PID in the z-Domain

Objectives:

- Design a PID compensator in the z-domain.

Discussion:

The purpose of a PID compensator is to remove the steady-state error for a step input (I) and speed up the system (PD). For discrete-time systems, the D stands for 'delay' not 'derivative' however.

$$K(z) = P + I\left(\frac{z}{z-1}\right) + D\left(\frac{1}{z}\right)$$

Example: Design a P, PI, and PID compensator for G(s) that results in 20% overshoot in the step response. Assume a sampling rate of 50ms.

$$G(s) = \left(\frac{1000}{(s+2)(s+4)(s+6)(s+8)}\right)$$

First, find G(z). Assume a sampling rate of 0.05 second.

Convert the poles in the s-plane to the z-plane:

s = -2	z = 0.9048
s = -4	z = 0.8187
s = -6	z = 0.7408
s = -8	z = 0.6703

so

$$G(z) \approx \left(\frac{kz^2}{(z-0.9048)(z-0.8187)(z-0.7408)(z-0.6703)}\right)$$

Matching the DC gain:

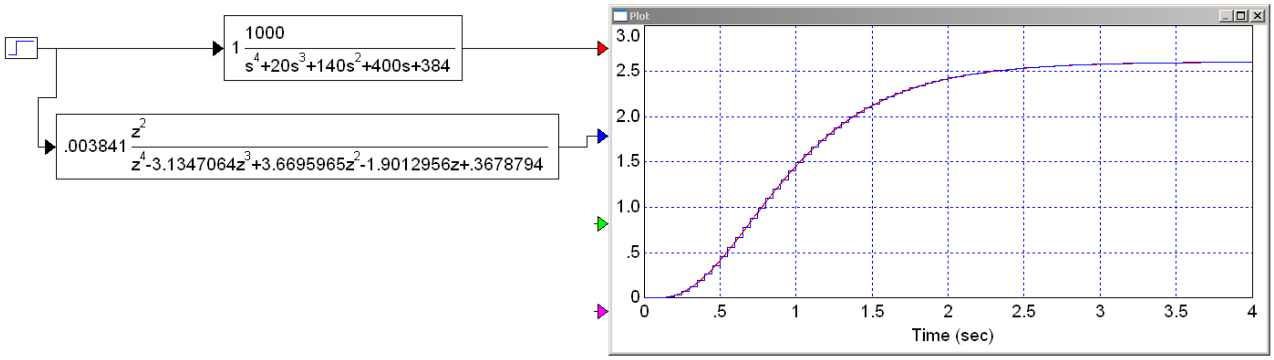
$$G(s=0) = 2.6042$$

To make

G(z=1) = 2.6042

k = 0.003841

$$G(z) \approx \left(\frac{0.003841z^2}{(z-0.9048)(z-0.8187)(z-0.7408)(z-0.6703)}\right)$$



To get the delay right, you can play with the numerator. A delay of two (z^2) in the numerator worked best.

P Compensation: K(z) = P = k.

Find the feedback gain, k, that results in 20% overshoot in the step response.

There are two ways to do this:

- You can analyze the system in the z-domain
- You can analyze the system in a hybrid domain (s and z) and avoid having to convert G(s) to G(z).

Method #1: Analyze the system in the z-plane.

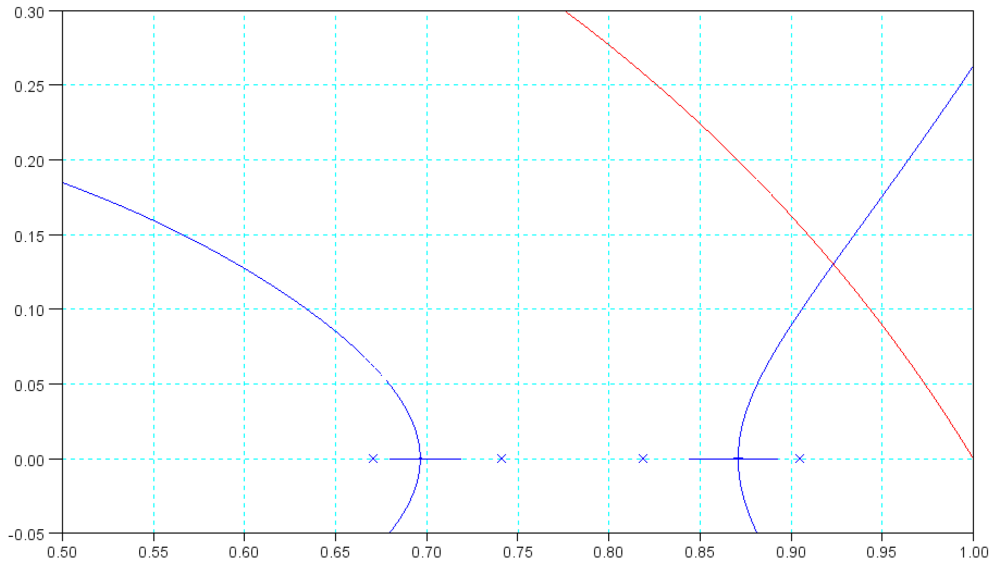
$$G(z) \approx \left(\frac{0.003841 z^2}{(z-0.9048)(z-0.8187)(z-0.7408)(z-0.6703)} \right)$$

Sketch the root locus and find the point which intersects the damping ratio of 0.4559:

```
G = zpk([0,0],[0.9048,0.8187,0.7408,0.6703],0.003841);
k = logspace(-2,2,1000)';
R = rlocus(G,k);

% add in the damping line

s = [0:0.01:10]' * (-1+j*2);
T = 0.05;
z = exp(s*T);
plot(real(z),imag(z),'r');
```



Root Locus of G(z) with K(z) = k

This gives

$$z = 0.9224 + j0.1289$$

and

$$G(z) = -2.4547k = -1$$

$$k = 0.4047$$

Method #2: Model the sample and hold with a 1/2 sample delay:

$$G(s) = \left(\frac{1000}{(s+2)(s+4)(s+6)(s+8)}\right)\left(e^{-0.025s}\right)$$

Search along the damping ratio of 0.4559

$$s = \alpha\angle 117.1229^0$$

Iterate by adjusting α until the angle of G(s) add up to 180 degrees. This is (solved numerically):

$$s = -1.3887 + j2.7111$$

This corresponds to the point in the z-plane (from $z = e^{sT}$)

$$z = 0.9244 + j0.1261$$

At any point on the root locus, GK = -1

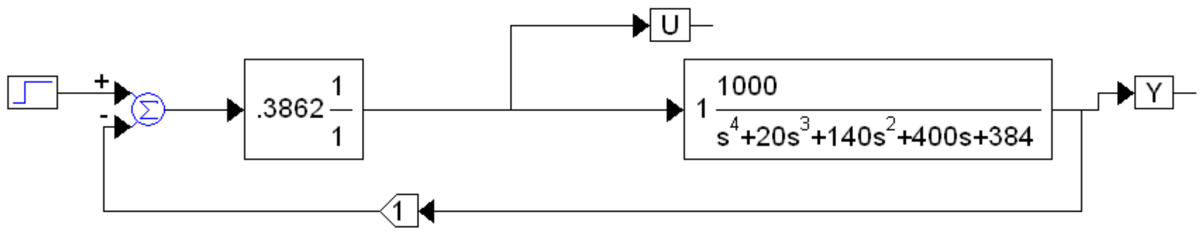
$$G(s) = -2.5892$$

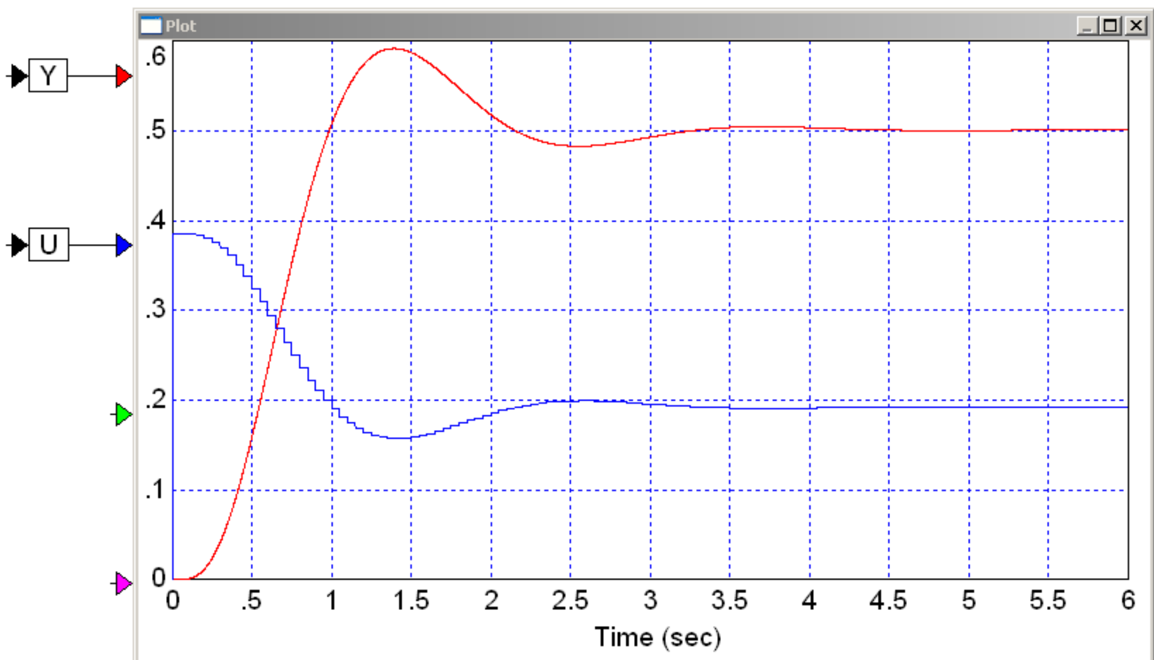
so

$$k = 0.3862$$

Note that the resulting pole location in the z-plane and the resulting gain, k, is almost what you got with the first method. The second method is a little more accurate since it doesn't depend upon any s to z conversions.

5b) Verify your design in VisSim.





Step Resonse with Proportional Control

Note that there is steady-state error. This is expected since this is a type-0 system.

PI Compensation: $K(z) = k\left(\frac{z-a}{z-1}\right)$

To make the steady-state error zero, add a pole at s = 0 (z = 1). Since you're adding a pole, you can also add a zero for free. This results in a PI compensator. Designing it for

- T = 0.05
- 20% overshoot

is as follows:

Method #1: Design in the z-plane.

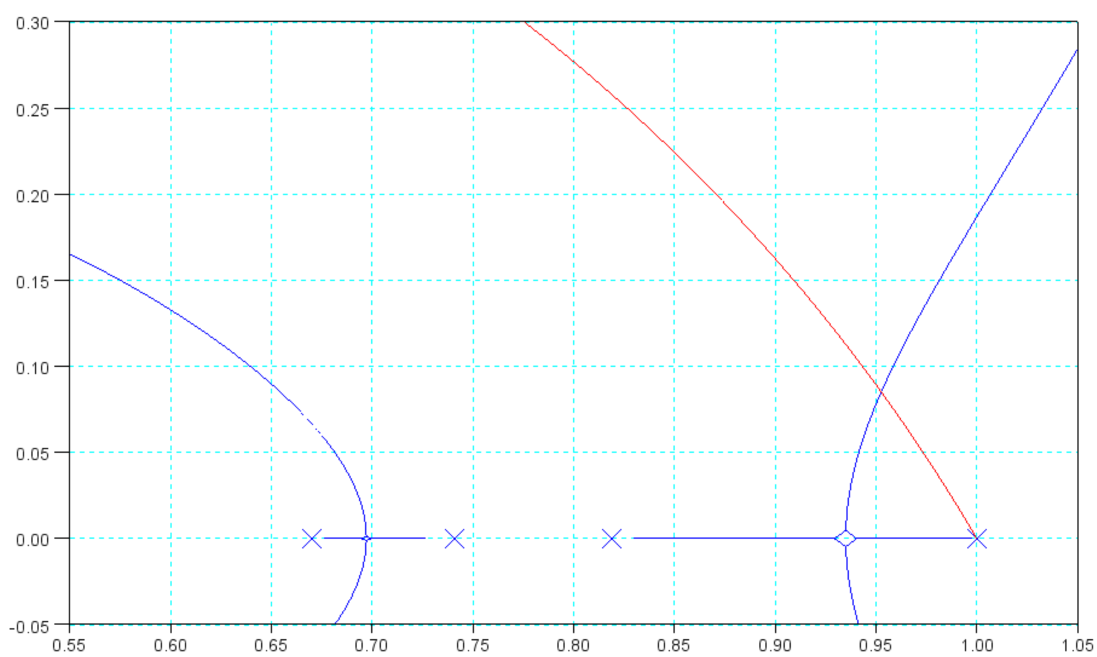
$$G(z) \approx \left(\frac{0.003841z^2}{(z-0.9048)(z-0.8187)(z-0.7408)(z-0.6703)}\right)$$

Make this a type 1 system and cancel the slowest stable pole

$$K(z) = k\left(\frac{z-0.9048}{z-1}\right)$$

Sketch the resulting root locus:

```
G = zpk([0,0],[1,0.8187,0.7408,0.6703],0.003841);
k = logspace(-2,2,1000)';
R = rlocus(G,k,0.4559);
// damping lines from before
plot(real(z),imag(z),'r');
```



Root Locus with a PI Compensator

Find the point which intersects the damping ratio of 0.4559:

$$z = 0.9522 + j0.0840$$

This gives

G(z) = -3.4289

k = 0.2908

$$K(z) = 0.2908\left(\frac{z-0.9048}{z-1}\right)$$

Method #2: Model the sample and hold as a 1/2 sample delay:

$$G(s) = \left(\frac{1000}{(s+2)(s+4)(s+6)(s+8)}\right)(e^{-0.025s})$$

Add K(z) to cancel the pole at s=-2 and add a pole at s=0:

$$K(z) = k\left(\frac{z-0.9048}{z-1}\right)$$

Search in the s (and corresponding z) plane along the damping ratio of 0.4559 until the angles add up to 180 degrees:

s = 0.8734 + j1.7050

z = 0.9538 + j0.0815

At any point on the root locus, GK = -1

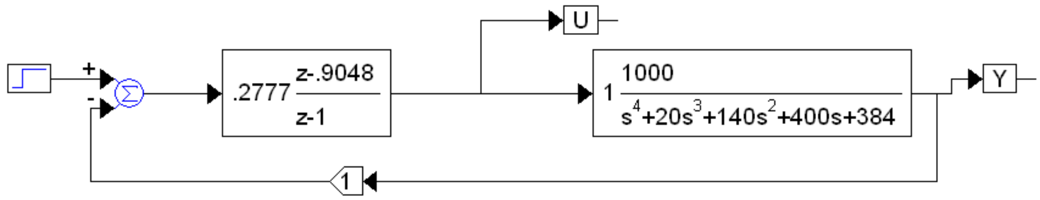
GK = -3.6004

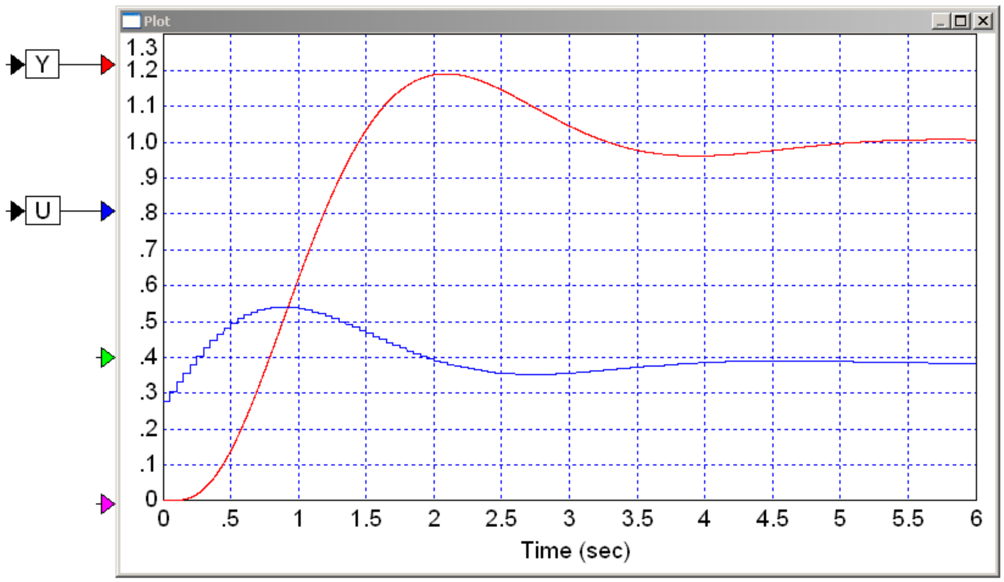
k = 0.2777

so

$$K(z) = 0.2777\left(\frac{z-0.9048}{z-1}\right)$$

Verify your design in VisSim.





Note again that the two methods give almost the same result. The latter method is a little more accurate, however, since it avoids the G(s) to G(z) conversion and it accounts for the 1/2 sample delay resulting from the sample and hold.

The difference between an I and PI compensator is

- You add a zero with a PI compensator, which allows you to cancel a pole and speed up the system, and
- The initial 'guess' for U isn't zero with a PI compensator. This speeds up the system as well.

PID Compensation: $K(z) = k\left(\frac{(z-a)(z-b)}{z(z-1)}\right)$

Design a PID compensator that results in 20% overshoot in the step response.

Method #1: Convert to the z-plane

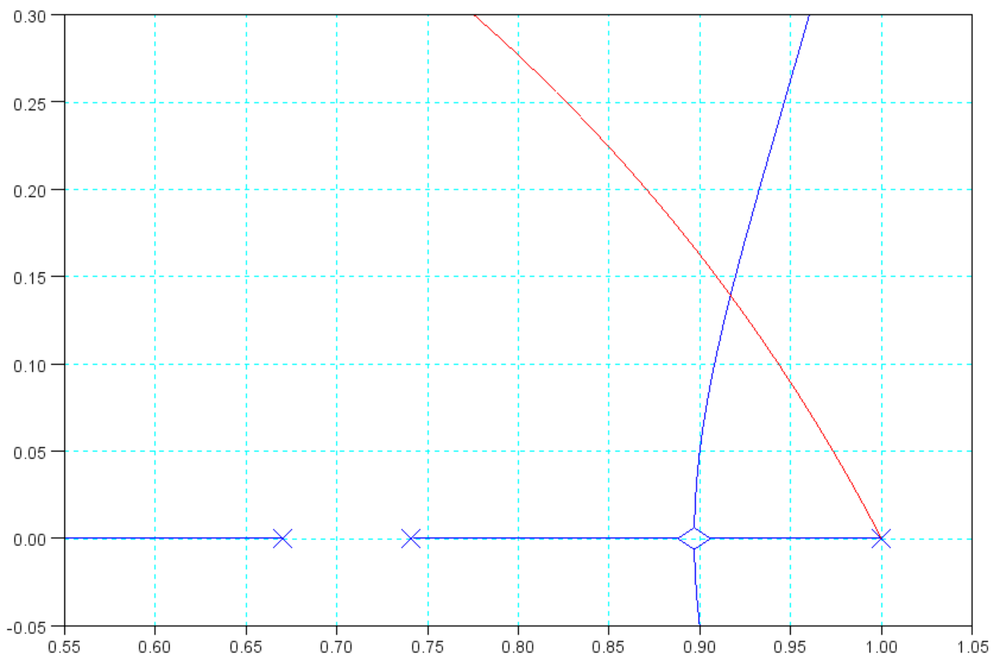
$$G(z) \approx \left(\frac{0.003841z^2}{(z-0.9048)(z-0.8187)(z-0.7408)(z-0.6703)}\right)$$

Cancel the two slowest poles. Replace them with a pole at z = +1 (to make it a type-1 system) and z pole at z=0 (to make it causal).

$$K(z) = k\left(\frac{(z-0.9048)(s-0.8187)}{z(z-1)}\right)$$

Sketch the resulting root locus:

```
G = zpk([0,0],[1,0,0.7408,0.6703],0.003841);
k = logspace(-2,2,1000)';
R = zlocus(G,k,0.4559);
// add damping line from before
plot(real(z),imag(z);
```



Root Locus with a PID compensator

Find the point on the root locus which intersects the damping ratio of 0.4559 curve:

$z = 0.9164 + j0.1373$

At this point

$GK = -0.3524k = -1$

k = 2.8381

and

$$K(z) = 2.8381\left(\frac{(z-0.9048)(s-0.8187)}{z(z-1)}\right)$$

Method #2: Model the sample and hold as a 1/2 sample delay

$$G(s) = \left(\frac{1000}{(s+2)(s+4)(s+6)(s+8)}\right)(e^{-0.025s})$$

Add a compensator, K(z), to cancel the slowest stable poles (or their corresponding spot in the z-plane)

$$K(z) = k\left(\frac{(z-0.9048)(s-0.8187)}{z(z-1)}\right)$$

Search along the damping ratio of 0.4559 until the angle of GK is 180 degrees

s = -1.4422 + j2.8155

z = 0.9212 + j0.1305

At this point, GK = -1

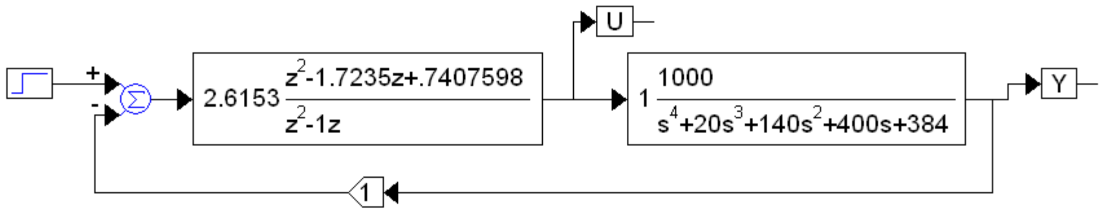
GK = -0.3824k = -1

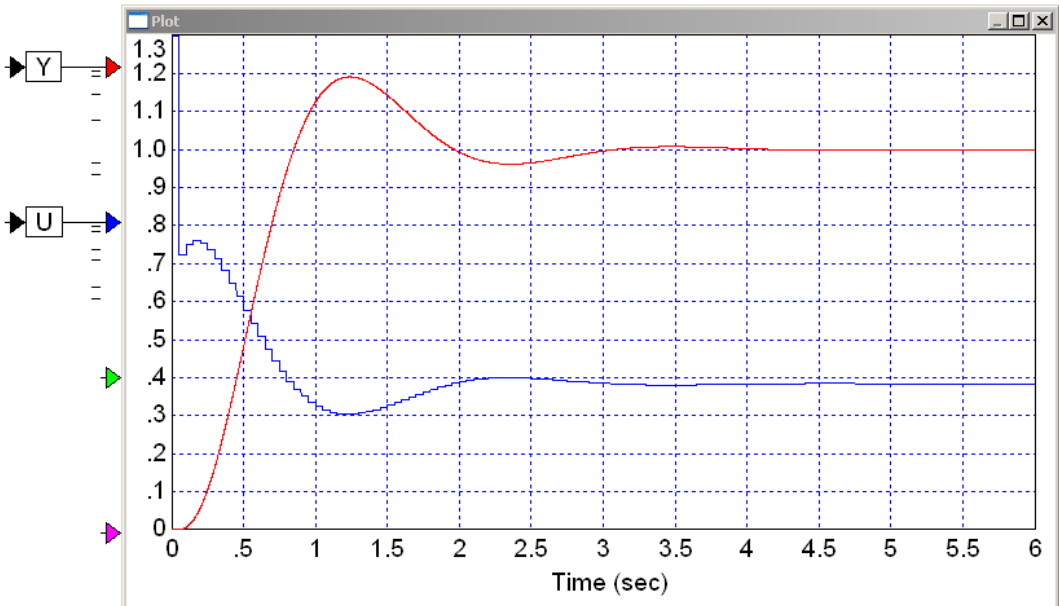
k = 2.6153

so

$$K(z) = 2.6153\left(\frac{(z-0.9048)(s-0.8187)}{z(z-1)}\right)$$

7b) Verify your design in VisSim or MATLAB.





Digital PID Compensator for G(s)

Note with a PID compensator

- You now have two zeros to use to cancel two poles
- This allows you to speed up the system significantly, but
- The initial input, U, is much larger. It starts out at 2.6153 (off the graph).

Part of the reason for the large spike at t=0 is the sampling rate is too small. The 2% settling time is about 3 seconds. With T = 0.05, this gives the controllers 60 samples to figure out the input - which is more than you really need. If you reduce this to 15 samples, meaning

$$T = 0.2$$

you get

$$G(s) = \left(\frac{1000}{(s+2)(s+4)(s+6)(s+8)}\right)$$

$$K(z) = k\left(\frac{(z-0.6703)(z-0.4493)}{z(z-1)}\right)$$

Searching along the line s = -a + ja until the angles of

$$angle(G(s)\cdot K(z)\cdot e^{-sT/2}) = 180^0$$

results in

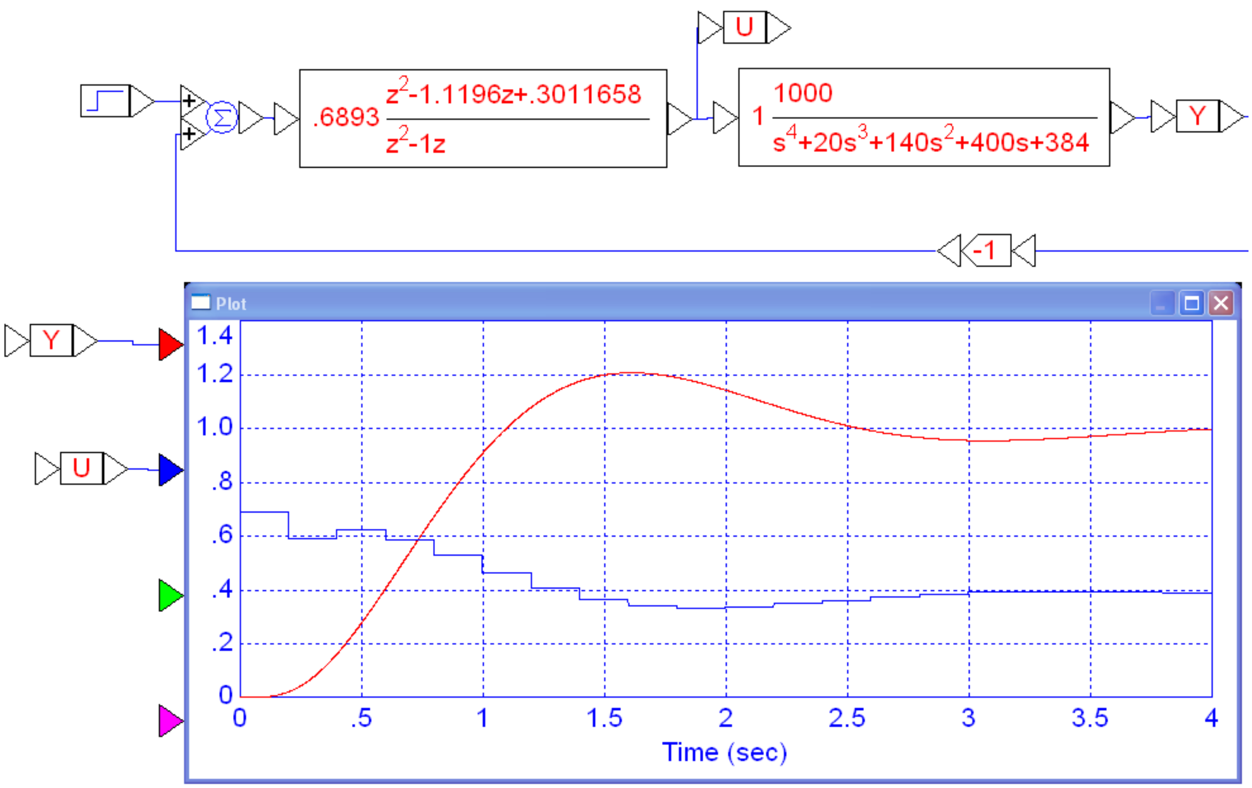
s = -1.0747 + j 2.1494

z = 0.7332 + j 0.3362

k = 0.6893

and

$$K(z) = 0.6893\left(\frac{(z-0.6703)(z-0.4493)}{z(z-1)}\right)$$



Step Resonse of a PID Compensator with T = 0.2 seconds

This is about the same response as we had before, only with

- A much more reasonable input at t = 0 (0.689 vs. 2.615)
- A much slower sampling rate (200ms vs 50ms)

Faster sampling rates are not always good. They can actually cause problems.