EE324 Control Systems Lab

Problem sheet 5

Tarun Sai Goddu | 190070024

Question 1

1.a We are given H_{closed} (s) the closed loop transfer function of a system with unity negative feedback. Thus, we find the open loop transfer function $G_{open}s$ as:

Gopen(s) = H(s)/1 - Hclosed(s) = 10 /s^3 + 4s^2 + 5s

Scilab Code for the same:

```
clear;
clc;
s = poly(0, 's');
num = 1;
den = (s^3+11*s^2+31*s+21);
G = num/den;
evans(syslin('c',G)); // constructing root locus
```

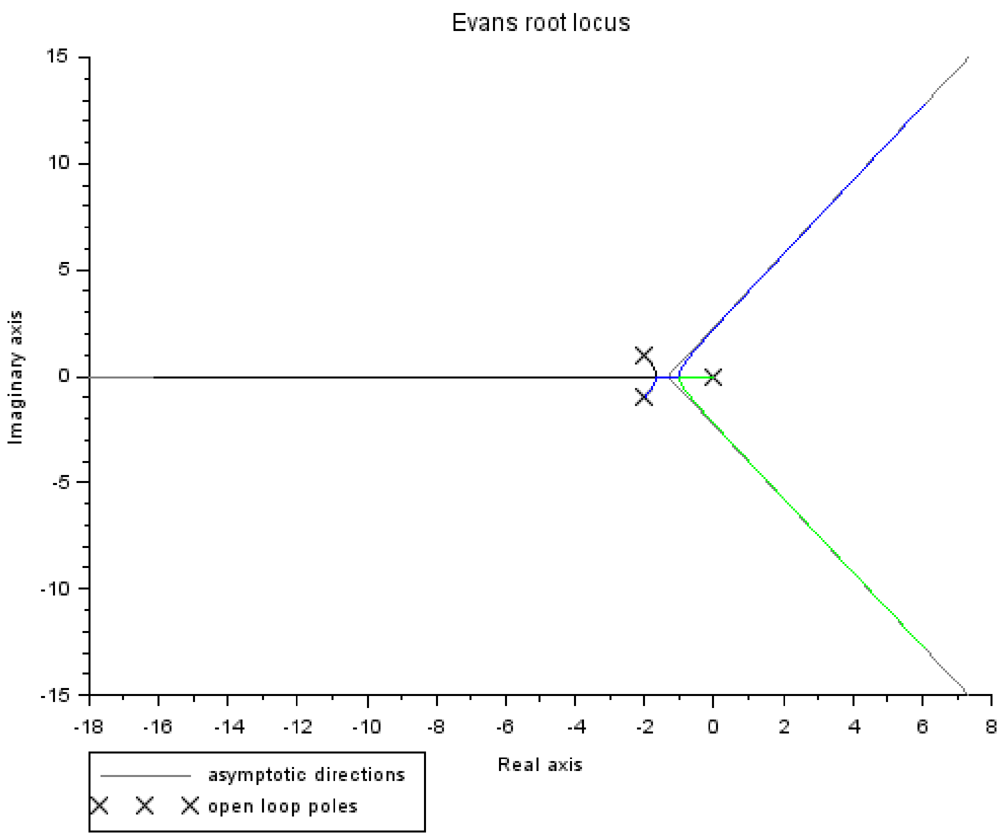

Figure 1: : Root-Locus for 1a

We observe there are 3 poles, and the system's poles lie in the LHP till about Critical gain = 2. Beyond that, the system becomes unstable.

1.b We are given $G_{open}(s)$ the open loop transfer function as:

$$G_{open}(s) = s + 1/s^2(s+3.6)$$

Scilab Code for the same:

```
clear;
clc;
s = poly(0, 's');
G = (s+1)/((s^2)*(s+3.6));
G = syslin('c',G);
poles = roots(G.den); //open loop poles
// evans finds root-locii of 1+K*G(s)
evans(G); // constructing root locus
```

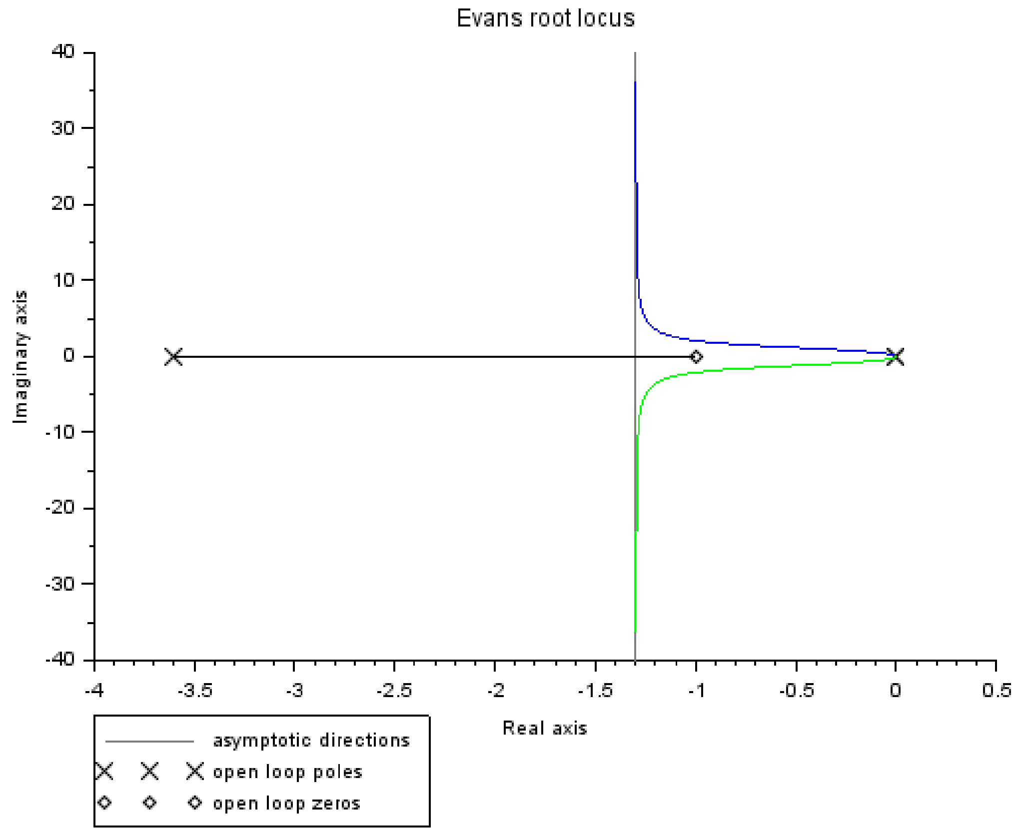

Figure 2: : Root-Locus for 1b

We observe there are 3 poles (including 1 pair of repeated poles at s = 0). We have 2 asymptotes (as the n_{poles} - n_{zeros}= 2)

1.c We are given G s the open loop transfer function as:

$$G_{open}(s) = s + 0.4/s^2(s+3.6)$$

Scilab Code for the same:

```
s = poly(0, 's');
G = (s+0.4)/((s^2)*(s+3.6));
G = syslin('c',G);
poles = roots(G.den); //open loop poles
evans(G,50); // constructing root locus
```

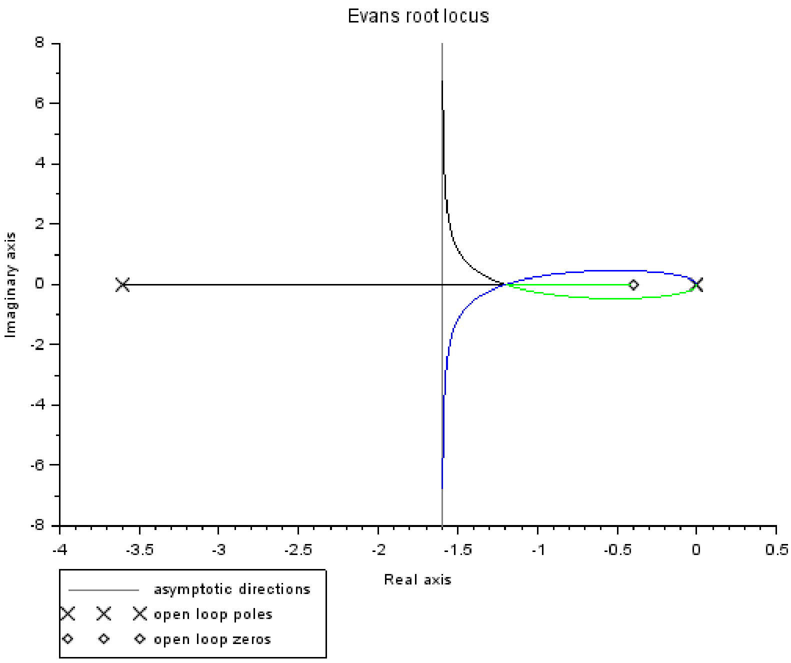

Figure3: Root Locus for 1c

1.d We are given G(s) the open loop transfer function as:

$$G_{open}(s) = s + p /s(s + 1)(s + 2)$$

The value of p was varied and the following observations were made: For $p < 0$, the system is bound to be unstable as one of the open loop poles will try to reach this open loop zero (lying in ORHP), implying a branch being in ORHP. Therefore, the system being unstable. For p = 0, 1 and 2, the system is stable due to the reduction to common factors cancelling out, and the system being reduced to a second order system with zeros at infinity and break-away point situated on negative real axis. One of the plots, for p = 0, is shown below:

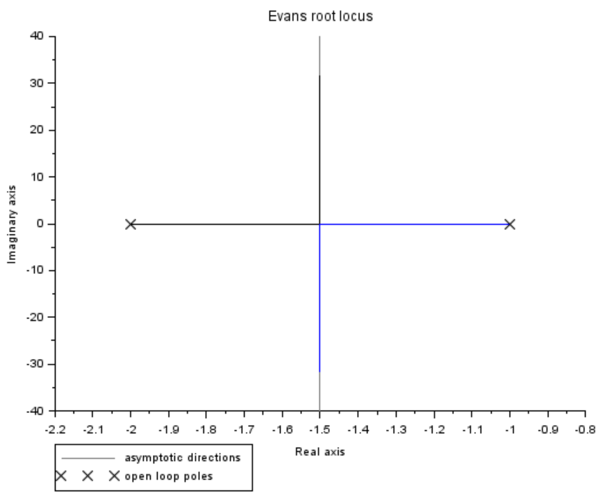

Figure 4: Root Locus for p = 0

For 0 < p < 2, I observed that the system was stable for all value of open loop gain (K>0). Root locus for one of the values (p = 1.5) is plotted below:

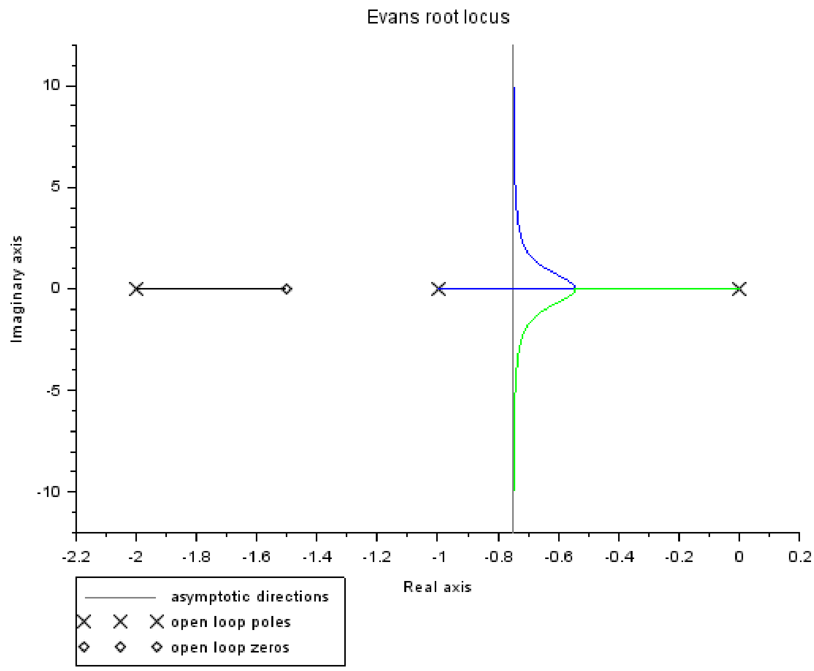

Figure 5: Root Locus for p = 1.5

For 2 < p < 3 as well the system was observed to be stable for all values of open loop gain (K>0). Root locus plot for p = 2.5 is given below:

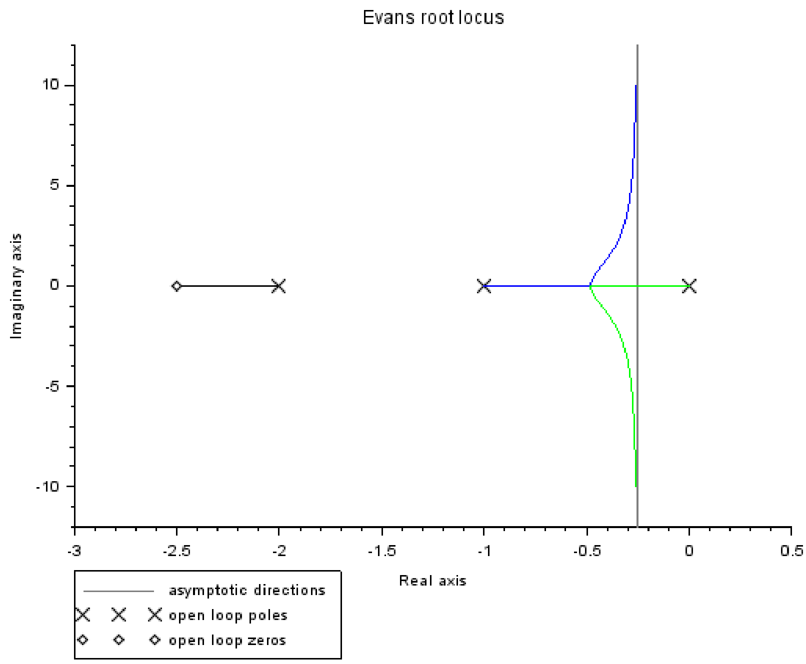

Figure 6: Root Locus for p = 2.5

For p = 3, the critical point is reached for system's stablilty as at p = 3, the asymptotes coincide with the imaginary axis, thereby the system being marginally stable as K tends to infinity. Root locus plot for p = 3, is shown below:

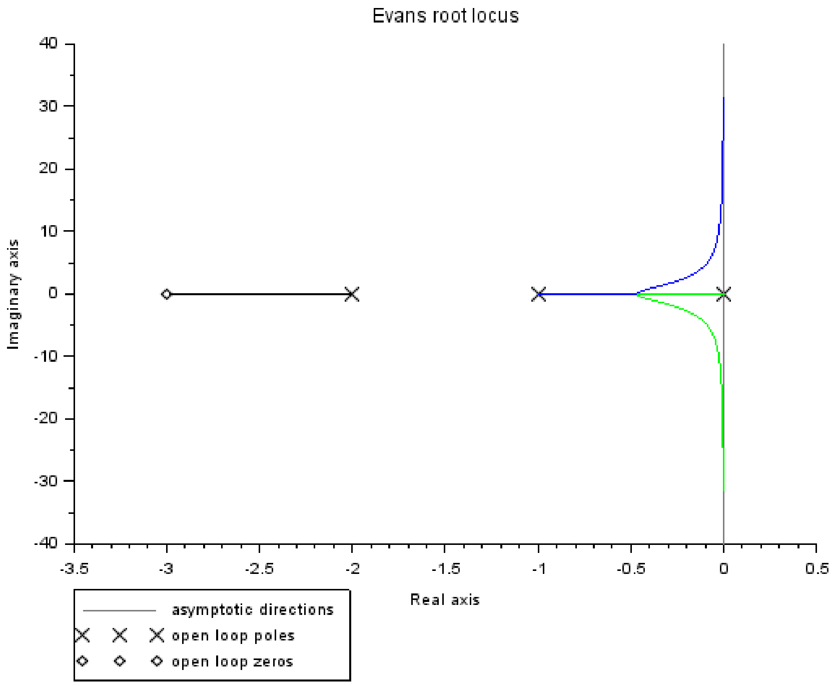

Figure 7: Root Locus for p = 3

For p > 3, we observe that the 2 assymptotes are situated in ORHP, implying the system being stable only for some values of open loop gain (K>0), and unstable for K > C, where C is a constant which varies with p. Root locus plot for p = 5 is shown below:

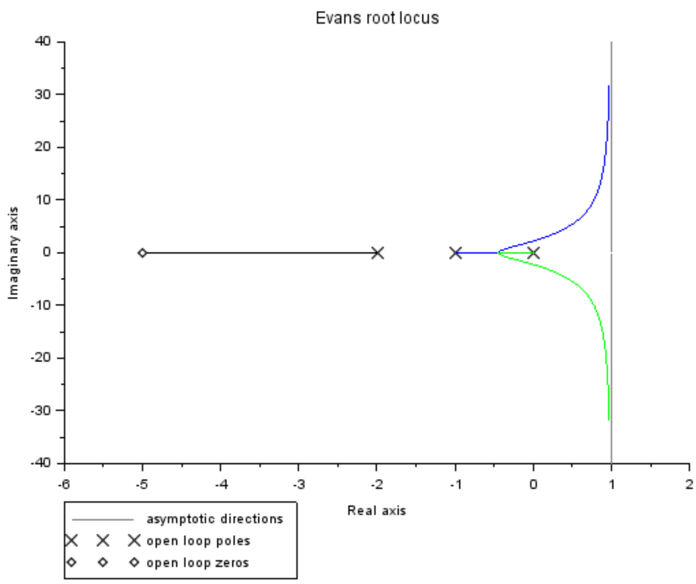

Figure 8: Root Locus for p = 5

Scilab Code for the same:

```
clear;
clc;
s = poly(0, 's');
p = 0;
G = (s+p)/(s*(s+1)*(s+2));
G = syslin('c',G);
poles = roots(G.den); //open loop poles
// evans finds root-locii of 1+K*G(s)
evans(G,1000); // constructing root locus
end
```

Question 2

2a We need the break-in and break-away points to coincide. So we take the specific case of poles placed on the real axis symmetrically about the origin, and the zeros placed symmetrically about the origin on the imaginary axis.

$$G_{open}(s) = s^2+4/ s^2 - 4$$

Scilab Code for the same:

```
clear;
clc;
s = poly(0, 's');
k = 0;
G = ((s+k)^2+4)/((s+k)^2-4);
G = syslin('c',G);
poles = roots(G.den); //open loop poles
// evans finds root-locii of 1+K*G(s)
evans(G); // constructing root locus
```

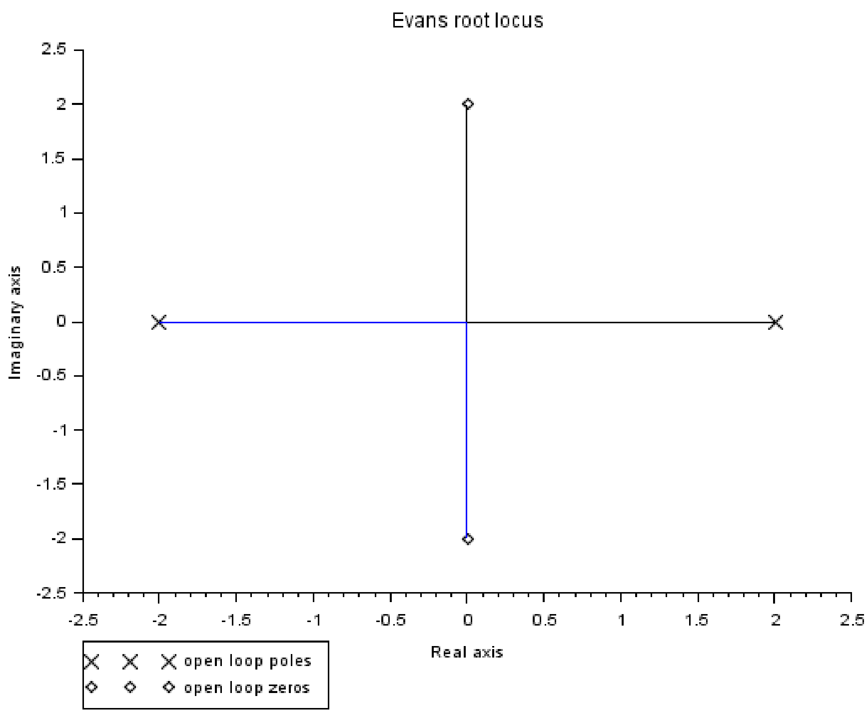

Figure 9: Root Locus for 2a

2b

We consider the case of 5 branches at the breakaway point (s=0):

Gopen(s) = s^2+ 1/ s^2- 1

Scilab Code:

```
clear;
clc;
s = poly(0, 's');
G = (s^5+1)/(s^5-1);
G = syslin('c',G);
poles = roots(G.den); //open loop poles
// evans finds root-locii of 1+K*G(s)
evans(G,100); // constructing root locus
```

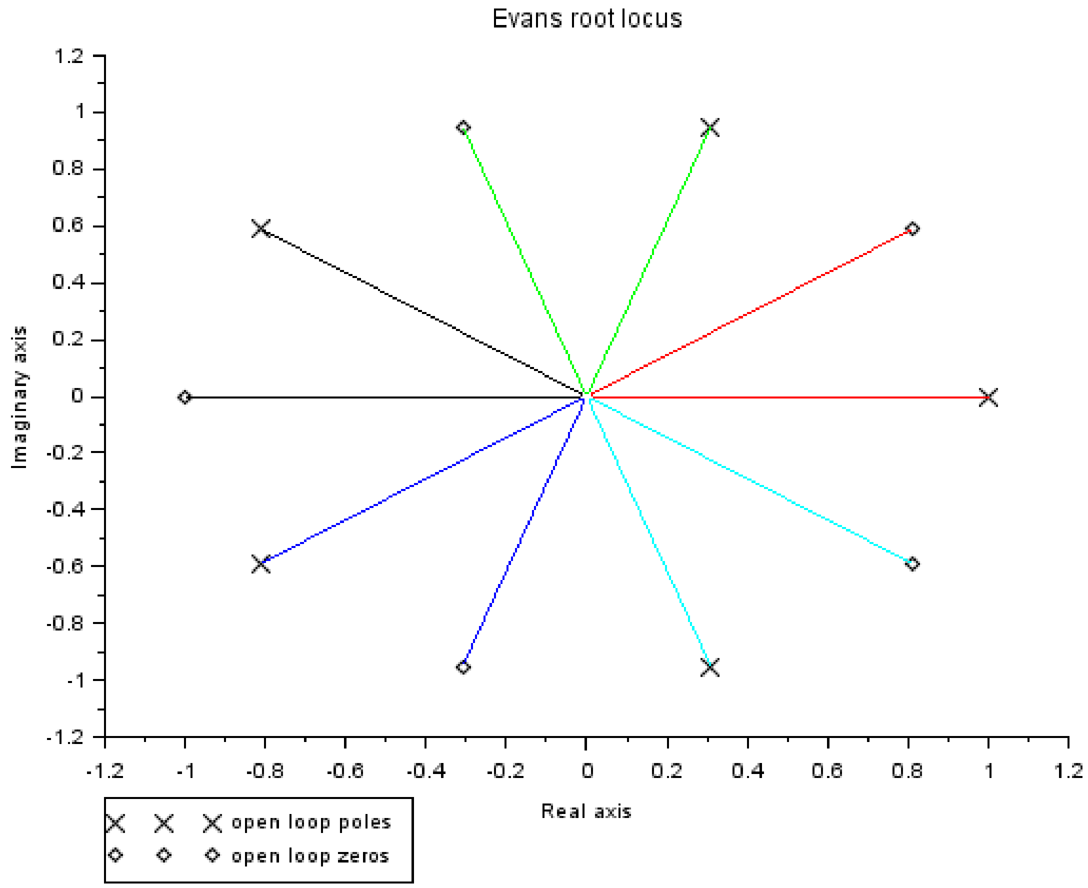

Figure 10: Root locus for 2b

By placing the poles at points that divide the circumference of the unit circle around the origin into 5 equal parts (and the zeros at five roots of unity as depicted by the numerator), we obtain the required condition. The point of coincidence (breakin and breakaway) is the origin.

2c

We can obtain a Root locus with the branches coinciding with the asymptotes by placing all the poles at the location where the asymptotes seem to meet, and the zeros at infinity (basically no zeros).

Gopen(s) = 1/(s + 1)^5

Scilab Code:

```
clear;
clc;
s = poly(0, 's');
G = 1/((s+1)^5);
G = syslin('c',G);
poles = roots(G.den); //open loop poles
// evans finds root-locii of 1+K*G(s)
evans(G,100); // constructing root locus
```

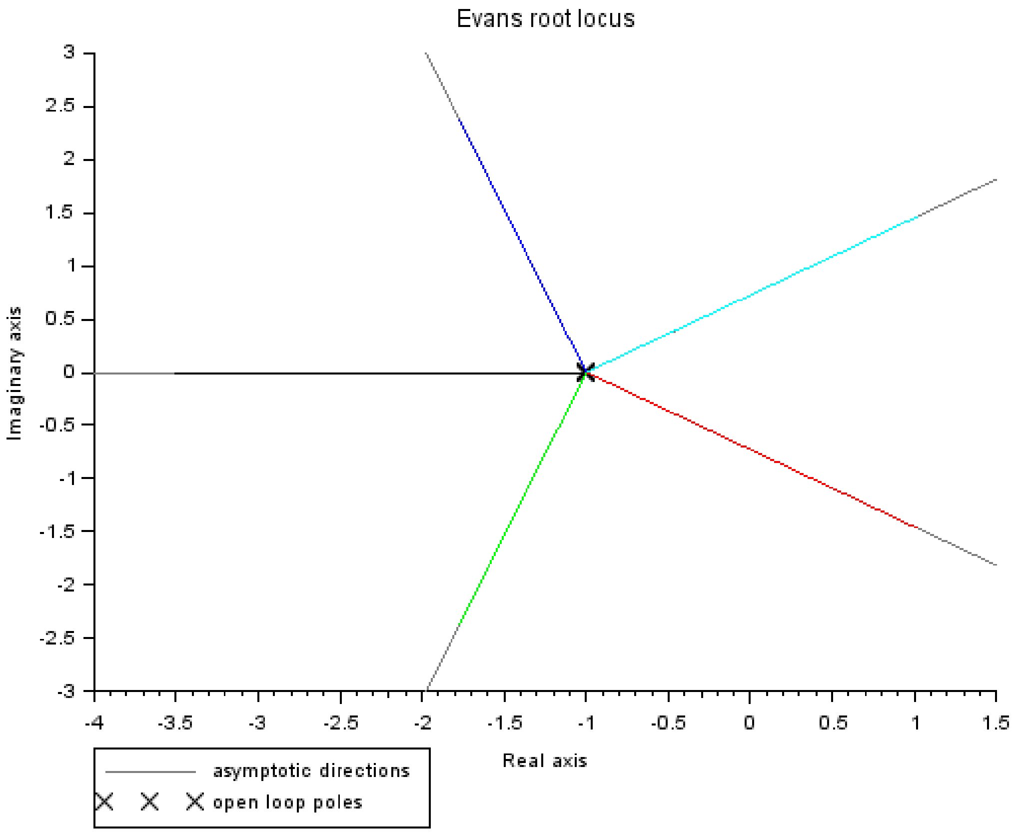

Figure 11: Root locus for 2c

The point of coincidence (breakin and breakaway) is the point s = -1.

2d

(i) We consider the transfer function G1 (s), with poles placed on the real axis, symmetrically about the jw axis:

G1 (s) = = 1/(s^2-1)(s^2-4) = 1/s^4-5s^2+4

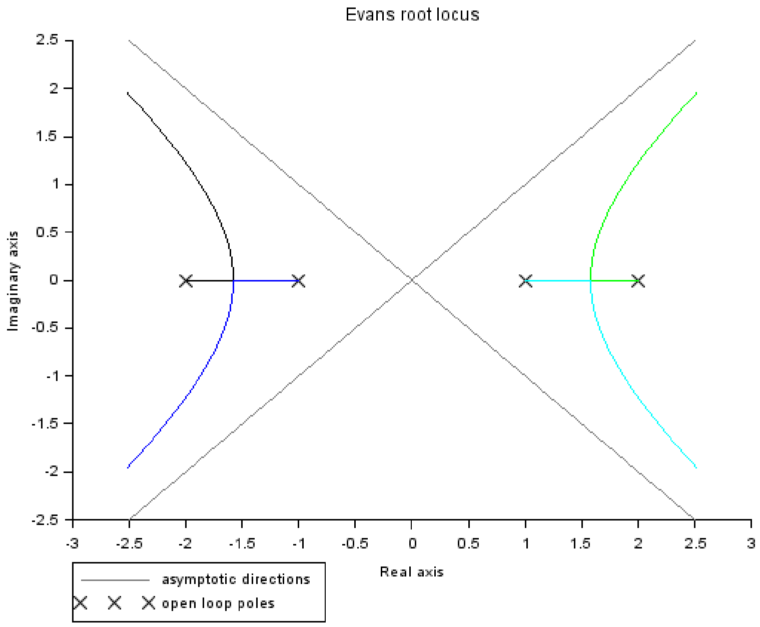

Figure 12: Root locus for G1 (s)

(ii) Obtain G s by substituting with s^2 with s^-2:

G2 (s) = 1/(s^2+1)(s^2+4)

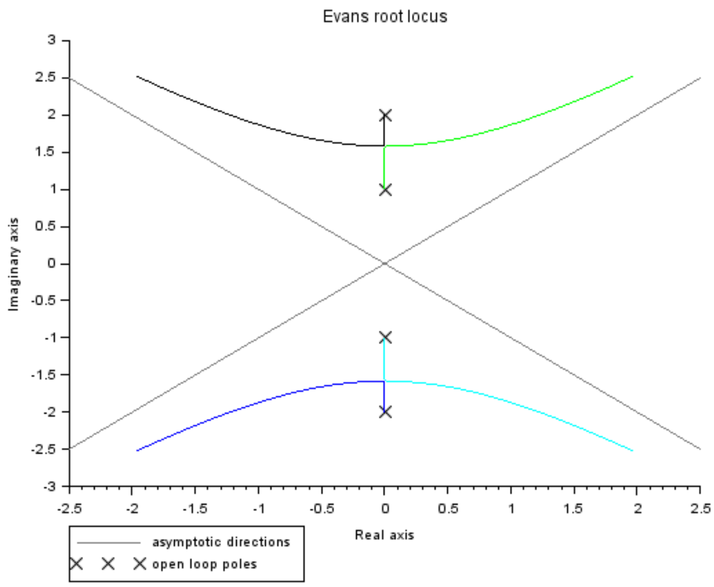

Figure 13: Root locus for G2 (s)

(iii) Finally replace s with s - 4:

G3 (s) = 1/((s-4)^2+1)((s-4)^2+4)

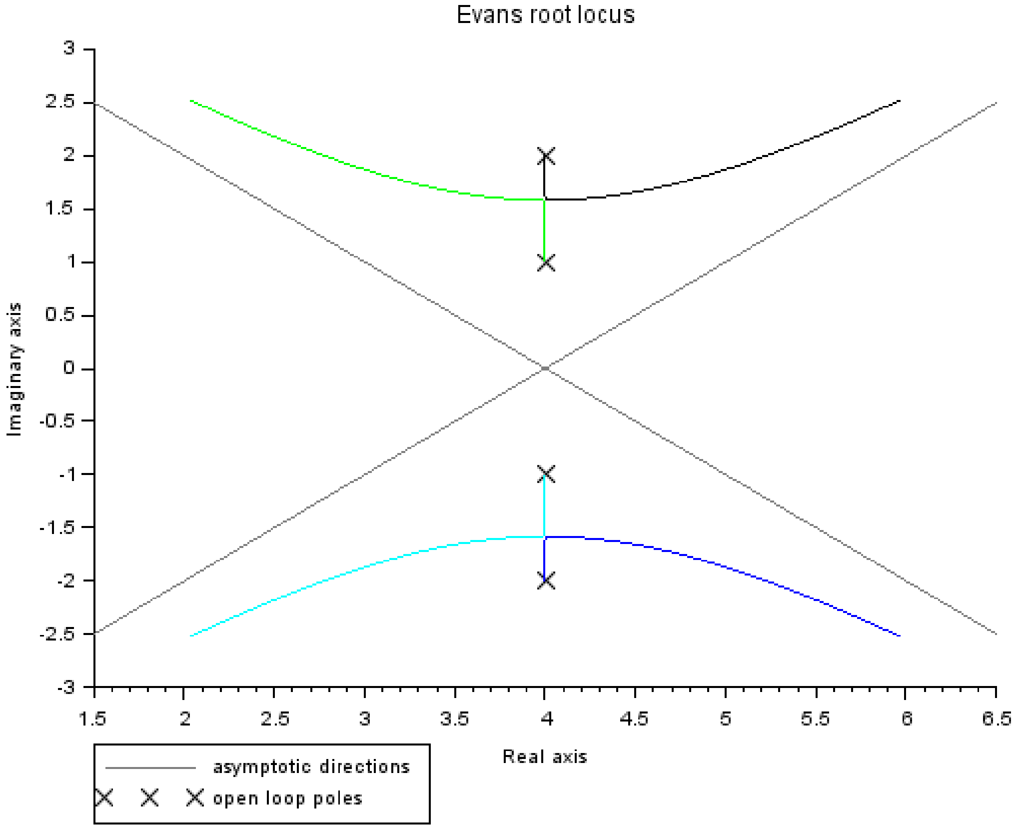

Figure 14: Root locus for G3 (s)

Scilab Code:

```
clear;
clc;
s = poly(0, 's');
G = 1/((s^2-1)*(s^2-4));
G = syslin('c',G);
poles = roots(G.den); //open loop poles
// evans finds root-locii of 1+K*G(s)
evans(G,100);  // constructing root locus
```

Question 3

G(s) = 1/s(s^2 + 3s + 5)

Scilab Code:

```
s=poly(0,'s');
G = syslin('c',1 / (s*(s^2 + 3*s +5)));
evans(G);

k_max = 15;
t=0:0.01:15;
kp = 0.1:0.1:k_max;
//kp=3.7:0.001:3.8;
for k1=kp
  G1 = (k1/(k1+s*(s^2 + 3*s +5)));
  G = syslin('c',G1);
  G_rep = csim('step',t,G);
  if(max(G_rep)>=0.9) then
    tr = t(find(G_rep > 0.9)(1))-t(find(G_rep > 0.1)(1));
    printf("k=%.3f , rise_time = %.3f\n",k1,tr);
  end;
end;
```

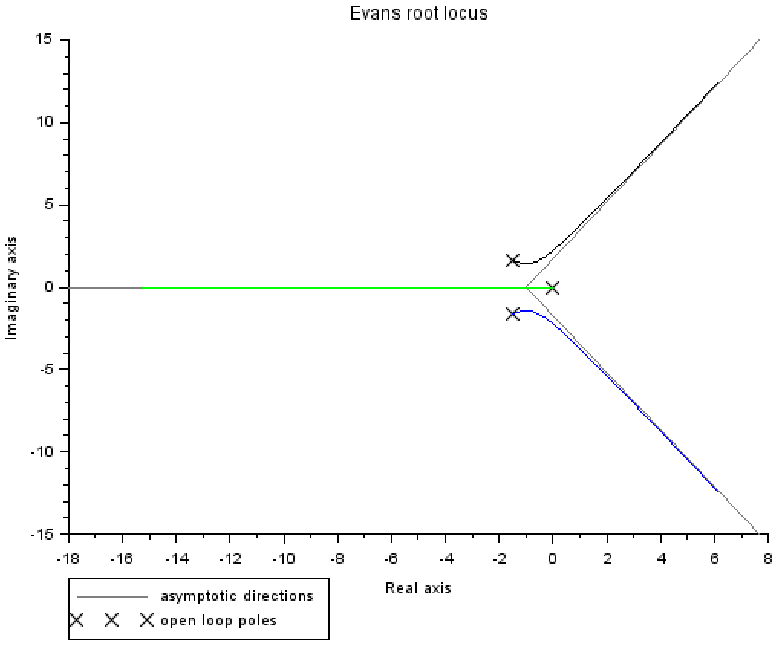

Figure 15: Root locus for G(s)

The system is stable for kp ⩽ 15, as was observed from the above RL plot. Upon plotting for values of kp from 0.1 to 15, and fine tuning for later simulations, kp = 3.7475 was found to be optimal value for rise time of 1.5s. For a stable system, the highest value of kp = 15 leads to lowest rise time (0.57s). If kp value is further increased, rise time further decreases.

Question 4

Open loop transfer function:

$$G(s) = 0.11\ (s + 0.6\ )/6s^2 + 3.6127s + 0.0572$$

The root locus plot for the G(s) as open loop transfer function is given below:

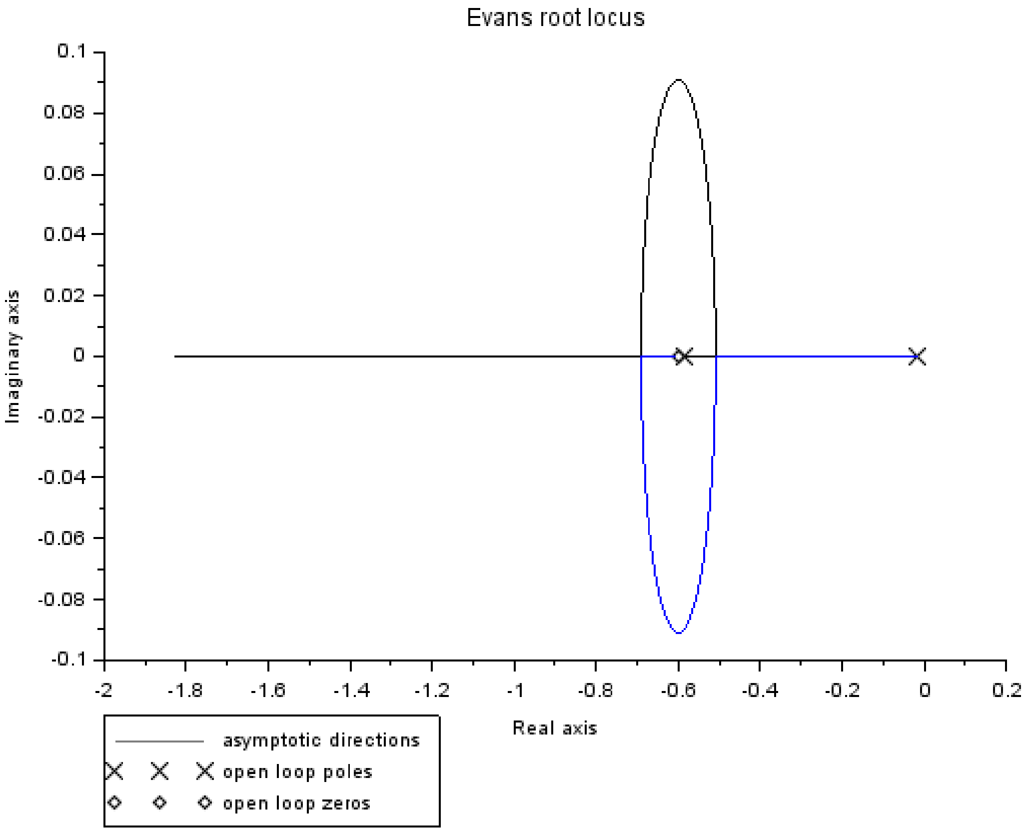

Figure 16: Root locus for kp > 0

For 1% steadystate error on step input ⟹ lim $1/1+K_pG(s) = 0.01$

$$Kp = 99*0.0572/0.11*0.6 = 85.8$$

Using Scilab, kp = 85.8 gave steady state error = 1%. The unit response is shown below:

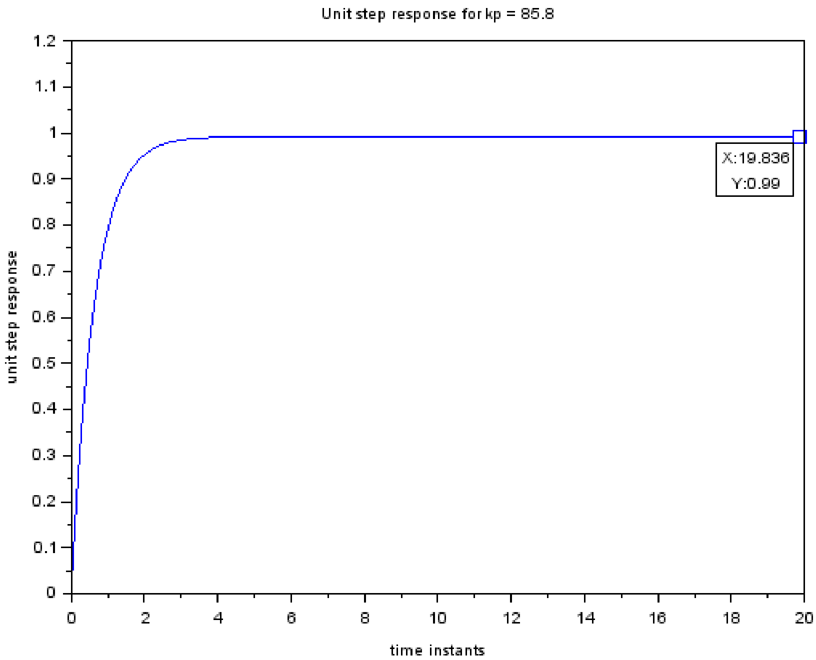

Figure 17: Unit step response for kp = 85.8

For kp > 0, system is stable as depicted by evans plot. For kp < 0, i just multiplied G with -1 and used evans so Loop gain marked in the RL plot below corresponds to kp = - 0.8659.

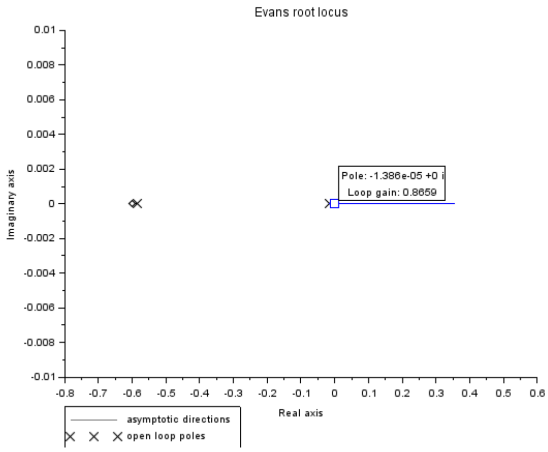

Figure 18: Root locus for kp < 0

The system is hence observed to be marginally stable for kp = - 0.867

```
s=poly(0,'s');

G = ((0.11*(s+0.6))/(6*s^2 + 3.6127*s + 0.0572));
G = syslin('c', G);
evans(G,100);

t=0:0.01:20;
kp = 85.8;
G1 = (kp*G/(1+kp*G));
H = syslin('c',G1);
```

```
H_rep = csim('step',t,H);
plot(t,H_rep);
var = gca();
var.data_bounds = [0,0; 20,1.2];
xtitle ('Unit step response for kp = 85.8',"time instants" , "unit step response" );
```

Question 5

$$G1 (s) = 25 /(s^2+ 50)(s^2 + s + 0.5)$$

$$G2 (s) = 0.5/(s^2 + s + 0.5)$$

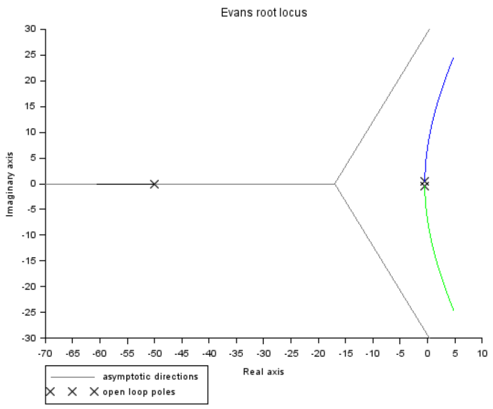

Figure 19: Root Locus for G1(s)

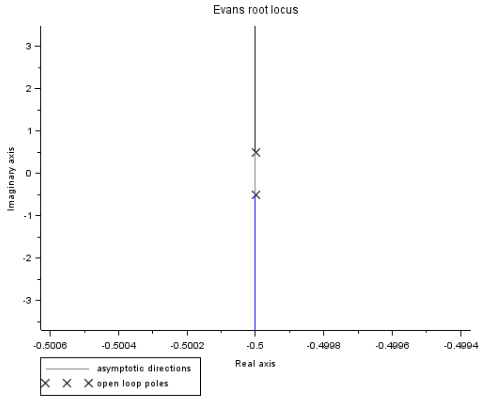

Figure 20: Root Locus for G2(s)

As kp increases, increases, the roots of the third order system move towards towards that RHP while the roots of the second order system have the same real part. Hence as kp increases, increases, the third order system experiences experiences lesser rising time and peak time but more settling settling time and percentage percentage overshoo overshoot; whereas whereas the second order system has a constant constant settling settling time and it's percentage percentage overshoot overshoot increases increases and peak time decreases decreases as kp increases. The difference amongst the step responses becomes prominent around k = 41.

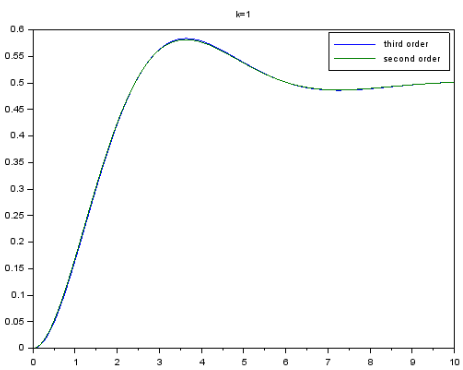

Figure 21: k = 1

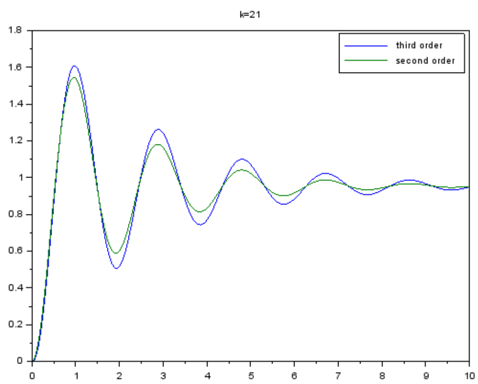

Figure 22: k = 21

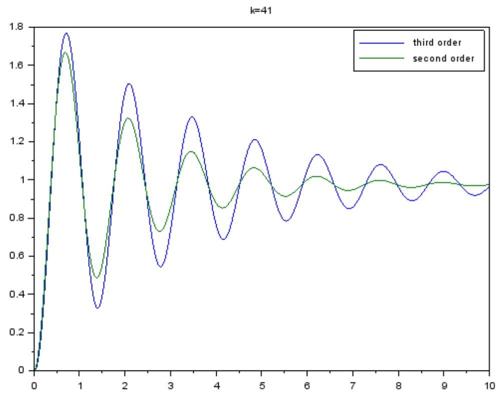

Figure 23: k = 41

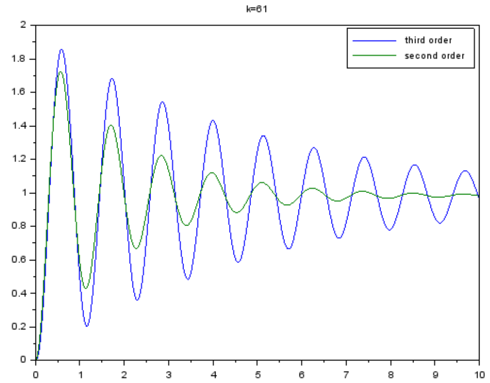

Figure 24: k = 61

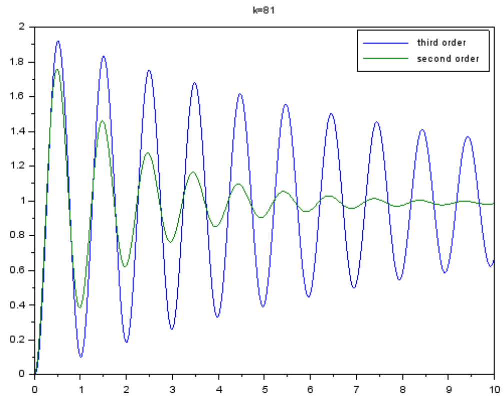

Figure 25: k = 81

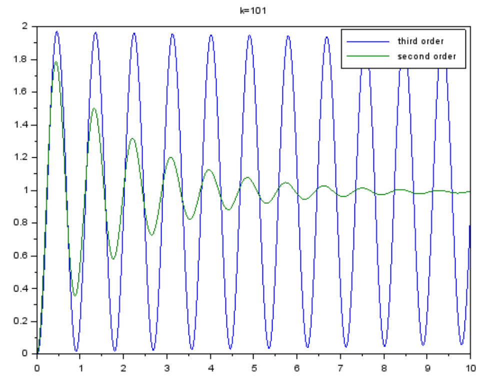

Figure 26: k = 101

Scilab code:

```
s=poly(0,'s');
G1 = (25/((s+50)*(s^2+s+0.5)));
G2 = 0.5/(s^2+s+0.5);

sysG1 = syslin('c',G1);
sysG2 = syslin('c',G2);

fig=scf();
evans(sysG1);
fig=scf();
evans(sysG2);

k = 1:20:101;
t=0:0.01:10;
for k1=k
  sysG1 = syslin('c',k1*G1/(1+k1*G1));
  sysG2 = syslin('c',k1*G2/(1+k1*G2));
  step1 = csim('step',t,sysG1);
  step2 = csim('step',t,sysG2);
  fig=scf();
  plot(t,step1,t,step2);
  title('k='+string(k1));
  h2=legend(['third order','second order'])
end
```