UNIVERSIDAD NACIONAL DE CÓRDOBA

FACULTAD DE CIENCIAS EXACTAS, FÍSICAS Y NATURALES

CÁTEDRA DE SISTEMAS DE CONTROL I

Ejercicios (11 al 25) Unidad 4

Nombre: Monja Ernesto Joaquín

DNI: 43.873.728

Año 2024

Problema 11:

Ecuación Característica 1:

```
EC1 =

    1.0000    2.9960    3.0000   10.9980

El Sistema 1 es Inestable
```

Ecuación Característica 2:

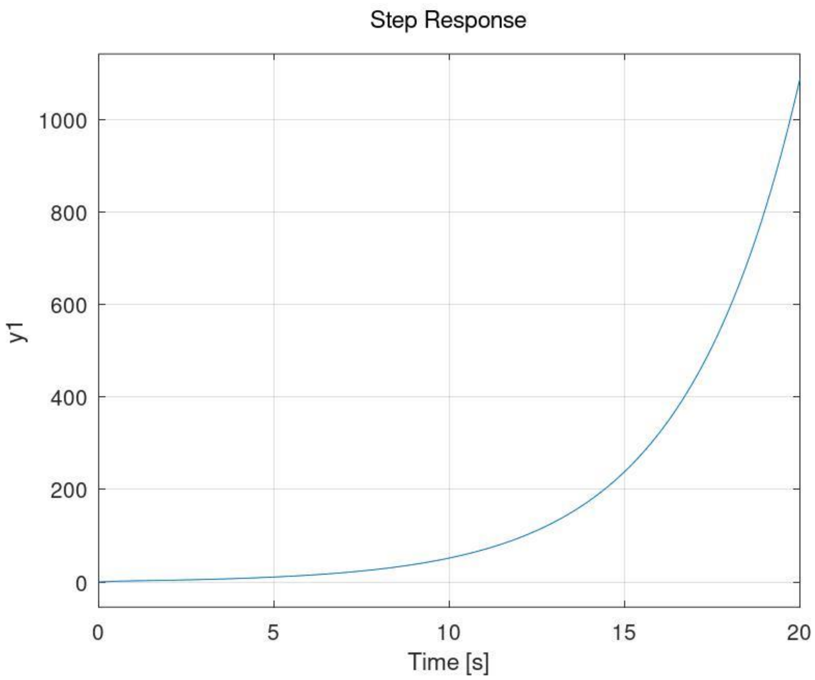
```
EC2 =

   2   2   2

El Sistema 2 es Estable
```

Código:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

% Ejercicio 11.1:
EC1 = [1 2.996 3 10.998];
PolosEC1 = roots(EC1);
if all(real(PolosEC1) < 0)
  disp("El Sistema 1 es Estable");
else
  disp("El Sistema 1 es Inestable");
end

% Ejercicio 11.2:
% 2*(s + 1)*(s + 1) = 2*(s^2 + s + s + 1) = 2*s^2 + 2*s + 2
EC2 = [2 2 2]
PolosEC2 = roots(EC2);
if all(real(PolosEC2) < 0)
  disp("El Sistema 2 es Estable");
else
  disp("El Sistema 2 es Inestable");
end

% Ejercicio 11.3:
% EC3 = 3*s^2 + (2 + k)*s + 1;
% Aplicando el criterio de Routh-Hurwitz se tiene que:
%   3      1
% (2+k)    0
%   x
% De aquí se calcula x como: x = ((2 + k)*1 - 3*0)/(2 + k) = 1
% Por lo tanto para que el sistema sea estable, según Routh-Hurwitz
% se tiene que dar que: (2 + k) > 0, esto es: k > -2
```

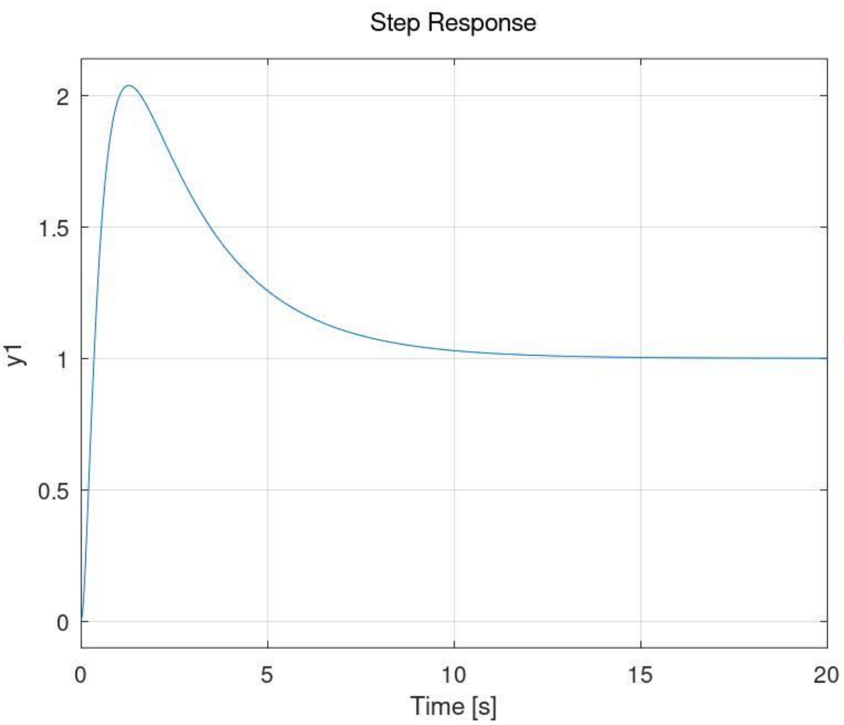
Problema 12:

G1(s):



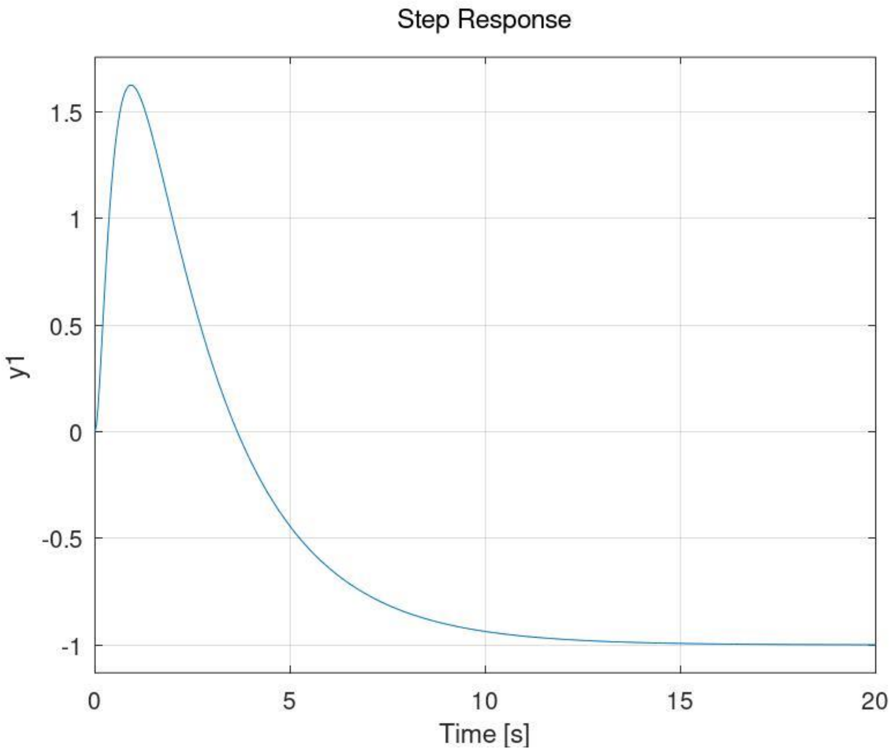
`El Sistema es Inestable`

G2(s):



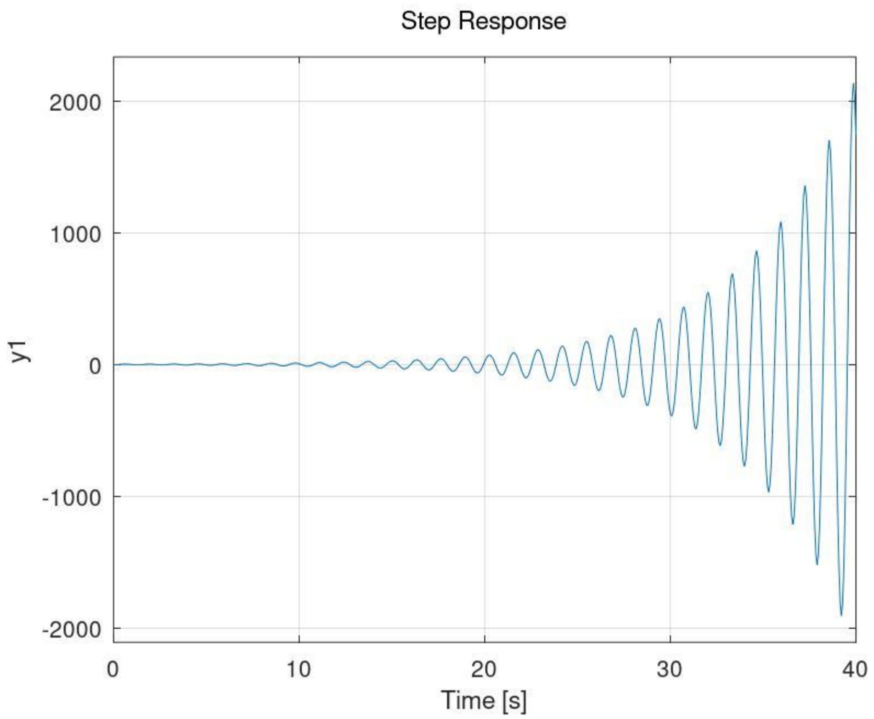
`El Sistema es Estable`

G3(s):



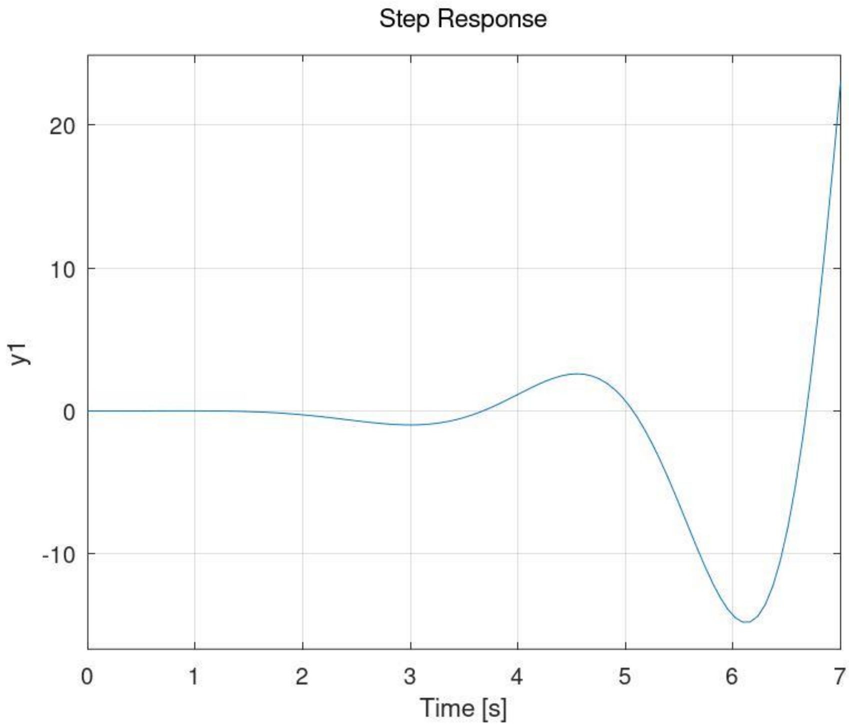
`El Sistema es Estable`

G4(s):



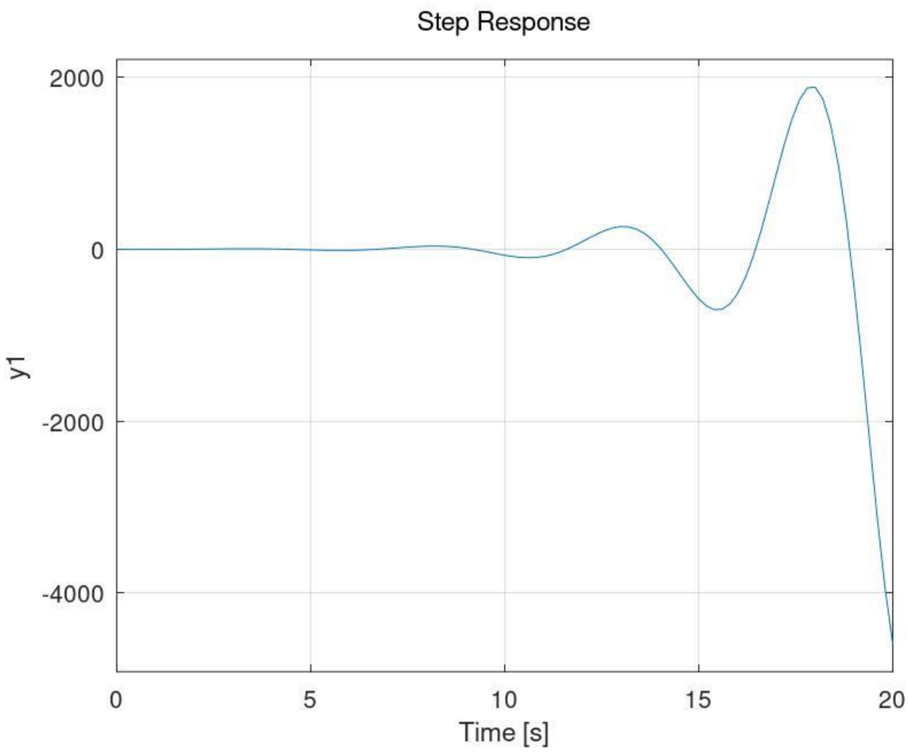
`El Sistema es Inestable`

G5(s):



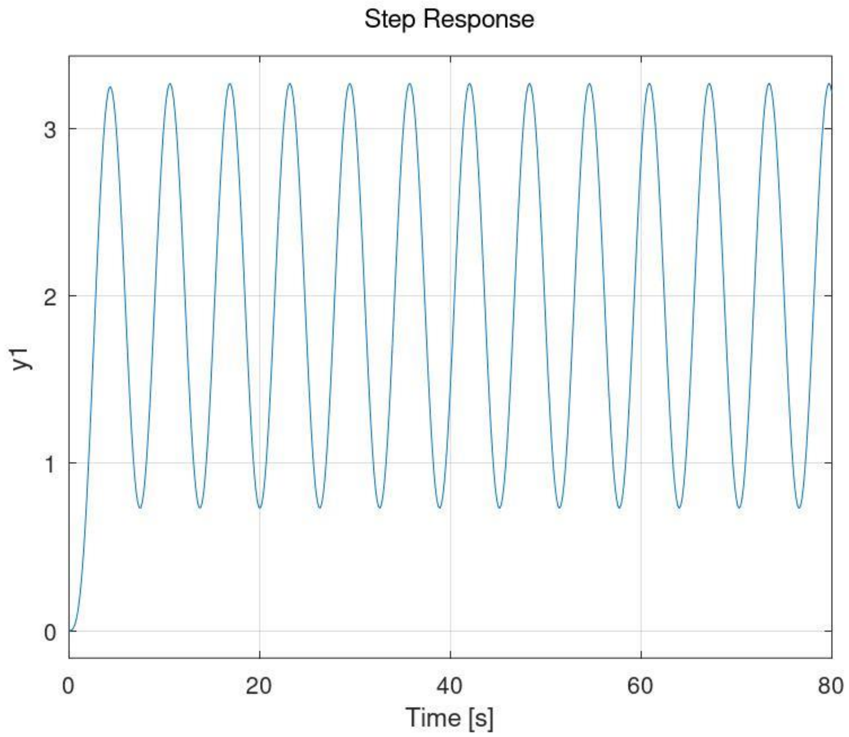
El Sistema es Inestable

G6(s):



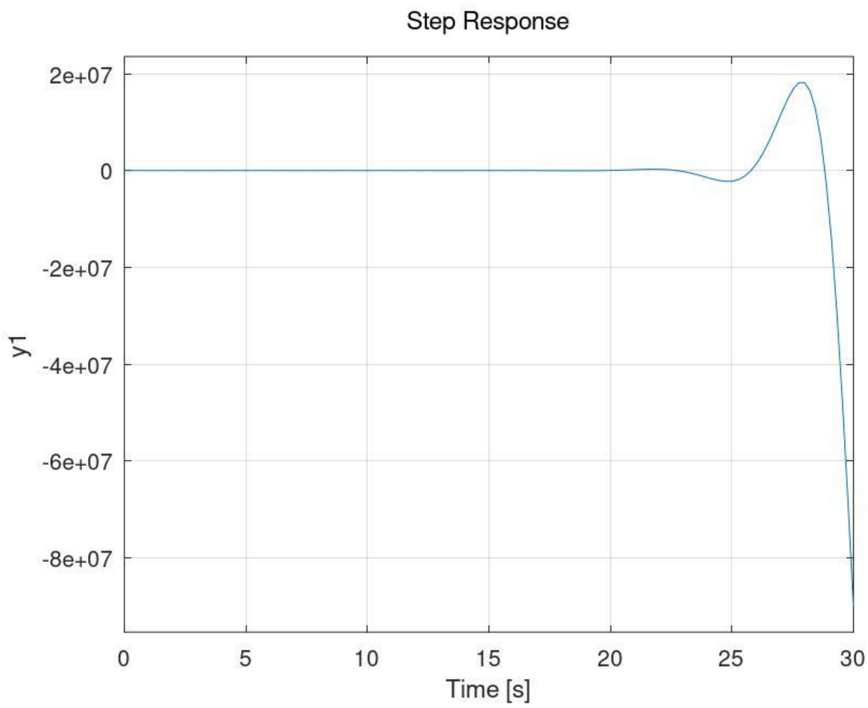
El Sistema es Inestable

G7(s):



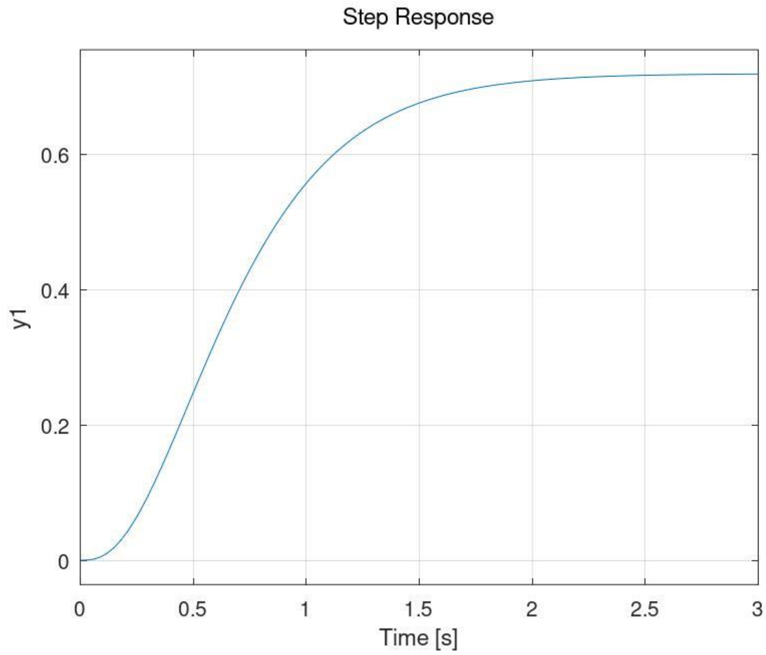

```
El Sistema es Inestable
```

G8(s):




```
El Sistema es Inestable
```

G9(s):




```
El Sistema es Estable
```

Código Genérico:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf("s");
G9 = 54/(s^3 + 13*s^2 + 55*s + 75);

% Enciso 1:
PolosG9 = pole(G9)

% Enciso 2:
pzmap(G9)

% Enciso 3:
if all(real(PolosG9)<0)
  disp("El Sistema es Estable");
else
  disp("El Sistema es Inestable");
end

% Enciso 4:
step(G9)
```

Problema 13:

Solución:

Solucion: $-(MI + mI + Ml^2 m) < 0$ $(M + m)glm > 0$ $bglm > 0$, Hay al menos 2 coeficientes que no tienen el mismo signo, eso asegura que el sistema es inestable.

Problema 14:

Sistema 1:

```
El Sistema 1 es Estable
```

```
close all; clear all; history -c; clc;
pkg load control;

s = tf("s");

% Inciso A:
% 2*(s + 1)*(s + 1) = 2*(s^2 + s + s + 1) = 2*s^2 + 2*s + 2
EC1 = [2 2 2];
PolosEC1 = roots(EC1);
if all(real(PolosEC1) < 0)
  disp("El Sistema 1 es Estable");
else
  disp("El Sistema 1 es Inestable");
end
```

Sistema 2:

```
El Sistema 2 es Estable para: 0 < k < 100
```

```
close all; clear all; history -c; clc;
pkg load simbolic;

syms s k real;

% Inciso B:
% s^3 + 11*s^2 + 10^s + k = 0
% Aplicando Routh-Hurwitz:
% 1     10
% 11    k
% x1    x2
% y1

x1 = (11*10 - 1*k)/11             % x1 = 10 - k/11
x2 = (11*0 - 1*0)/11              % x2 = 0

y1 = simplify((x1*k - 11*x2)/x1)  % y1 = k

% Se tiene que de la expresion de x1 e y1 que:
% 10 - k/11 > 0  ====>  0 < k < 100
disp("El Sistema 2 es Estable para: 0 < k < 100")
```

Sistema 3:

```
El Sistema 3 es Inestable
```

```
close all; clear all; history -c; clc;
pkg load control;

s = tf("s");

% Inciso C:
EC3 = [1 2.996 3 10.998];
PolosEC3 = roots(EC3);
if all(real(PolosEC3) < 0)
  disp("El Sistema 3 es Estable");
else
  disp("El Sistema 3 es Inestable");
end
```

Sistema 4:

```
El Sistema 4 es Inestable
```

```
close all; clear all; history -c; clc;
pkg load control;

s = tf("s");

% Inciso D:
EC4 = [2 2 3 1 3 2 1];
PolosEC4 = roots(EC4);
if all(real(PolosEC4) < 0)
  disp("El Sistema 4 es Estable");
else
  disp("El Sistema 4 es Inestable");
end
```

Sistema 5:

```
El Sistema 5 es Inestable
```

```
close all; clear all; history -c; clc;
pkg load control;

s = tf("s");

% Inciso E:
EC5 = [1 3 -2 7 12];
PolosEC5 = roots(EC5);
if all(real(PolosEC5) < 0)
  disp("El Sistema 5 es Estable");
else
  disp("El Sistema 5 es Inestable");
end
```

Sistema 6:

```
El Sistema 6 es Inestable para todo valor de k
```

```
close all; clear all; history -c; clc;
pkg load symbolic;

syms s k real;

% Inciso F:
% s^5 + s^4 + s^3 + 3*s^2 + (2 + k)*s + 1 = 0
% Aplicando Routh-Hurwitz:
%  1    1   (2 + k)
%  1    3      1
%  x1   x2
%  y1   y2
%  z1

x1 = (1*1 - 1*3)/1;                % x1 = -2
x2 = (1*(2 + k) - 1*1)/1;          % x2 = k + 1

y1 = (x1*3 - 1*x2)/x1;             % y1 = (k+7)/2
y2 = (x1*1 - 1*0)/x1;              % y2 = 1

z1 = simplify((y1*x2 - x1*y2)/y1); % z1 = ((k + 1)*(k + 7) + 4)/(k + 7)

% Se tiene que el sistema es inestable para todo valor de k, ya que x1<0
disp("El Sistema 6 es Inestable para todo valor de k")
```

Sistema 7:

```
El Sistema 7 es Estable para todo k > 0,5
```

```
close all; clear all; history -c; clc;
pkg load symbolic;

syms s k real;

% Inciso G:
% 6*s^4 + s^3 + 3*s^2 + k*s + 1 + k = 0
% Aplicando Routh-Hurwitz:
%  6    3   (1 + k)
%  1    k      0
%  x1   x2
%  y1   y2
%  z1

x1 = (1*3 - 6*k)/1;                % x1 = 3 - 6*k
x2 = (1*(1 + k) - 6*0)/1;          % x2 = 1 + k

y1 = simplify((x1*k - 1*x2)/x1);   % y1 = (6*k^2 - 2*k + 1)/(6*k - 3)
y2 = (x1*0 - 1*0)/x1;              % y2 = 0

z1 = simplify((y1*x2 - x1*y2)/y1); % z1 = 1 + k

% Si miramos el criterio de estabilidad para x1, se tiene que: 3 - 6*k > 0 =>
% k > 0,5. Luego para y1, se puede graficar esta funcion de k y se observa que
% y1 es positiva para k > - 0,5. Por ultimo se tiene que mirando a z1, k > -1
% para que z1 sea positiva, por lo tanto el sistema es estable para todo k mayor
% a 0,5 de modo que se cumplan las 3 condiciones.
disp("El Sistema 7 es Estable para todo k > 0,5")
```

Problema 15:

Rango de estabilidad:

```
El sistema es estable para todo k que cumpla que: 0 < k < 8
```

Código:

```
close all; clear all; history -c; clc;
pkg load symbolic;

syms s k real;
% Para determinar la estabilidad del sistema es necesario estudiar su ecuación
% característica, por lo tanto necesitamos la función de transferencia de lazo
% cerrado, la cual es:
G = 1/(s + 1)^3;
FdTLC = collect(simplify(k*G/(1 + k*G)), 's')
% Se obtuvo que FdTLC = k/(k + (s+1)^3) = k/(s^3 + 3*s^2 + 3*s + 1 + k)
% Por lo que se puede plantear Routh-Hurwitz con la ecuación característica
% siendo esta igual a: s^3 + 3*s^2 + 3*s + 1 + k = 0
%   1      3
%   3    (1 + k)
%   x1
%   y1

x1 = (3*3 - 1*(1 + k))/3;      % x1 = (8-k)/3
y1 = (x1*(1 + k) - 3*0)/x1;    % y1 = 1 + k

% Se verifica entonces que de la expresión de x1, que 8/3 - k/3 > 0 => k < 8 y
% al ser una ganancia, no tiene sentido que esta sea menor a 0, por lo que el
% rango de estabilidad del sistema es el siguiente:
disp("El sistema es estable para todo k que cumpla que: 0 < k < 8")
```

Problema 16:

Código:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

% Sistema 1:
G1 = 10/(s^2 + 10);
rlocusx(G1);
% Sistema Inestable para todo k

% Sistema 2:
G2 = (10*s + 20)/(s^2 + 120*s + 10);
rlocusx(G2);
% Sistema Estable para todo k > 0

% Sistema 3:
G3 = 45/(s^3 + 12*s^2 + 10*s + 45);
rlocusx(G3);
% Sistema Estable para todo k < 1.67

% Sistema 4:
G4 = ((s + 10)*(s + 20))/((s + 1)*(s + 5));
rlocusx(G4);
% Sistema Estable para todo k > 0

% Sistema 5:
G5 = ((s + 10)*(s + 20))/((s - 1)*(s + 5));
rlocusx(G5);
% Sistema Estable para todo k > 0.026

% Sistema 6:
G6 = 1/(s + 10)*s/(s^2 + s + 1);
rlocusx(G6);
% Sistema Estable para todo k > 0
```

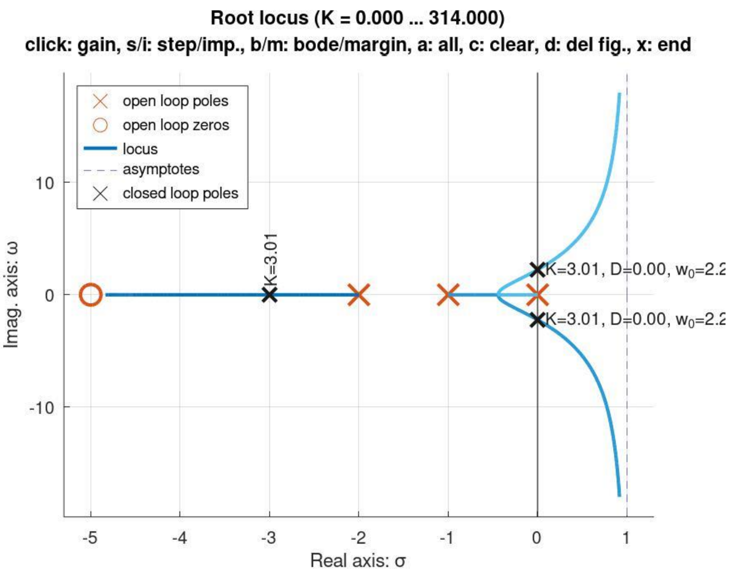
Problema 17:

Código:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

% Inciso a)
rlocus(zpk([-8], [0], 1)); sgrid
% Inciso b)
rlocus(zpk([-4], [0 -2], 1)); sgrid
% Inciso c)
rlocus(zpk([0], [-1 -1 -1], 1)); sgrid
% Inciso d)
rlocus(zpk([-4 -6], [0 -1], 1)); sgrid
% Inciso e)
rlocus(zpk([-4], [0 -2], 1)); sgrid
% Inciso f)
rlocus(zpk([-8 -10], [3 -4], 1)); sgrid
% Inciso g)
rlocus(zpk([], [1 -1 -3 -4], 1)); sgrid
% Inciso h)
rlocus(zpk([], [-2+j -2-j], 1)); sgrid
% Inciso i)
rlocus(zpk([], [-2+j -2-j -10], 1)); sgrid
% Inciso j)
rlocus(zpk([], [-2+j -2-j -1], 1)); sgrid
% Inciso k)
rlocus(zpk([-10], [-2+j -2-j], 1)); sgrid
% Inciso l)
rlocus(zpk([-1 -4], [-2+j -2-j], 1)); sgrid
% Inciso m)
rlocus(zpk([1 -4], [-2+j -2-j], 1)); sgrid
% Inciso n)
rlocus(zpk([1+3j 1-3j], [-8+j -8-j], 1)); sgrid
% Inciso o)
rlocus(zpk([1+3j 1-3j], [-8.1], 1)); sgrid
```

Problema 18:

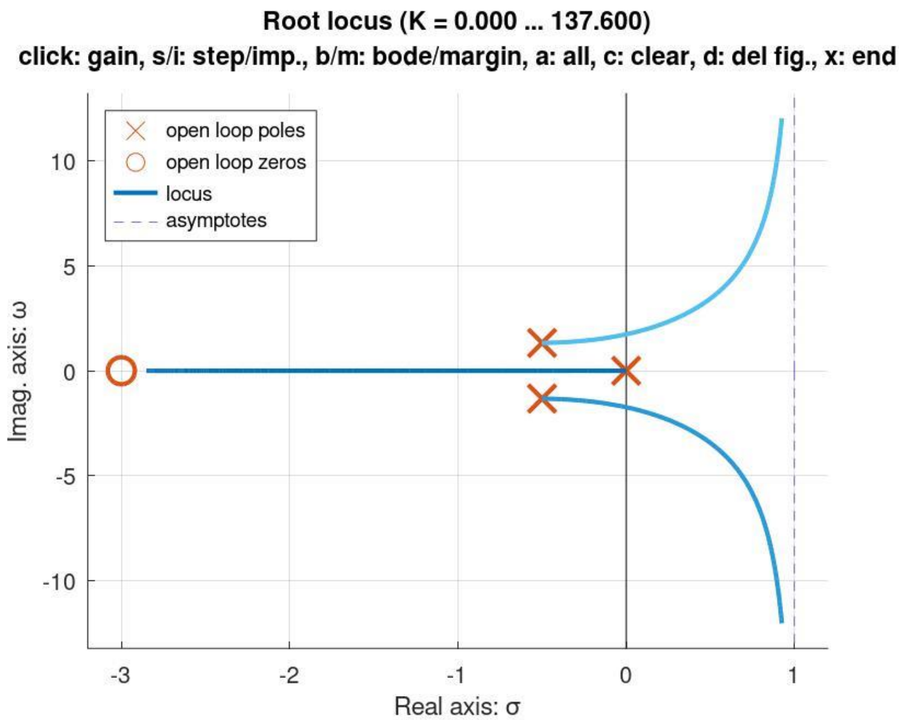
Lugar de Raíces del Sistema 1:



Código 1:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

% s^3 + 3*s^2 + (k + 2)*s + 5*k = 0
% s^3 + 3*s^2 + 2*s + k*(s + 5) = 0
% Luego, se tiene que el denominador sera el termino que no depende de K y el
% numerador sera el termino que si depende de K, por lo tanto se define a:

GH1 = (s + 5)/(s^3 + 3*s^2 + 2*s);
rlocusx(GH1)
```

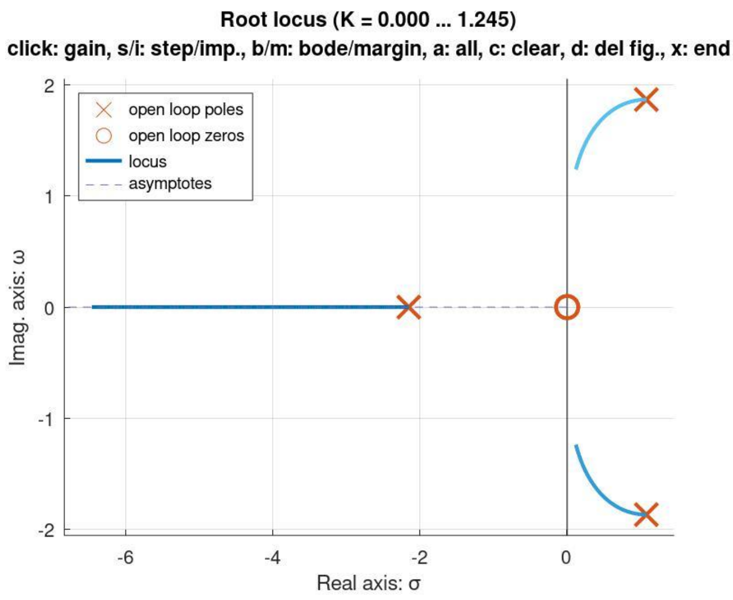
Lugar de Raíces del Sistema 2:



Código 2:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

% s^3 + s^2 + (k + 2)*s + 3*k = 0
% s^3 + s^2 + 2*s + k*(s + 3) = 0
% Luego, se tiene que el denominador sera el termino que no depende de K y el
% numerador sera el termino que si depende de K, por lo tanto se define a:

GH2 = (s + 3)/(s^3 + s^2 + 2*s);
rlocusx(GH2)
```

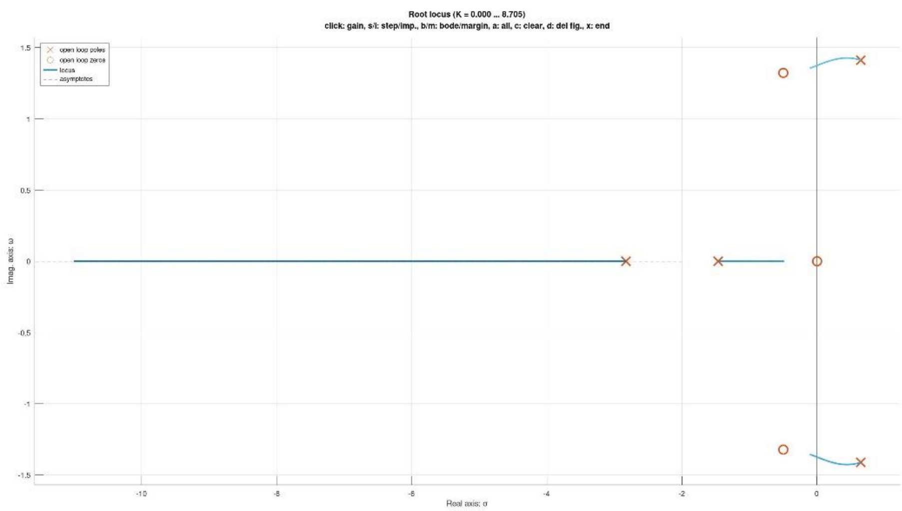
Lugar de Raíces del Sistema 3:



Código 3:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

% s^3 + 5*k*s^2 + 10 = 0
% s^3 + 10 + k*(5*s^2) = 0
% Luego, se tiene que el denominador sera el termino que no depende de K y el
% numerador sera el termino que si depende de K, por lo tanto se define a:

GH3 = (5*s^2)/(s^3 + 10);
rlocusx(GH3)
```

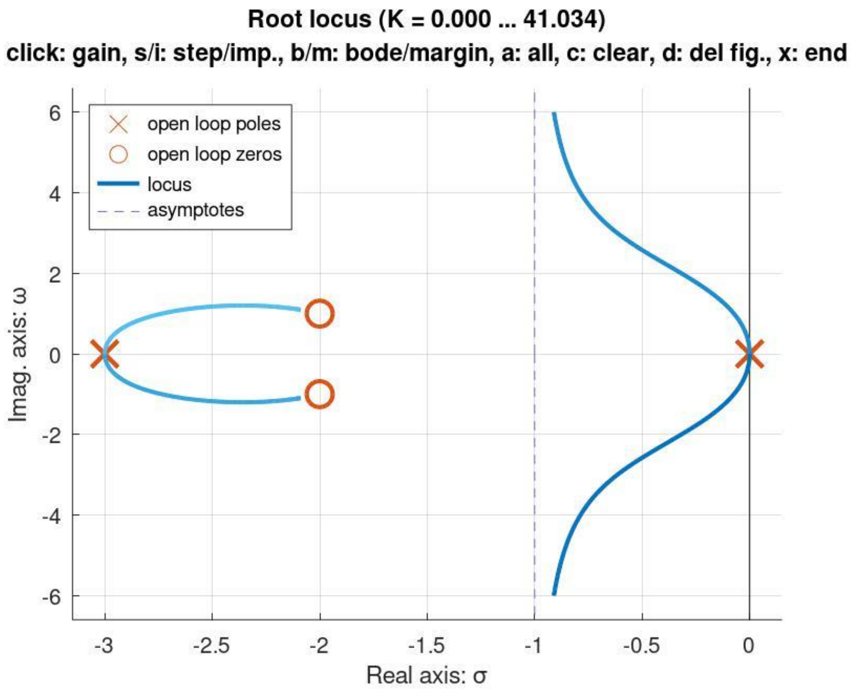
Lugar de Raíces del Sistema 4:



Código 4:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

% s^4 + (k + 3)*s^3 + (k + 1)*s^2 + (2*k + 5)*s + 10 = 0
% s^4 + k*s^3 + 3*s^3 + k*s^2 + s^2 + 2*k*s + 5*s + 10 = 0
% s^4 + 3*s^3 + s^2 + 5*s + 10 + k*(s^3 + s^2 + 2*s)= 0
% Luego, se tiene que el denominador sera el termino que no depende de K y el
% numerador sera el termino que si depende de K, por lo tanto se define a:

GH4 = (s^3 + s^2 + 2*s)/(s^4 + 3*s^3 + s^2 + 5*s + 10);
rlocusx(GH4)
```

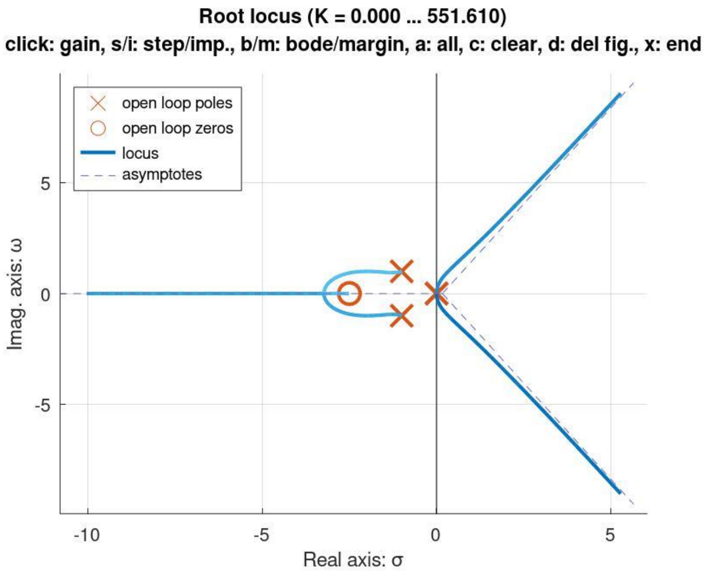
Lugar de Raíces del Sistema 5:



Código 5:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

% s^4 + 6*s^3 + 9*s^2 + k*(s^2 + 4*s + 5) = 0
% Luego, se tiene que el denominador sera el termino que no depende de K y el
% numerador sera el termino que si depende de K, por lo tanto se define a:

GH5 = (s^2 + 4*s + 5)/(s^4 + 6*s^3 + 9*s^2);
rlocusx(GH5)
```

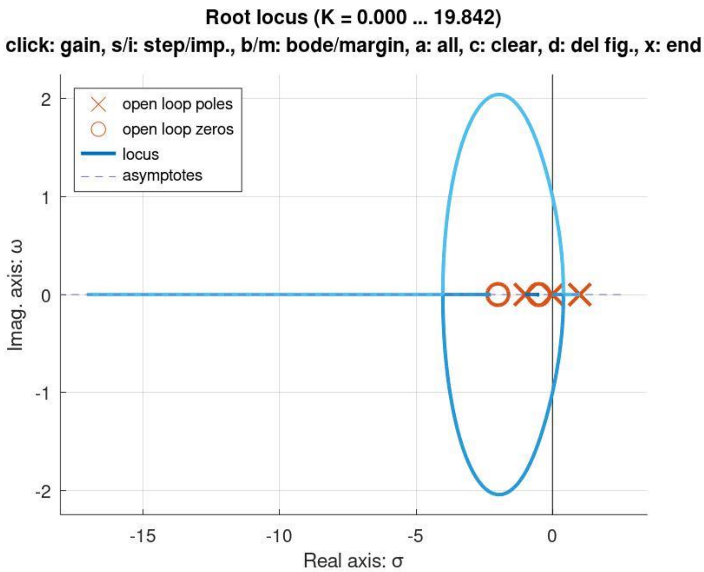
Lugar de Raíces del Sistema 6:



Código 6:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

% s^4 + 2*s^3 + 2*s^2 + 2*k*s + 5*k = 0
% s^4 + 2*s^3 + 2*s^2 + k*(2*s + 5) = 0
% Luego, se tiene que el denominador sera el termino que no depende de K y el
% numerador sera el termino que si depende de K, por lo tanto se define a:

GH6 = (2*s + 5)/(s^4 + 2*s^3 + 2*s^2);
rlocusx(GH6)
```

Lugar de Raíces del Sistema 7:

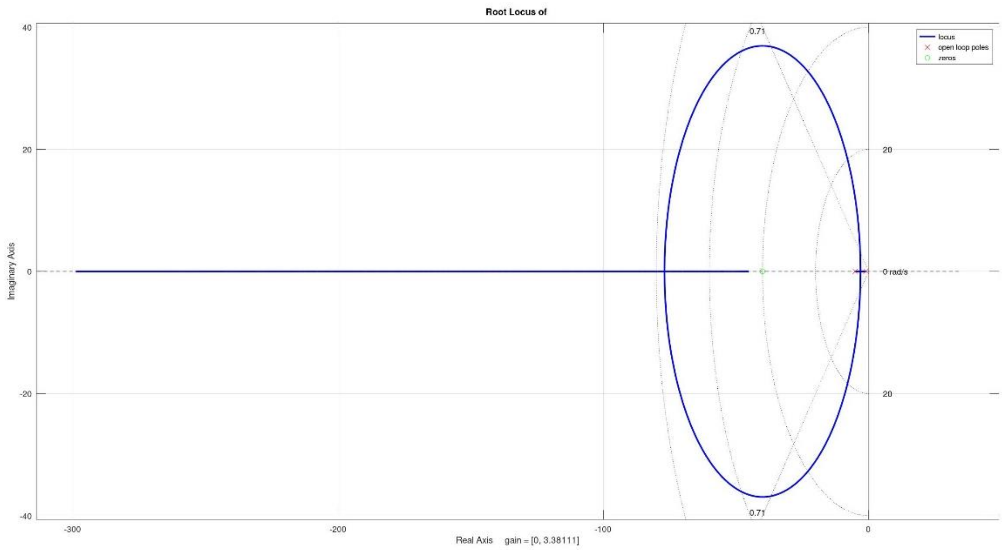


Código 7:

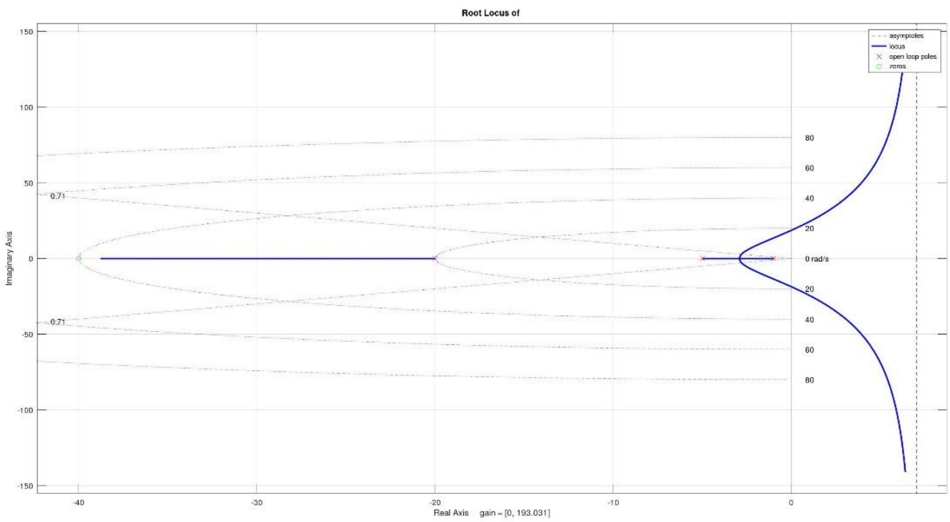
```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

% s*(s^2 - 1) + k*((s + 2)*(s + 0.5)) = 0
% Luego, se tiene que el denominador sera el termino que no depende de K y el
% numerador sera el termino que si depende de K, por lo tanto se define a:

GH7 = (s + 2)*(s + 0.5)/(s*(s^2 - 1));
rlocusx(GH7)
```

Problema 19:

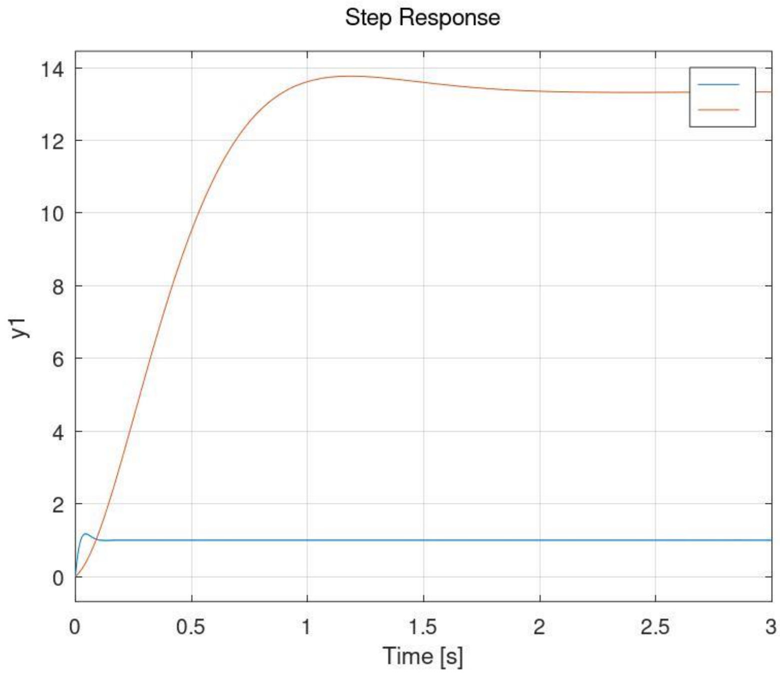
19.1) Lugar de Raíces con H1(s):



19.2) Lugar de Raíces con H2(s):



19.3) y 19.4)



Código:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

% Inciso 1:
G = zpk([-40], [-1 -5], [100]);
H1 = 1;
rlocus(G*H1); sgrid(0.707, [20 40 60 80])

% Inciso 2:
H2 = zpk([], [-20], [1]);
rlocus(G*H2); sgrid(0.707, [20 40 60 80])

% Inciso 3:
% Un sobrepasamiento maximo del 4% equivale a un cierto valor de psita el cual
% calcularemos a continuación
psita = sqrt((log(0.04)^2)/(pi^2 + log(0.04)^2))  % psita = 0.7156

K1 = 0.7;
K2 = 0.05;
step(feedback(K1*G, H1),feedback(K2*G, H2))
```

Problema 20:

Código:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

G1 = 100/(s*(s + 5));
rlocus(G1); sgrid       % El sistema oscilará si k = 0

G2 = 100*(s^2 + 40*s + 800)/((s + 80)*(s + 50));
rlocus(G2); sgrid       % El sistema no oscilará

G3 = 100/((s + 80)*(s + 50)*(s - 10));
rlocusx(G3)             % El sistema oscilará si k = 3640

G4 = 100*(s + 40)/((s + 5)*(s^2 + 20*s + 1700))
rlocusx(G4)             % El sistema oscilará si k = 24.35

G5 = 100*(s + 40)/((s - 5)*(s^2 + 20*s + 1700))
rlocusx(G5)             % El sistema oscilará si k = 13

G6 = 100*(s - 40)/((s + 25)*(s^2 + 20*s + 1700))
rlocusx(G6)             % El sistema oscilará si k = 10.64
```

Problema 21:

Código:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

% Encontraremos primero el cruce entre el LdR y el psita de 0.707 para luego
% utilizar rlocusx() y localizar el valor de K en el punto donde el LdR y el
% psita coincidan. Nótese que estos valores son aproximados

G1 = 100/(s*(s + 5));
rlocus(G1); sgrid(0.707, [20 40 60 80]);          % s = -2.5 +- j2.5
rlocusx(G1);                                       % k = 0.12

G2 = 100*(s^2 + 40*s + 800)/((s + 80)*(s + 50));
rlocus(G2); sgrid(0.707, [20 40 60 80]);          % s = -20 +- j20
rlocusx(G2);                                       % k = 10.71

G3 = 100/((s + 80)*(s + 50)*(s - 10));
rlocus(G3); sgrid(0.707, [20 40 60 80]);          % s = -12.5 +- j12.5
rlocusx(G3);                                       % k = 700

G4 = 100*(s + 40)/((s + 5)*(s^2 + 20*s + 1700));
rlocus(G4); sgrid(0.707, [20 40 60 80]);          % No hay interseccion
rlocusx(G4);

G5 = 100*(s + 40)/((s - 5)*(s^2 + 20*s + 1700));
rlocus(G5); sgrid(0.707, [20 40 60 80]);          % s = 0
rlocusx(G5);                                       % k = 2.13

G6 = 100*(s - 40)/((s + 25)*(s^2 + 20*s + 1700));
rlocus(G6); sgrid(0.707, [20 40 60 80]);          % s = 0
rlocusx(G6);                                       % k = 10.64
```

Problema 22:

Código:

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');

% Encontraremos primero el cruce entre el LdR y el psita de 0.66 para luego
% utilizar rlocusx() y localizar el valor de K en el punto donde el LdR y el
% psita coincidan. Nótese que estos valores son aproximados

G = 1/(s^3 + 4*s^2 + 5*s);
rlocus(G); sgrid(0.66, [20 40 60 80]);          % s = -0.74 +- j0.845

% Método Analítico:
s1 = -0.74 + j*0.845;
K = 1/abs(1/(s1^3 + 4*s1^2 + 5*s1))             % k = 3.1858

% Método Gráfico:
rlocusx(G);                                     % k = 3.17
```

Problema 23:

23.1)

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');
G1 = 20/((s + 10)*(s + 100));

% Inciso 1:
% Si queremos una respuesta criticamente amortiguada, se tendran dos raices
% reales negativas e iguales, por lo que buscaremos que el LdR coincida con
% el eje x, osea que graficaremos el LdR y veremos este punto:
rlocus(G1)                                      % s1 = -55

% Se verifica que podemos obtener el valor, tanto analíticamente mediante su
% formula, como graficamente mediante el uso del rlocusx()
s11 = -55;
K = 1/abs(20/((s11 + 10)*(s1 + 100)))           % k = 101.25

rlocusx(G1)                                     % k = 101.25

% Inciso 2:
% Un sobrepasamiento máximo del 4% corresponde a un cierto valor de psita el
% cual se puede calcular a continuación:
psita = sqrt((log(0.04)^2)/(pi^2 + log(0.04)^2))  % psita = 0.7156
% Se verifica donde coinciden los valores de psita con el LdR mediante el rlocus
rlocus(G1); sgrid(psita, [20 40 60 80])

s12 = -55 + j*53.68;
K = 1/abs(20/((s12 + 10)*(s12 + 100)))          % k = 245.33
% Verificamos este resultado mediante el rlocusx():
rlocusx(G1)                                     % k = 252 aproximadamente

% Inciso 3:
FdTLC1 = feedback(G1*245.33,1);
step(FdTLC1)
```

23.2)

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');
G2 = 5/(s + 10);

% Inciso 1:
% Si queremos una respuesta criticamente amortiguada, se tendran dos raices
% reales negativas e iguales, por lo que buscaremos que el LdR coincida con
% el eje x, osea que graficaremos el LdR y veremos este punto:
rlocus(G2)                                      % s1 = -20

% Se verifica que podemos obtener el valor, tanto analíticamente mediante su
% formula, como graficamente mediante el uso del rlocusx()
s21 = -20;
K = 1/abs(5/(s21 + 10))                         % k = 2

rlocusx(G2)                                     % k = 2

% Inciso 2:
% Un sobrepasamiento máximo del 4% corresponde a un cierto valor de psita el
% cual se puede calcular a continuación:
psita = sqrt((log(0.04)^2)/(pi^2 + log(0.04)^2))     % psita = 0.7156
% Se verifica donde coinciden los valores de psita con el LdR mediante el rlocus
rlocus(G2); sgrid(psita, [20 40 60 80])
% No hay interseccion entre psita y el LdR

% Inciso 3: No es posible
```

23.3)

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');
G3 = 1200/(s^2 + 70*s + 1000);

% Inciso 1:
% Si queremos una respuesta criticamente amortiguada, se tendran dos raices
% reales negativas e iguales, por lo que buscaremos que el LdR coincida con
% el eje x, osea que graficaremos el LdR y veremos este punto:
rlocus(G3)                                     % s31 = -35

% Se verifica que podemos obtener el valor, tanto analíticamente mediante su
% formula, como graficamente mediante el uso del rlocusx()
s31 = -35;
K = 1/abs(1200/(s31^2 + 70*s31 + 1000))        % k = 0.1875

rlocusx(G3)                                    % k = 0.19 aproximadamente

% Inciso 2:
% Un sobrepasamiento máximo del 4% corresponde a un cierto valor de psita el
% cual se puede calcular a continuación:
psita = sqrt((log(0.04)^2)/(pi^2 + log(0.04)^2))     % psita = 0.7156
% Se verifica donde coinciden los valores de psita con el LdR mediante el rlocus
rlocus(G3); sgrid(psita, [20 40 60 80])

s32 = -35 + j*35.15;
K = 1/abs(1200/(s32^2 + 70*s32 + 1000))         % k = 1.2171
% Verificamos este resultado mediante el rlocusx():
rlocusx(G3)                                     % k = 1.21 aproximadamente

% Inciso 3:
FdTLC3 = feedback(G3*1.2171, 1);
step(FdTLC3)
```

23.4)

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');
G4 = 12*(s + 10)/(s*(s + 30));

% Inciso 1:
% Si queremos una respuesta criticamente amortiguada, se tendran dos raices
% reales negativas e iguales, por lo que buscaremos que el LdR coincida con
% el eje x, osea que graficaremos el LdR y veremos este punto:
rlocus(G4)                                        % s41 = -80

% Se verifica que podemos obtener el valor, tanto analíticamente mediante su
% formula, como graficamente mediante el uso del rlocusx()
s41 = -80;
K = 1/abs(12*(s41 + 10)/(s41*(s41 + 30)))         % k = 4.7619

rlocusx(G4)                                       % k = 4.76 aproximadamente

% Inciso 2:
% Un sobrepasamiento máximo del 4% corresponde a un cierto valor de psita el
% cual se puede calcular a continuación:
psita = sqrt((log(0.04)^2)/(pi^2 + log(0.04)^2))     % psita = 0.7156
% Se verifica donde coinciden los valores de psita con el LdR mediante el rlocus
rlocus(G4); sgrid(psita, [20 40 60 80])

s42 = 0;
K = 1/abs(12*(s42 + 10)/(s42*(s42 + 30)))          % k = 0
% Verificamos este resultado mediante el rlocusx():
rlocusx(G4)                                        % k = 0

% Inciso 3:
FdTLC4 = feedback(G4*4.7619, 1);
step(FdTLC4)
```

23.5)

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');
G5 = 0.75/((s + 1)^3);

% Inciso 1:
% Si queremos una respuesta criticamente amortiguada, se tendran dos raices
% reales negativas e iguales, por lo que buscaremos que el LdR coincida con
% el eje x, osea que graficaremos el LdR y veremos este punto:
rlocusx(G5)

% Se observa que dos de las tres ramas no se encuentran sobre el eje x para
% ningun K, por lo tanto no hay criterio para poder elegir un punto de trabajo
% que satisfaga la condicion de respuesta criticamente amortiguada.

% Inciso 2:
% Un sobrepasamiento máximo del 4% corresponde a un cierto valor de psita el
% cual se puede calcular a continuación:
psita = sqrt((log(0.04)^2)/(pi^2 + log(0.04)^2))     % psita = 0.7156
% Se verifica donde coinciden los valores de psita con el LdR mediante el rlocus
rlocus(G5); sgrid(psita, [20 40 60 80])

s52 = 0.64 + j*0.64;
k = 1/abs(12*(s52 + 10)/(s52*(s52 + 30)))           % k = 0.3741

% Inciso 3:
FdTLC5 = feedback(G5*0.3741, 1);
step(FdTLC5)
```

23.6)

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');
G6 = 150*(s + 10)*(s + 20)/(s*(s + 5));

% Inciso 1:
% Si queremos una respuesta criticamente amortiguada, se tendran dos raices
% reales negativas e iguales, por lo que buscaremos que el LdR coincida con
% el eje x, osea que graficaremos el LdR y veremos este punto:
rlocus(G6)                                              % s61 = -12.9

% Se verifica que podemos obtener el valor, tanto analíticamente mediante su
% formula, como graficamente mediante el uso del rlocusx()
s61 = -12.9;
K = 1/abs(150*(s61 + 10)*(s61 + 20)/(s61*(s61 + 5))) % k = 0.032997

rlocusx(G6)                                            % k = 0.03 aproximadamente

% Inciso 2:
% Un sobrepasamiento máximo del 4% corresponde a un cierto valor de psita el
% cual se puede calcular a continuación:
psita = sqrt((log(0.04)^2)/(pi^2 + log(0.04)^2))      % psita = 0.7156
% Se verifica donde coinciden los valores de psita con el LdR mediante el rlocus
rlocus(G6); sgrid(psita, [20 40 60 80])
% No hay interseccion entre psita y el LdR

% Inciso 3: No es posible
```

23.7)

```
close all; clear all; history -c; clc;
pkg load control;

s = tf('s');
G7 = 25/((s - 1)*(s + 20));

% Inciso 1:
% Si queremos una respuesta criticamente amortiguada, se tendran dos raices
% reales negativas e iguales, por lo que buscaremos que el LdR coincida con
% el eje x, osea que graficaremos el LdR y veremos este punto:
rlocus(G7)                                             % s71 = -9.5

% Se verifica que podemos obtener el valor, tanto analíticamente mediante su
% formula, como graficamente mediante el uso del rlocusx()
s71 = -9.5;
K = 1/abs(25/((s71 - 1)*(s71 + 20)))                  % k = 4.41

rlocusx(G7)                                            % k = 4.41 aproximadamente

% Inciso 2:
% Un sobrepasamiento máximo del 4% corresponde a un cierto valor de psita el
% cual se puede calcular a continuación:
psita = sqrt((log(0.04)^2)/(pi^2 + log(0.04)^2))      % psita = 0.7156
% Se verifica donde coinciden los valores de psita con el LdR mediante el rlocus
rlocus(G7); sgrid(psita, [20 40 60 80])

s72 = -9.5 + j*9.27;
K = 1/abs(25/((s72 - 1)*(s72 + 20)))                  % k = 7.8473
% Verificamos este resultado mediante el rlocusx():
rlocusx(G7)                                            % k = 7.85 aproximadamente

% Inciso 3:
FdTLC7 = feedback(G7*7.8473, 1);
step(FdTLC7)
```

23.8)

-

23.9)

-

23.10)

-

23.11)

-

23.12)

-

Problema 24:

Código: Se trata del mismo ejercicio que el 4.19 por lo que se omite su resolución ya que ya está resuelto en este mismo documento.

Problema 25:

Código: Se trata del mismo ejercicio que el 4.21, por lo que se omite su resolución ya que ya está resuelto en este mismo documento.