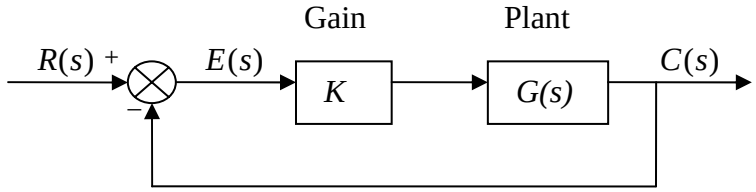
Example 1: Lag Compensator Design

A unit feedback system



with $KG(s) = \dfrac{K}{(s+1)(s+2)(s+10)}$ is operating with 25% overshoot ($\zeta = 0.4$).

(a) Design a compensator that will yield $e_{ss} = 1\%$.

(b) Use control toolkit to simulate the uncompensated and compensated systems.

Solution:

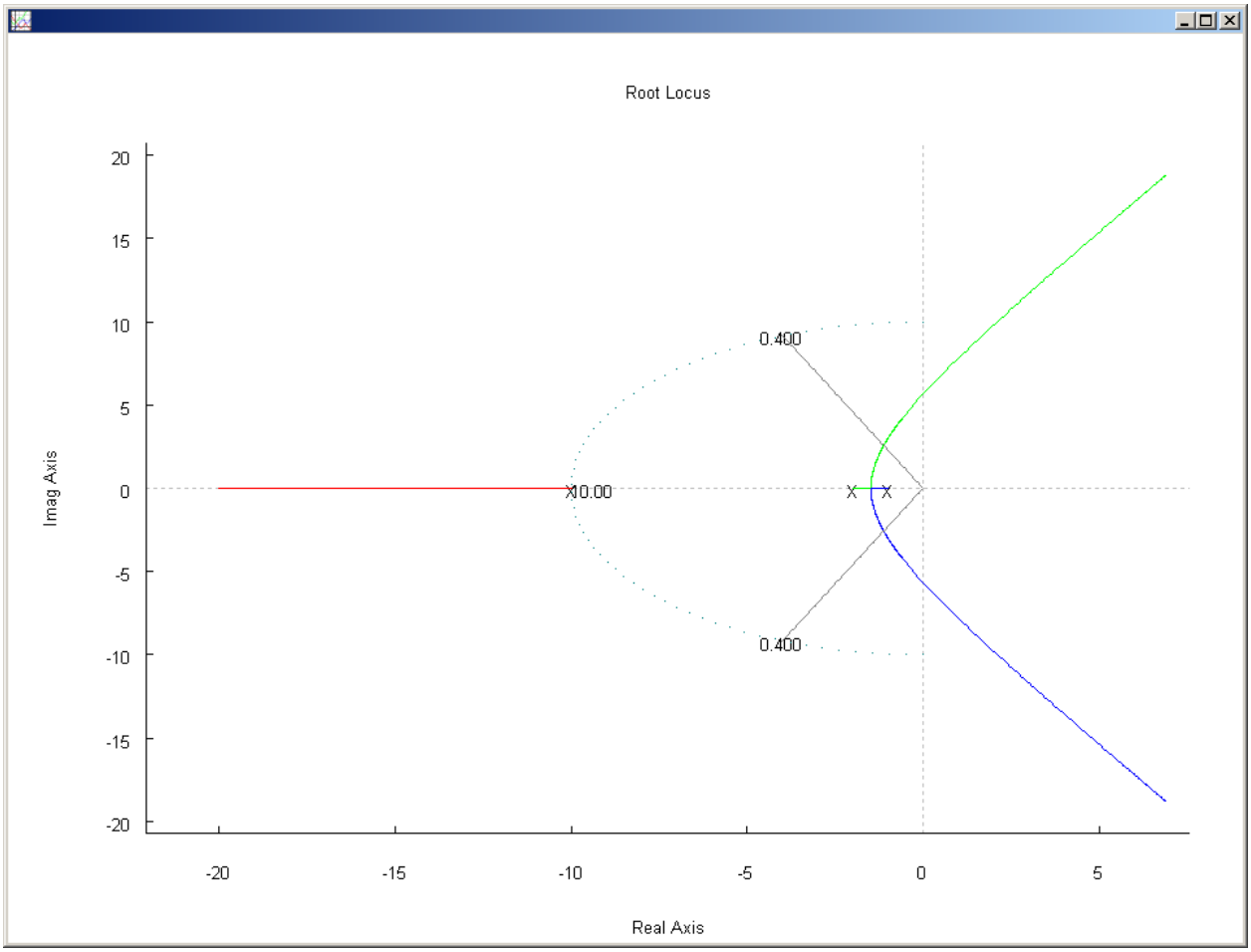
(a)

1. Use program 1 to plot the root locus of the ***uncompensated system***.

Program 1:

```
/*************************
 * File name: rl_uncomp.ch
 *************************/
#include <control.h>

int main() {
    double k = 1;
    array double complex p[3] = {complex(-1, 0),
                                 complex(-2, 0),
                                 complex(-10, 0)};
    array double zeta[1] = {0.4};
    array double omega[1] = {10};
    CControl sys;
    CPlot plot;

    sys.model("zpk", NULL, p, k);
    sys.sgrid(1, zeta, omega);
    sys.rlocus(&plot, NULL, NULL);

    return 0;
}
```

Output:



2. From the root locus, measure the real and imaginary parts of the dominant poles.

The dominant poles are found to be $-1.12 \pm i2.58$.

3. Use program 2 with one of the dominant poles to find the gain K and the third pole.

The gain K and the third pole are found to be 65 and -10.76, respectively.

Program 2:

```
/***************************
 * File name: rlf_uncomp.ch
 ***************************/

#include <control.h>

int main() {
    // default system gain
    double dk = 1;

    // uncompensated system poles
    array double complex up[3] = {complex(-1, 0),
                                  complex(-2, 0),
                                  complex(-10, 0)};

    // dominant pole selected from the root locus
    array double complex dp[1] = {complex(-1.12, 2.58)};

    // closed-loop poles when one of the dominant poles is selected
    array double complex p[3];

    // system gain when one of the dominant poles is selected
    array double k[1];

    CControl sys;

    sys.model("zpk", NULL, up, dk);
    sys.rlocfind(k, p, dp);
    printf("k: %f\n", k);
    printf("poles: %f\n", p);

    return 0;
}
```

Output:

```
k: 65.105433

poles: complex(-10.761278,0.000000) complex(-1.119361,2.579829)
complex(-1.119361,-2.579829)
```

4. Find the steady-state error of the uncompensated system.

Type 0, $e_{ss} = \dfrac{1}{1+K_p}$

$$K_{p_O} = \lim_{s\to 0} KG(s) = \frac{K}{1\times 2\times 10} = \frac{65}{20} = 3.25$$

$$e_{ss} = \frac{1}{1+K_{p_O}} = \frac{1}{1+3.25} = 0.24$$

5. Find $\dfrac{K_{p_N}}{K_{p_O}}$ that yields $e_{ss} = 0.01$.

$$K_{p_N} = \frac{1}{0.01} = 99$$

$$\frac{K_{p_N}}{K_{p_O}} = \frac{99}{3.25} = 30.46$$

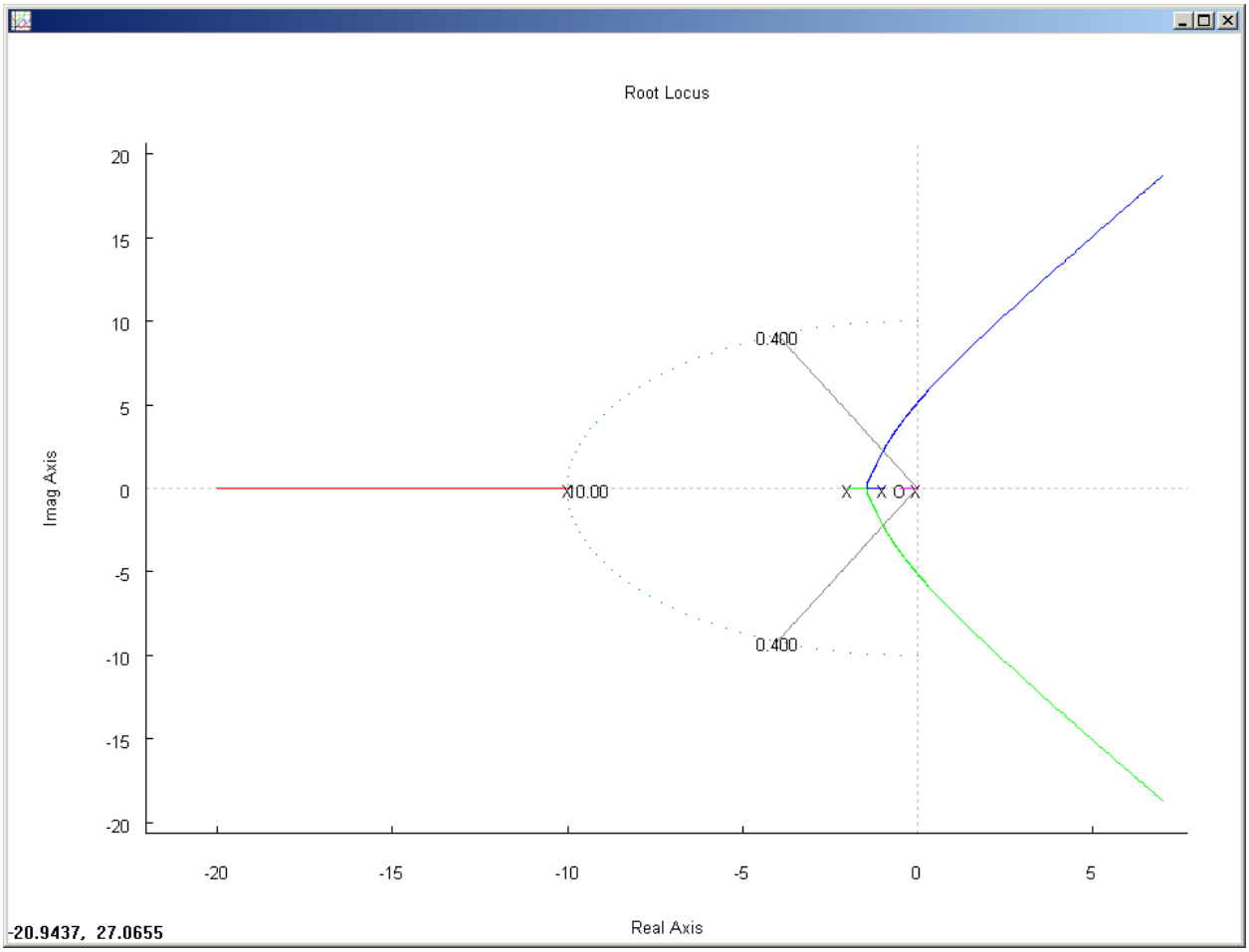
6. Select the compensator pole close to the origin. The compensator pole is chosen to be -0.015, which means $p_c = 0.015$. $z_c = \dfrac{K_{p_N}}{K_{p_O}} \times p_c = 30.46\times 0.015 \cong 0.5$, which means the compensator zero is -0.5.

7. Use program 3 to plot the root locus of the ***compensated system***.

Program 3:

```
/*************************
 * File name: rl_comp.ch
 *************************/

#include <control.h>

int main() {
    double k = 1;
    array double complex z[1] = {complex(-0.5, 0)};
    array double complex p[4] = {complex(-0.015, 0),
                                 complex(-1, 0),
                                 complex(-2, 0),
                                 complex(-10, 0)};
    array double zeta[1] = {0.4};
    array double omega[1] = {10};
    CControl sys;
    CPlot plot;

    sys.model("zpk", z, p, k);
    sys.sgrid(1, zeta, omega);
    sys.rlocus(&plot, NULL, NULL);

    return 0;
}
```

Output:



8. From the root locus, measure the real and imaginary parts of the dominant poles.

The dominant poles are found to be $-0.97 \pm i2.25$.

9. Use program 4 with one of the dominant poles to find the gain K. The gain K is found to be 55.

Program 4:

```
/**************************
 * File name: rlf_comp.ch
 **************************/

#include <control.h>

int main() {
    // default system gain
    double dk = 1;

    // compensated system zero
    array double complex cz[1] = {complex(-0.5, 0)};

    // compensated system poles
    array double complex cp[4] = {complex(-0.015, 0),
                                  complex(-1, 0),
                                  complex(-2, 0),
                                  complex(-10, 0)};

    // dominant pole selected from the root locus
    array double complex dp[1] = {complex(-0.97, 2.25)};

    // closed-loop poles when one of the dominant poles is selected
    array double complex p[4];

    // system gain when one of the dominant poles is selected
    array double k[1];

    CControl sys;

    sys.model("zpk", cz, cp, dk);
    sys.rlocfind(k, p, dp);
    printf("k: %f\n", k);
    printf("poles: %f\n", p);

    return 0;
}
```

Output:

```
k: 55.103584

poles: complex(-10.632421,0.000000) complex(-0.973519,2.250924)
complex(-0.973519,-2.250924) complex(-0.435541,0.000000)
```

10. Find the steady-state error of the compensated system.

Type 0, $e_{ss} = \dfrac{1}{1+K_p}$

$$K_{p_N} = \lim_{s\to 0} KG_{Lag}(s) = \frac{55\times 0.5}{1\times 2\times 10\times 0.015} = 91.67$$

$$e_{ss} = \frac{1}{1+K_{p_N}} = \frac{1}{1+91.67} \cong 0.01 = 1\%$$

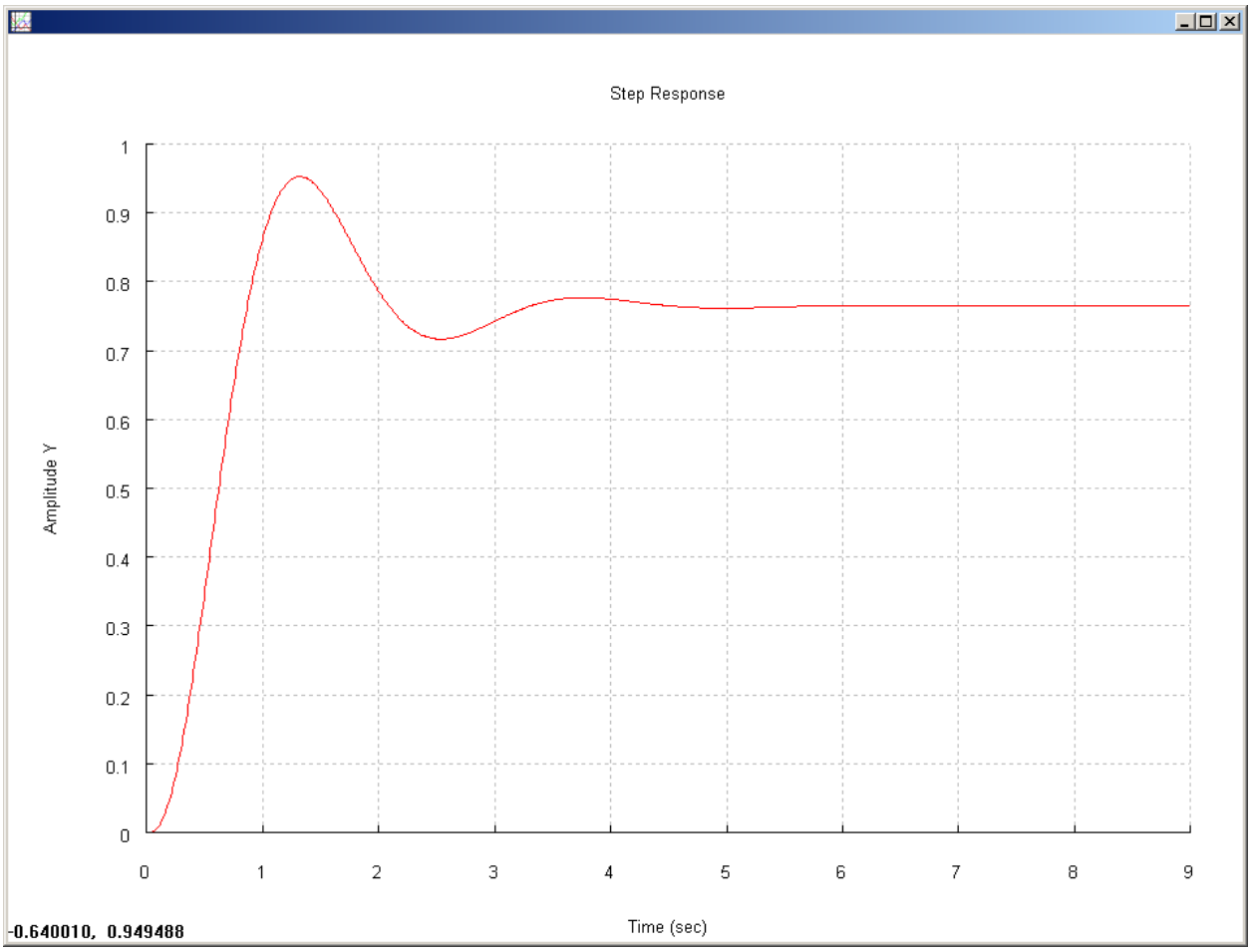
(b)

1. Use program 5 to obtain the step response of the ***uncompensated system***.

Program 5:

```
/*****************************
 * File name: step_uncomp.ch
 *****************************/

#include <control.h>

int main() {
    // system gain
    double k = 65;

    // uncompensated system poles
    array double complex up[3] = {complex(-1, 0),
                                  complex(-2, 0),
                                  complex(-10, 0)};
    array double num[1] = {1};
    array double den[1] = {1};
    double tf = 9;

    CControl sys1, sys2, *sys3;
    CPlot plot;

    sys1.model("zpk", NULL, up, k);
    sys2.model("tf", num, den);
    sys3 = sys1.feedback(&sys2);
    sys3->grid(1);
    sys3->step(&plot, NULL, NULL, NULL, tf);

    return 0;
}
```

Output:

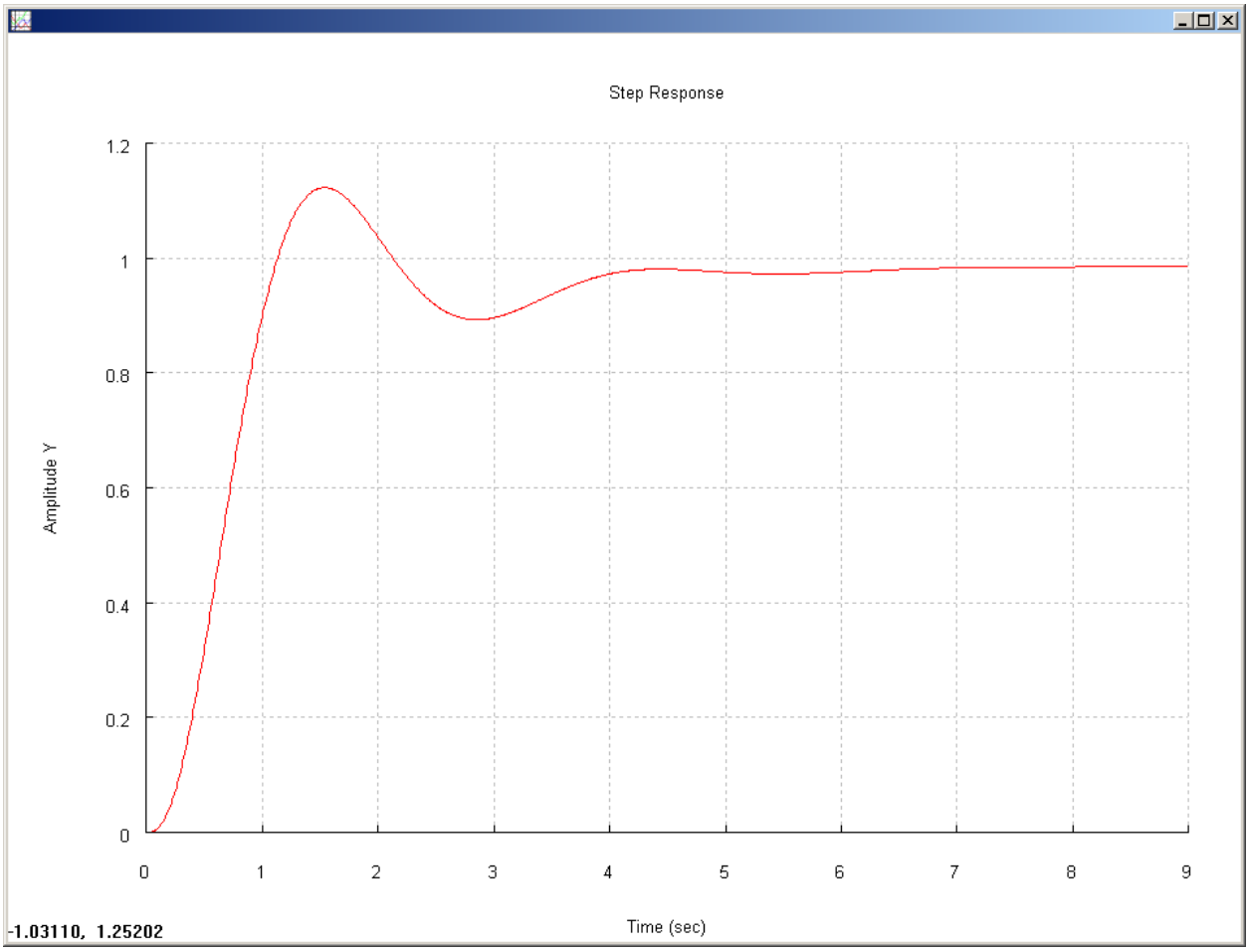

Step Response
Amplitude Y
Time (sec)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
1
2
3
4
5
6
7
8
9
-0.640010, 0.949488

2. Use program 6 to obtain the step response of the ***compensated system***.

Program 6:

```
/*****************************
 * File name: step_comp.ch
 *****************************/

#include <control.h>

int main() {
    // system gain
    double k = 55;

    // compensated system zero
    array double complex cz[1] = {complex(-0.5, 0)};

    // compensated system poles
    array double complex cp[4] = {complex(-0.015, 0),
                                  complex(-1, 0),
                                  complex(-2, 0),
                                  complex(-10, 0)};
    array double num[1] = {1};
    array double den[1] = {1};
    double tf = 9;

    CControl sys1, sys2, *sys3;
    CPlot plot;

    sys1.model("zpk", cz, cp, k);
    sys2.model("tf", num, den);
    sys3 = sys1.feedback(&sys2);
    sys3->grid(1);
    sys3->step(&plot, NULL, NULL, NULL, tf);

    return 0;
}
```

Output:


Step Response
Amplitude Y
Time (sec)
0
0.2
0.4
0.6
0.8
1
1.2
0
1
2
3
4
5
6
7
8
9
-1.03110, 1.25202