170 200]
    closed_loop = (Kp*s+Kp*24)/(s^2+(Kp-26)*s+(144+24*Kp));
    step(closed_loop);
end

title('Step Response with Different Kp');
legend('Kp = 50', 'Kp = 110', 'Kp = 170', 'Kp = 200');
```



Increasing the proportional gain results in a shorter rise time, and a smaller steady state error. However, we cannot completely eliminate the steady state error completely, without introducing an integral term.

P-control stability

We can also plot the poles of this system as a function of the proportional gain K_p The poles are given by the roots of the polynomial in the denominator of our transfer function: We can use the quadratic formula to find the roots and plot $Re\{s\}$ and $Im\{s\}$ separately. We solve the equation:

$$s^2 + (Kp - 26)s + (144 + 24)Kp = 0$$

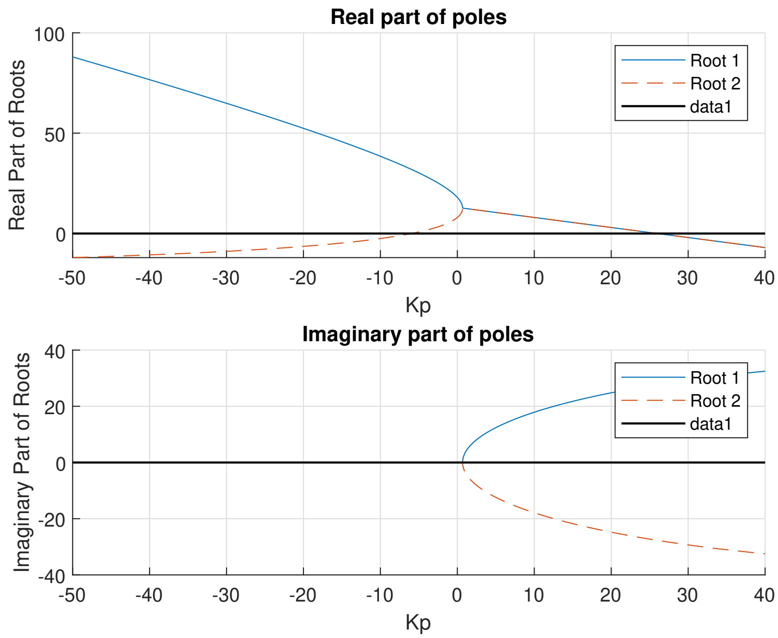
```
close;
```

```
Kp = -50:0.05:40; %range of Kp's to analyse
%find poles using the quadratic formula
s_1 = (-(Kp-26)+sqrt((Kp-26).^2-4*(144+24.*Kp)))/2;
s_2 = (-(Kp-26)-sqrt((Kp-26).^2-4*(144+24.*Kp)))/2;

figure;

subplot(2,1,1);
grid on;
hold on;
plot(Kp, real(s_1), 'DisplayName', 'Root 1', 'LineStyle', '-');
plot(Kp, real(s_2), 'DisplayName', 'Root 2', 'LineStyle', '--');
plot(Kp, zeros(size(Kp)), 'k', 'LineWidth', 1);  % x-axis
xlabel('Kp'); ylabel('Real Part of Roots');
title('Real part of poles');
legend;

subplot(2,1,2);
grid on;
hold on;
plot(Kp, imag(s_1), 'DisplayName', 'Root 1', 'LineStyle', '-');
plot(Kp, imag(s_2), 'DisplayName', 'Root 2', 'LineStyle', '--');
plot(Kp, zeros(size(Kp)), 'k', 'LineWidth', 1);  % x-axis
xlabel('Kp'); ylabel('Imaginary Part of Roots');
title('Imaginary part of poles');
legend;
```



Kp-stable values

From inspecting the plot we see that both poles/eigenvalues have negative real part when $K_p > 26$.

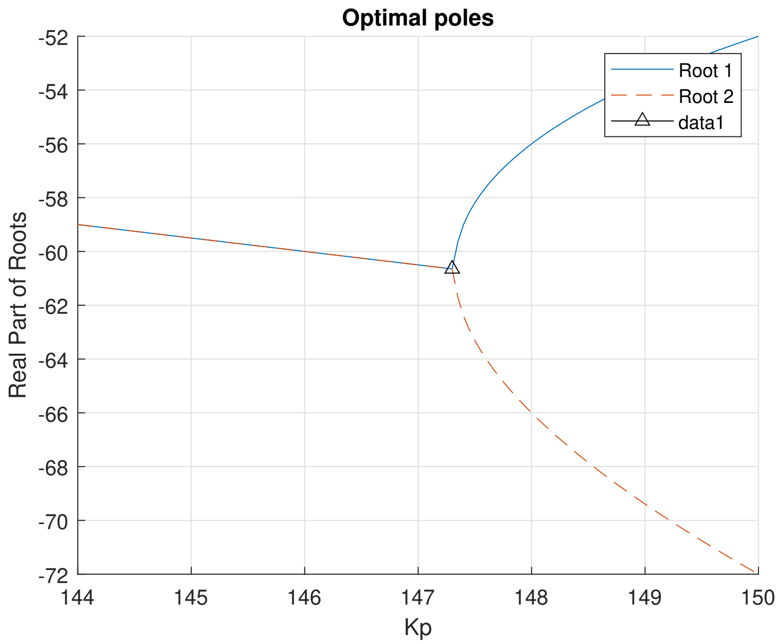
For the gain, i somewhat arbitrarily chose $Kp = 147.3$, which is where the poles start to separate again:

```
close;
figure;
grid on;
hold on;

Kp = 144:0.05:150;
s_1 = (-(Kp-26)+sqrt((Kp-26).^2-4*(144+24.*Kp)))/2;
s_2 = (-(Kp-26)-sqrt((Kp-26).^2-4*(144+24.*Kp)))/2;
plot(Kp, real(s_1), 'DisplayName', 'Root 1', 'LineStyle', '-');
plot(Kp, real(s_2), 'DisplayName', 'Root 2', 'LineStyle', '--');

Kp = 147.3;
plot(Kp, (-(Kp-26)+sqrt((Kp-26).^2-4*(144+24.*Kp)))/2, 'marker', '^', 'Color', 'k')
xlabel('Kp'); ylabel('Real Part of Roots');
title('Optimal poles');
legend;
```

```
Warning: Imaginary parts of complex X and/or Y arguments ignored.
```



We can also show this mathematically by finding the Kp values where all our poles are in the negative half-plane (negative real part). To do this we look we analyze the denominator of our transfer function with the quadratic formula.

equation:

$$s^2 + (K_p - 26)s + (144 + 24Kp) = 0$$

We need the real parts of the roots to both be negative. The roots are found by the quadratic equation:

$$\frac{-(K_p - 26) \pm \sqrt{(K_p - 26)^2 - 4(144 + 24K_p)}}{2}$$

The strictly real part is $\frac{26-K_p}{2}$. If the discriminant $(K_p - 26)^2 - 4(144 + 24K_p)$ is negative or zero, then the real part is just $\frac{26-K_p}{2}$, and our system is obviously stable for all $K_p > 26$. If the discriminant is positive however, the root will be strictly real, and we need to verify that they are still negative, by checking that

$$-(K_p - 26) > \sqrt{(K_p-26)^2 - 4(144+24K_p)}, \quad \forall K_p > 26$$

$$(-K_p + 26)^2 > (K_p - 26)^2 - 4(14 + 24K_p)$$

$$K_p{}^2 - 52K_p + 26^2 > K_p^2 - 52K_p + 26^2 - 576 - 96K_p$$

$$96K_p > 576$$

$$K_p > 6$$

This inequality will obviously also hold for $K_p > 26$. We have now shown that all the roots have negative real parts for all $K_p > 26$, so that our system is stable.

Verifying that $Kp = 26$ gives us zero-poles (marginally stable):

```
close;
Kp = 26;
closed_loop = (Kp*s+Kp*24)/(s^2+(Kp-26)*s+(144+24*Kp));
real(pole(closed_loop))
```

```
ans =

     0
     0
```

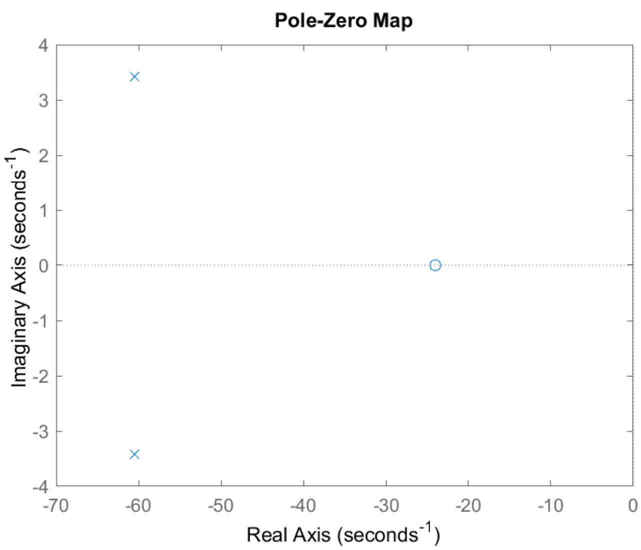
We can check the poles for our new controller with $Kp = 147.3$

```
Kp = 147;
closed_loop = (Kp*s+Kp*24)/(s^2+(Kp-26)*s+(144+24*Kp));
real(pole(closed_loop))
```

```
ans =

  -60.5000
  -60.5000
```

```
close;
pzplot(closed_loop);
```



Control system step response and steady-state error

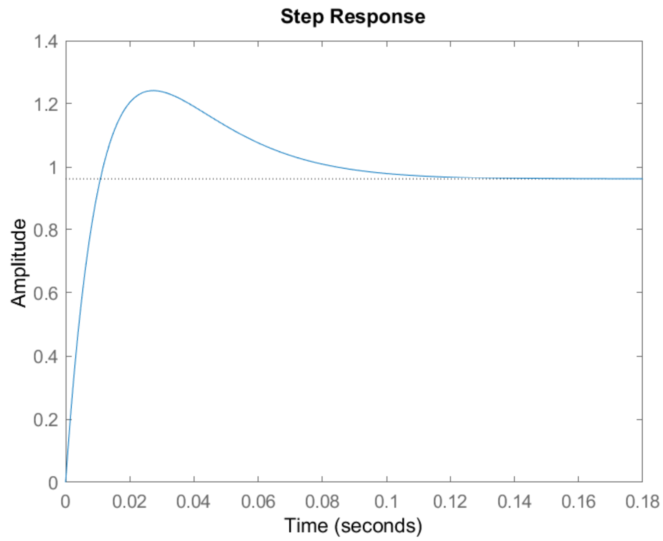
We have verified that the poles of our new system are all negative real part. This means the system stable, and we can verify this by viewing the step response of the closed loop system:

```
step(closed_loop)
[y,t]=step(closed_loop); %save the output values to check steady state
SS_error = abs(1-y(end));
%verifying that the new system is stable
isstable(closed_loop)
```

```
ans =

  logical

   1
```



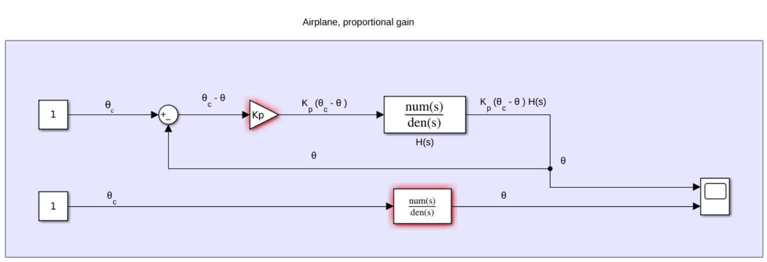
Thus we have verified that our control system is stable and reaches a steady state error of 3.89%, with a steady state value of

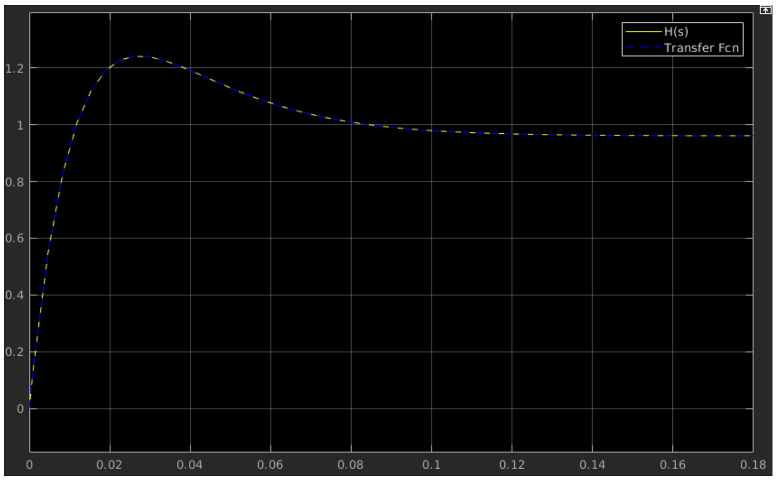
$$(1 - \frac{3.89}{100}) = 0.9611$$

We do have some overshoot here, and there is also a steady-state error in our system. The overshoot could be compensated for by introducing a derivative term, and the steady-state error could be eliminated by an integral term, giving us a full PID controller.

Simulink step response

Both the original simulink block diagram and the reduced transfer function show the same step response as the matlab code:





H(s)
Transfer Fcn