EJERCICIOS ENTREGABLES PRÁCTICA 3

Polos y ceros de una función de transferencia, respuesta transitoria de un sistema de control con MATLAB y LTIViewer, precisión y error en estado estacionario

<u>Ejercicio 1 (2,75 puntos).</u> Polos y ceros de una función de transferencia con MATLAB.

Se tiene la planta de un sistema con función de transferencia $G(s) = \dfrac{1}{s^2 + 3s - 4}$, controlador en serie, $C(s) = K$, y una realimentación negativa, cuyo bloque principal es $H(s) = \dfrac{1}{s+8}$. Se pide analizar, utilizando comandos de control de flujo (for, while, etc.) el "movimiento" de los polos y las implicaciones en la estabilidad del sistema en lazo cerrado, variando el parámetro K en el intervalo [0,∞).

Se declaran las funciones de transferencia individuales y la general, en lazo cerrado, calculando los respectivos polos considerando el sistema sin controlador.

```
G=tf(1,[1 3 -4]);
H=tf(1,[1 8]);
FTlc=feedback(G,H); polos=pole(FTlc);
```

Se estudia y clasifica la estabilidad del sistema original y se muestra por pantalla:

```
for j=1:1:length(polos)
    if real(polos(j))<0
        state="ESTABLE";
    elseif real(polos(j))==0
        state="CRÍTICAMENTE ESTABLE";
    else
        state="INESTABLE";
    end
end
i=1; disp("0<K<K"+i+": "+state);
```

```
0<K<K1: INESTABLE
```

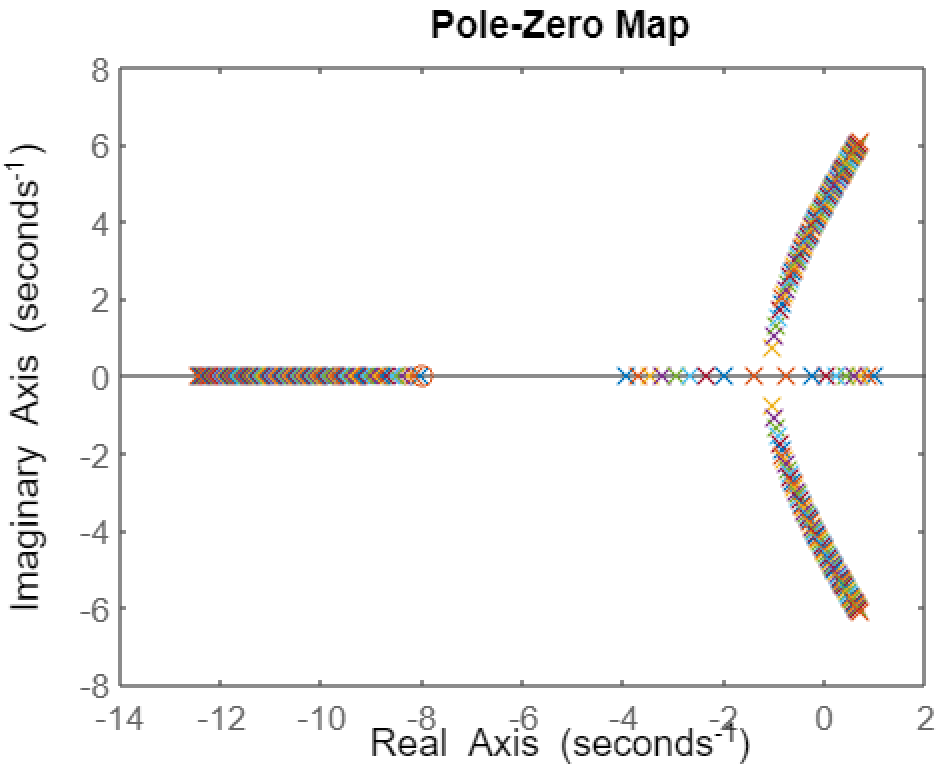
Se genera el vector de ganancias del controlador proporcional, de 1 a 500. Téngase en cuenta que no se controla el número de puntos para precisar los cambios de estabilidad en el lazo de control.

A continuación, se realiza el bucle "corazón" del problema, explicado en el código.

```
K=linspace(1,500);
figure; hold on % se "llama" a una figura para pintar todos los
% mapas de polos y ceros variables en función de K
for t=1:1:length(K)
    % Función de transferencia en lazo cerrado y cálculo de polos
    FTlc=feedback(K(t)*G,H);
```

```
    polos=pole(FTlc);
    for j=1:1:length(polos)
        % Si el sistema está clasificado como ESTABLE pero se encuentra
        % algún polo situado en el semiplano real positivo, el sistema pasa
        % a ser inmediatamente CRÍTICAMENTE ESTABLE
        if real(polos(j))>0 && state=="ESTABLE"
            state="CRÍTICAMENTE ESTABLE";
            % Se muestra por pantalla el nuevo estado
            disp("K"+i+"="+K(t)+": "+state); i=i+1;
            % A continuación, el sistema pasa a ser INESTABLE
            state="INESTABLE"; disp("K"+(i-1)+"<K<K"+i+": "+state);
            break % salida del bucle para parar de escanear los polos
        end
        % Si el sistema está clasificado como INESTABLE pero se encuentran
        % todos los polos situados en el semiplano real negativo, el
        % sistema pasa a ser inmediatamente CRÍTICAMENTE ESTABLE
        if real(polos(1))<0 && real(polos(2))<0 && real(polos(3))<0 ...
                && state=="INESTABLE"
            state="CRÍTICAMENTE ESTABLE";
            % Se muestra por pantalla el nuevo estado
            disp("K"+i+"="+ K(t)+": "+state); i=i+1;
            % A continuación, el sistema pasa a ser ESTABLE
            state="ESTABLE"; disp("K"+(i-1)+"<K<K"+i+": "+state);
            break % salida del bucle para parar de escanear los polos
        end
    end
    pzmap(FTlc); % mapeo de los polos y ceros del sistema
end
```

```
K1=36.2828: CRÍTICAMENTE ESTABLE
K1<K<K2: ESTABLE
K2=253.0202: CRÍTICAMENTE ESTABLE
K2<K<K3: INESTABLE
```



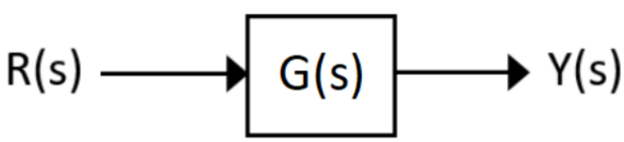

```
disp("K"+i+"=Infinito"); % llamada al último rango: K final es infinito
```

```
K3: Infinito
```

Los rangos donde se analiza la estabilidad del sistema se muestran por pantalla, junto con el mapa de polos y ceros que muestra el movimiento de los mismos.

Ejercicio 2 (3 puntos). Estudio de la respuesta transitoria de sistemas de control.

Construye un sistemas de control cuya función de transferencia G(s) no tengan ceros, tenga un par de polos complejos conjugados en s=-5±5j, y además tenga un polo variable, de -0,5 a -50.



Realice una gráfica de respuestas impulsionales, a escalón de fuerza 2 y rampa de pendiente 3 para diferentes valores del polo simple. Justifique el tipo de sistema y especifique como varían los parámetros característicos, extraídos a partir del comando *stepinfo*.

Se declara el vector que define el polo variable, donde se asumen 10 componentes dados entre los valores límite, -0.5 y -50.

```
p=linspace(-0.5,-50,10);
```

A continuación, se define el vector tiempo:

```
time=0:0.1:30;
```

Variable s perteneciente a la variable de Laplace para interactuar con el comando stepinfo:

```
s=tf('s');
```

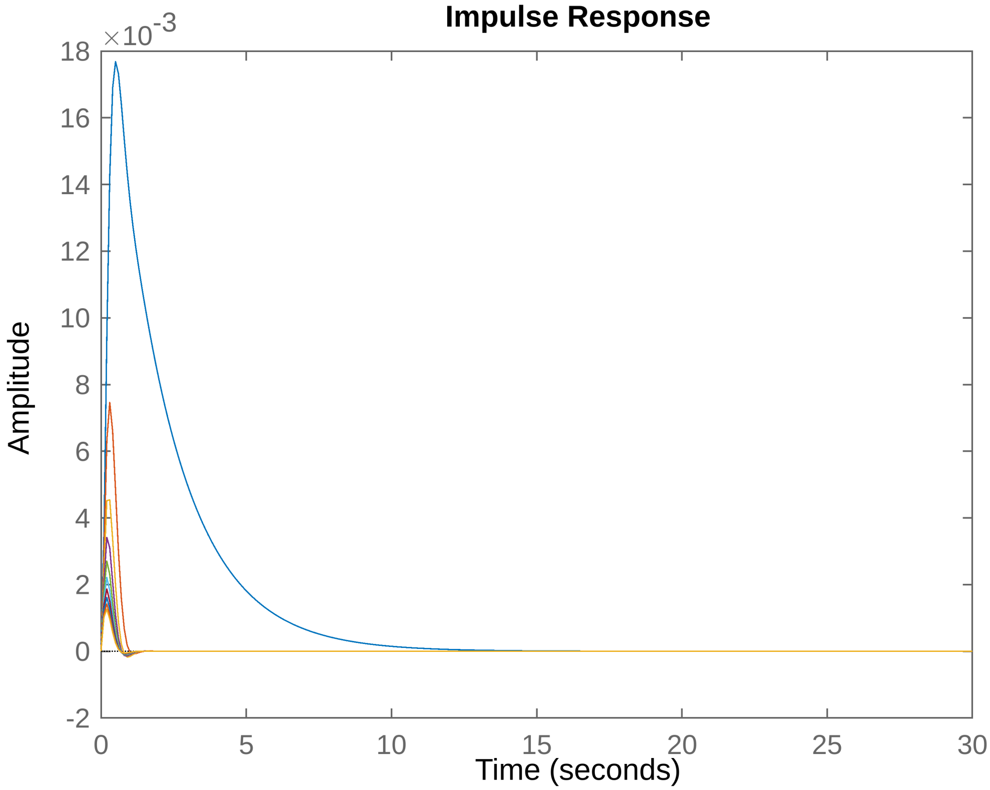
Gráficas y parámetros característicos variables dependientes de la situación del polo simple ante una entrada impulsional unitaria:

```
figure; hold on % figura donde se grafican las respuestas impulsionales
for j=1:1:length(p)
    % Función de transferencia variable
    G(j)=zpk([],[-5+5i -5-5i p(j)],1);
    disp("Polo: "+p(j)) % se muestra por pantalla el polo bajo estudio
    % Extracción de los parámetros característicos
    A=stepinfo(G(j)*s) % ya que stepinfo extrae los valores de
    % la respuesta ante escalón, es necesario operar con la función de
    % transferencia. Al ser un sistema lineal, se multiplica por
    % la variable s para "pasar" de respuesta ante escalón a impulsional
    % Gráfica de la respuesta ante impulsos de fuerza 2
    impulse(G(j),time)
end
```

```
Polo: -0.5
A = struct with fields:
         RiseTime: NaN
    TransientTime: 8.2693
     SettlingTime: NaN
      SettlingMin: NaN
      SettlingMax: NaN
        Overshoot: 0
       Undershoot: 2.5492e+17
             Peak: 0.0177
         PeakTime: 0.5158
Polo: -6
A = struct with fields:
         RiseTime: 6.3317e-16
    TransientTime: 0.9100
     SettlingTime: NaN
      SettlingMin: -8.1912e-05
      SettlingMax: 0.0075
        Overshoot: 2.1505e+17
       Undershoot: 2.3609e+15
             Peak: 0.0075
         PeakTime: 0.2947
Polo: -11.5
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.1067
     SettlingTime: NaN
      SettlingMin: -1.7591e-04
      SettlingMax: 0.0048
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0048
         PeakTime: 0.2483
Polo: -17
A = struct with fields:
```

```
         RiseTime: 0
    TransientTime: 1.0738
     SettlingTime: NaN
      SettlingMin: -1.4440e-04
      SettlingMax: 0.0035
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0035
         PeakTime: 0.2221
Polo: -22.5
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.0545
     SettlingTime: NaN
      SettlingMin: -1.1589e-04
      SettlingMax: 0.0027
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0027
         PeakTime: 0.2088
Polo: -28
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.0427
     SettlingTime: NaN
      SettlingMin: -9.5569e-05
      SettlingMax: 0.0022
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0022
         PeakTime: 0.1974
Polo: -33.5
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.0349
     SettlingTime: NaN
      SettlingMin: -8.0944e-05
      SettlingMax: 0.0019
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0019
         PeakTime: 0.1925
Polo: -39
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.0295
     SettlingTime: NaN
      SettlingMin: -7.0066e-05
      SettlingMax: 0.0016
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0016
         PeakTime: 0.1866
Polo: -44.5
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.0255
     SettlingTime: NaN
      SettlingMin: -6.1703e-05
      SettlingMax: 0.0014
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0014
         PeakTime: 0.1821
```

```
Polo: -50
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.0224
     SettlingTime: NaN
      SettlingMin: -5.5094e-05
      SettlingMax: 0.0013
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0013
         PeakTime: 0.1787
```



De los resultados, se extrae principalmente que, a medida que el polo se aleja del origen, el tiempo de establecimiento decrece. Lógicamente, el sistema se vuelve más rápido. Se obtiene la misma dinámica de variación para todos los tiempos. Sin embargo, van apareciendo oscilaciones más prominentes (valores de *overshoot* y *undershoot*) ya que, cada vez, los polos con parte real y compleja conjugada van siendo más dominantes. Esto va en consonancia con los valores de pico.

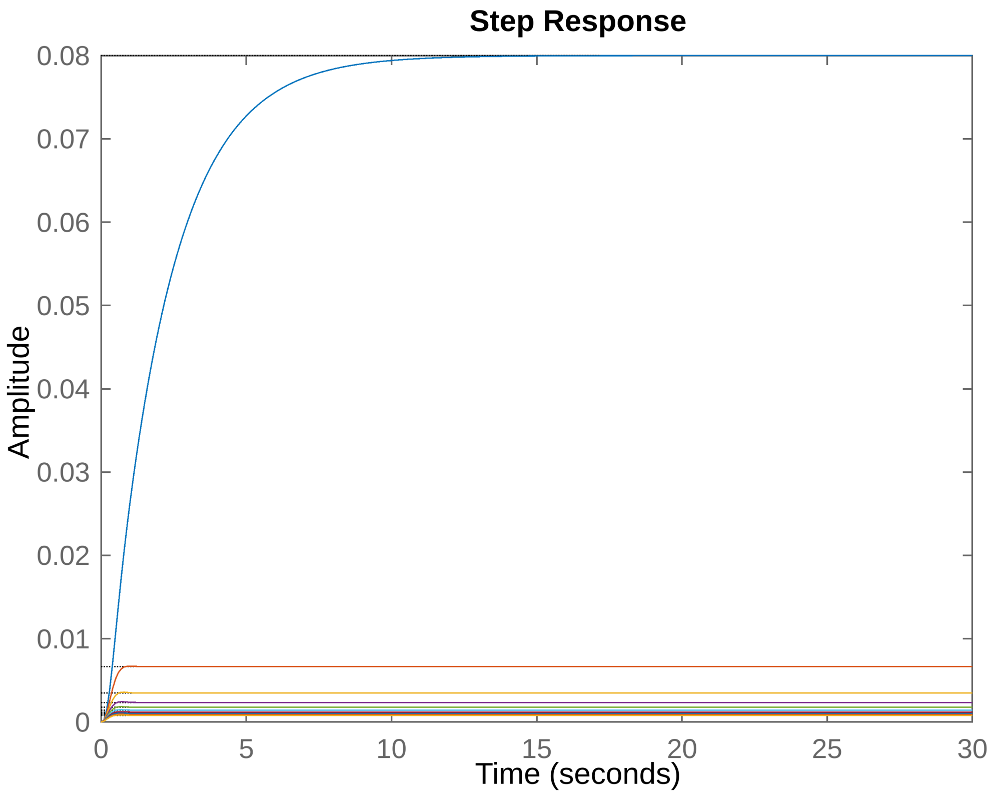
Gráficas y parámetros característicos variables dependientes de la situación del polo simple ante una entrada escalón de fuerza 2:

```
figure; hold on % figura donde se grafican las respuestas impulsionales
for j=1:1:length(p)
    % Función de transferencia variable
    G(j)=zpk([],[-5+5i -5-5i p(j)],1);
    disp("Polo: "+p(j)) % se muestra por pantalla el polo bajo estudio
    % Extracción de los parámetros característicos
    A=stepinfo(2*G(j)*s) % ya que stepinfo extrae los valores de
```

```
    % la respuesta ante escalón, es necesario operar con la función de
    % transferencia. Al ser un sistema lineal, se multiplica por
    % la variable s para "pasar" de respuesta ante escalón a impulsional
    % Gráfica de la respuesta ante impulsos de fuerza 2
    step(2*G(j),time)
end
```

```
Polo: -0.5
A = struct with fields:
         RiseTime: NaN
    TransientTime: 8.2693
     SettlingTime: NaN
      SettlingMin: NaN
      SettlingMax: NaN
        Overshoot: 0
       Undershoot: 2.5492e+17
             Peak: 0.0354
         PeakTime: 0.5158
Polo: -6
A = struct with fields:
         RiseTime: 6.3317e-16
    TransientTime: 0.9100
     SettlingTime: NaN
      SettlingMin: -1.6382e-04
      SettlingMax: 0.0149
        Overshoot: 2.1505e+17
       Undershoot: 2.3609e+15
             Peak: 0.0149
         PeakTime: 0.2947
Polo: -11.5
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.1067
     SettlingTime: NaN
      SettlingMin: -3.5182e-04
      SettlingMax: 0.0095
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0095
         PeakTime: 0.2483
Polo: -17
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.0738
     SettlingTime: NaN
      SettlingMin: -2.8880e-04
      SettlingMax: 0.0069
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0069
         PeakTime: 0.2221
Polo: -22.5
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.0545
     SettlingTime: NaN
      SettlingMin: -2.3179e-04
      SettlingMax: 0.0054
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0054
         PeakTime: 0.2088
Polo: -28
```

```
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.0427
     SettlingTime: NaN
      SettlingMin: -1.9114e-04
      SettlingMax: 0.0044
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0044
         PeakTime: 0.1974
Polo: -33.5
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.0349
     SettlingTime: NaN
      SettlingMin: -1.6189e-04
      SettlingMax: 0.0038
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0038
         PeakTime: 0.1925
Polo: -39
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.0295
     SettlingTime: NaN
      SettlingMin: -1.4013e-04
      SettlingMax: 0.0032
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0032
         PeakTime: 0.1866
Polo: -44.5
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.0255
     SettlingTime: NaN
      SettlingMin: -1.2341e-04
      SettlingMax: 0.0029
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0029
         PeakTime: 0.1821
Polo: -50
A = struct with fields:
         RiseTime: 0
    TransientTime: 1.0224
     SettlingTime: NaN
      SettlingMin: -1.1019e-04
      SettlingMax: 0.0025
        Overshoot: Inf
       Undershoot: Inf
             Peak: 0.0025
         PeakTime: 0.1787
```



Con estos resultados, se puede aportar adicionalmente a lo explicado anteriormente que, el valor en estado estacionario decrece a medida que el polo se aleja del origen. El resto de parámetros mantienen la misma dinámica de variación teniendo en cuenta la naturaleza de la entrada y la respuesta. Lógicamente, el comando *stepinfo* funciona bien para estimar los valores característicos de la respuesta.

Gráficas y parámetros característicos variables dependientes de la situación del polo simple ante una entrada rampa de pendiente 3:

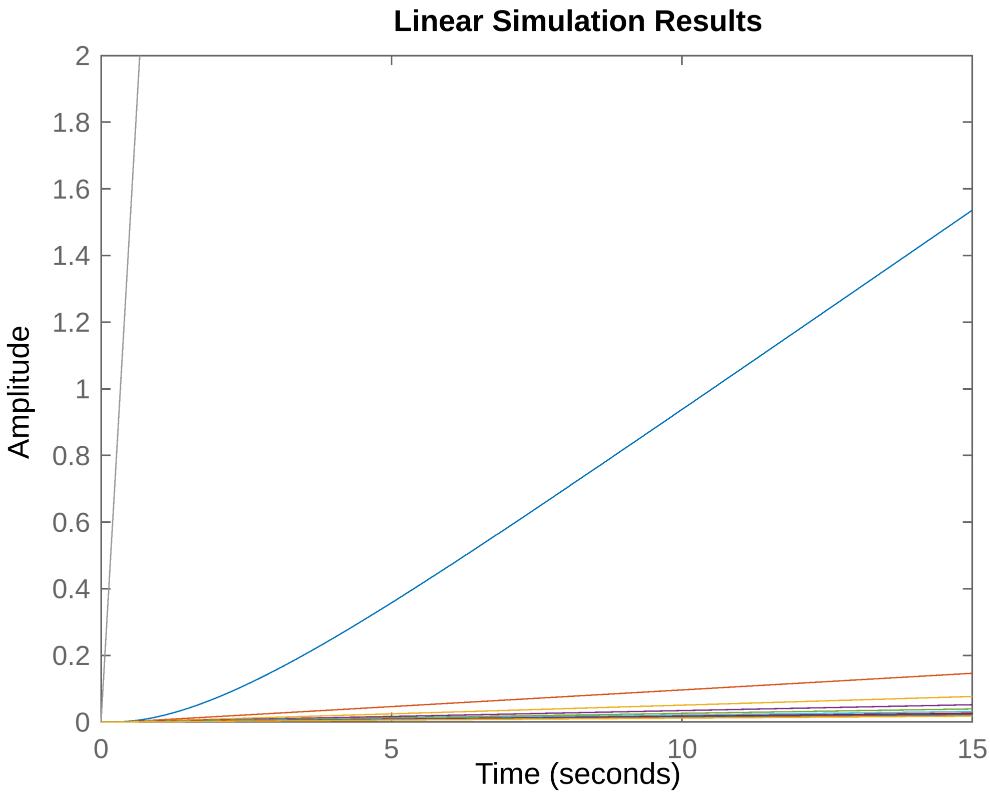
```
figure; hold on % figura donde se grafican las respuestas ante rampa
input=3*time; % vector entrada
for j=1:1:length(p)
    % Función de transferencia variable
    G(j)=zpk([],[-5+5i -5-5i p(j)],1);
    disp("Polo: "+p(j)) % se muestra por pantalla el polo bajo estudio
    % Extracción de los parámetros característicos
    A=stepinfo(3*G(j)/s) % ya que stepinfo extrae los valores de
    % la respuesta ante escalón, es necesario operar con la función de
    % transferencia. Al ser un sistema lineal, se divide por
    % la variable s para "pasar" de respuesta ante escalón a rampa
    % Gráfica de la respuesta ante rampas de pendiente 3
    lsim(G(j),input,time)
end
```

```
Polo: -0.5
A = struct with fields:
```

```
        RiseTime: NaN
    TransientTime: NaN
     SettlingTime: NaN
      SettlingMin: NaN
      SettlingMax: NaN
        Overshoot: NaN
       Undershoot: NaN
             Peak: Inf
         PeakTime: Inf
Polo: -6
A = struct with fields:
         RiseTime: NaN
    TransientTime: NaN
     SettlingTime: NaN
      SettlingMin: NaN
      SettlingMax: NaN
        Overshoot: NaN
       Undershoot: NaN
             Peak: Inf
         PeakTime: Inf
Polo: -11.5
A = struct with fields:
         RiseTime: NaN
    TransientTime: NaN
     SettlingTime: NaN
      SettlingMin: NaN
      SettlingMax: NaN
        Overshoot: NaN
       Undershoot: NaN
             Peak: Inf
         PeakTime: Inf
Polo: -17
A = struct with fields:
         RiseTime: NaN
    TransientTime: NaN
     SettlingTime: NaN
      SettlingMin: NaN
      SettlingMax: NaN
        Overshoot: NaN
       Undershoot: NaN
             Peak: Inf
         PeakTime: Inf
Polo: -22.5
A = struct with fields:
         RiseTime: NaN
    TransientTime: NaN
     SettlingTime: NaN
      SettlingMin: NaN
      SettlingMax: NaN
        Overshoot: NaN
       Undershoot: NaN
             Peak: Inf
         PeakTime: Inf
Polo: -28
A = struct with fields:
         RiseTime: NaN
    TransientTime: NaN
     SettlingTime: NaN
      SettlingMin: NaN
      SettlingMax: NaN
        Overshoot: NaN
       Undershoot: NaN
             Peak: Inf
         PeakTime: Inf
```

```
Polo: -33.5
A = struct with fields:
         RiseTime: NaN
    TransientTime: NaN
     SettlingTime: NaN
      SettlingMin: NaN
      SettlingMax: NaN
        Overshoot: NaN
       Undershoot: NaN
             Peak: Inf
         PeakTime: Inf
Polo: -39
A = struct with fields:
         RiseTime: NaN
    TransientTime: NaN
     SettlingTime: NaN
      SettlingMin: NaN
      SettlingMax: NaN
        Overshoot: NaN
       Undershoot: NaN
             Peak: Inf
         PeakTime: Inf
Polo: -44.5
A = struct with fields:
         RiseTime: NaN
    TransientTime: NaN
     SettlingTime: NaN
      SettlingMin: NaN
      SettlingMax: NaN
        Overshoot: NaN
       Undershoot: NaN
             Peak: Inf
         PeakTime: Inf
Polo: -50
A = struct with fields:
         RiseTime: NaN
    TransientTime: NaN
     SettlingTime: NaN
      SettlingMin: NaN
      SettlingMax: NaN
        Overshoot: NaN
       Undershoot: NaN
             Peak: Inf
         PeakTime: Inf
```

```
axis([0 15 0 2]) % regulación de los ejes para una mejor visualización
```

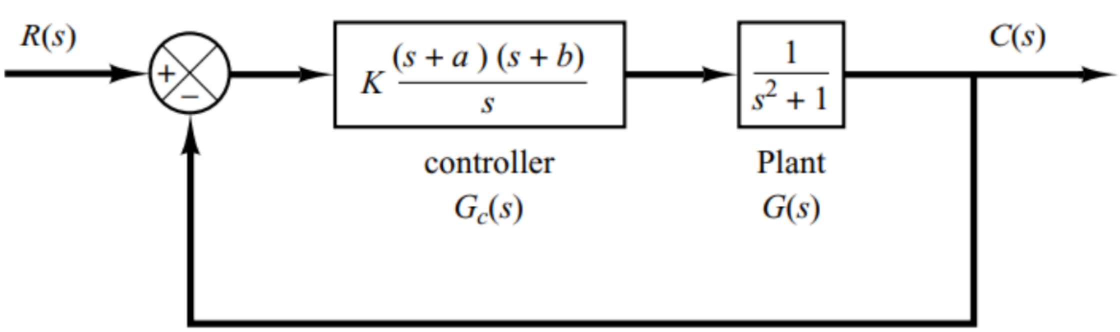



```
% de la grafica resultante
```

El comando stepinfo esta vez no funciona. No opera adecuadamente pues las respuestas temporal no paran de crecer y nunca se estabilizan. Nótese que las dinámicas temporales son análogas a las obtenidas con las respuestas impulsionales.

Ejercicio 3 (2,5 puntos). Respuesta transitoria de sistemas de orden superior.

Considere el sistema que se muestra en la figura inferior:



Se desea diseñar un controlador de tal forma que los polos dominantes en lazo cerrado estén ubicados en $s = -1 \pm j\sqrt{3}$. Para facilitar el proceso, se indica que para el controlador, seleccione a=1 y K=2,33. Por tanto, determine tan solo el valor de b. Dibuje el mapa de polos y ceros, a modo de verificación.

Razone su respuesta, especificando por qué un cero del controlador puede fijar los polos y variar el tipo de respuesta en lazo cerrado.

Función de transferencia de la planta:

```
G=tf(1,[1 0 1]);
```

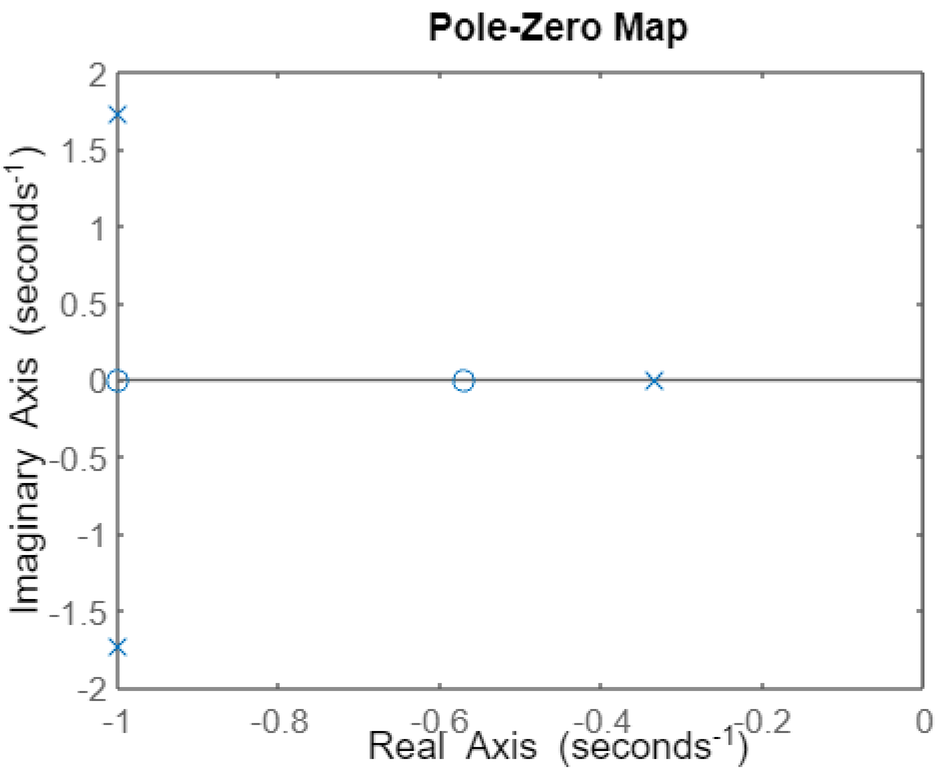
Se declaran las constantes, dadas en el enunciado del problema, y que determinan un cero y la ganancia del controlador:

```
a=1; K=2.33;
```

Bucle que escanea el valor del cero final del controlador para obtener los polos del sistema en lazo cerrado deseados. Nótese que se considera el valor de b, entre 0 y 10, como aproximación por el valor de los otros parámetros ya definidos.

```
for b=0:0.01:10
    draw=false; % variable auxiliar que determina si se muestran los resultados
    % por pantalla
    C=zpk([-a -b],0,K); % definición del controlador
    FTlc=feedback(G*C,1); % función de transferencia en lazo cerrado
    polos=pole(FTlc); % cálculo de los polos del sistema
    % Escaneo de todos los polos del sistema en bucle cerrado
    for i=1:1:length(polos)
        % Si algún polo se encuentra, dentro de un rango de aproximación
        % del 0.2%, en la localización deseada, se modifica la variable
        % auxiliar (nótese que se considera el polo con parte imaginaria
        % positiva)
        if real(polos(i))>(-1.002) && real(polos(i))<(-0.998) && ...
                imag(polos(i))<1.732 && imag(polos(i))>1.728
            draw=true;
        end
    end
    % Se muestran los datos por pantalla en caso de que se haya hallado
    % el valor del polo resultante
    if draw==true
        disp("Valor de b: "+b) % valor de b (cero del controlador)
        disp("Polos: "+polos) % polos del sistema en lazo cerrado
        figure; pzmap(FTlc); % gráfica de los polos y ceros del sistema
        % con el controlador final, a modo de comprobación
    end
end
```

```
Valor de b: 0.57
    "Polos: -0.33253+0i"
    "Polos: -0.99873+1.731i"
    "Polos: -0.99873-1.731i"
```



Tenga en cuenta que el cero del controlador puede modificar los polos del sistema en bucle cerrado porque el denominador de la función de transferencia global es 1+C(s)*G(s). El numerador de C(s) constituirá, de alguna forma, el denominador de la función de transferencia en lazo cerrado y, por tanto, determinará la situación y valor de los polos del sistema.

<u>Ejercicio 4 (1,75 puntos).</u> Precisión y error en estado estacionario de un sistema de control.

Obtenga la respuesta de un sistema con función de transferencia en lazo cerrado $\frac{C(s)}{R(s)} = \frac{5}{s^2+s+5}$ ante una entrada $r(t) = 2 + t$. Después, cuando se introduce $r(t) = \frac{1}{2}t^2$. Especifique el tipo de sistema en ambos escenarios, en función del error de control. Razone la respuesta.

En primer lugar, se declara la función de transferencia del sistema en lazo cerrado:

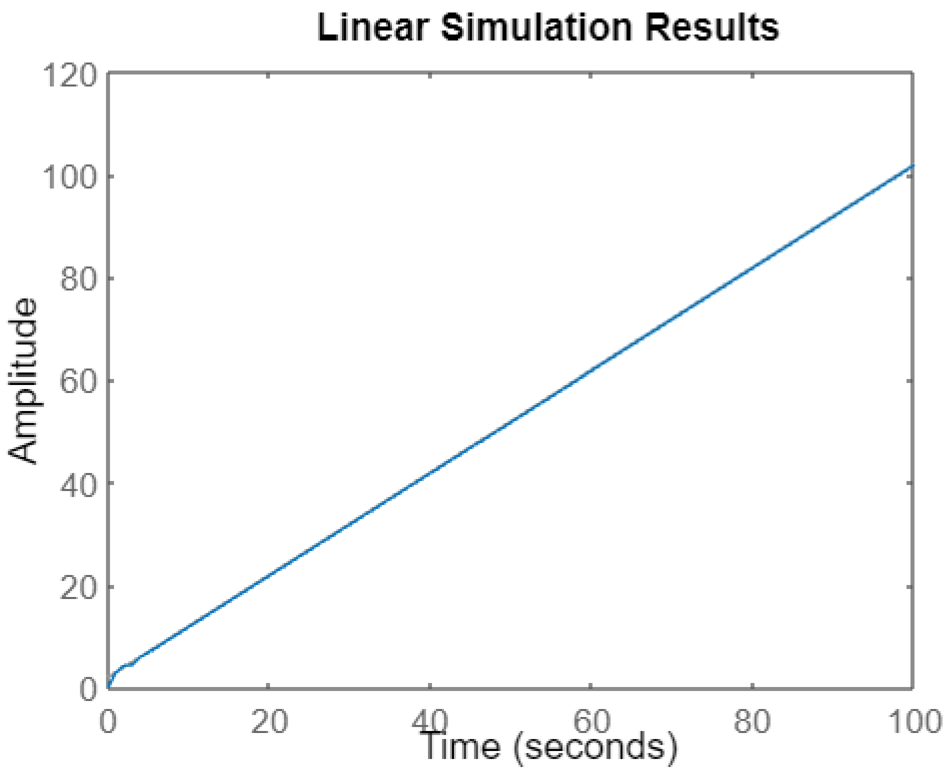
```
G=tf(5,[1 1 5]);
```

Después, el vector temporal y la entrada rampa:

```
time=linspace(0,100,100);
rampa=2+time;
```

Se llama a una nueva figura y se grafica la respuesta ante una entrada rampa:

```
figure; lsim(G,rampa,time)
```



Se graban los vectores tiempo y salida de la respuesta obtenida previamente:

```
[y,t]=lsim(G,rampa,time);
```

Se calculan los errores absoluto y relativo:

```
ess_vel_absoluto=rampa(end)-y(end)
```

```
ess_vel_absoluto = 0.2000
```

```
ess_vel_relativa=(rampa(end)-y(end))*100/rampa(end)
```

```
ess_vel_relativa = 0.1961
```

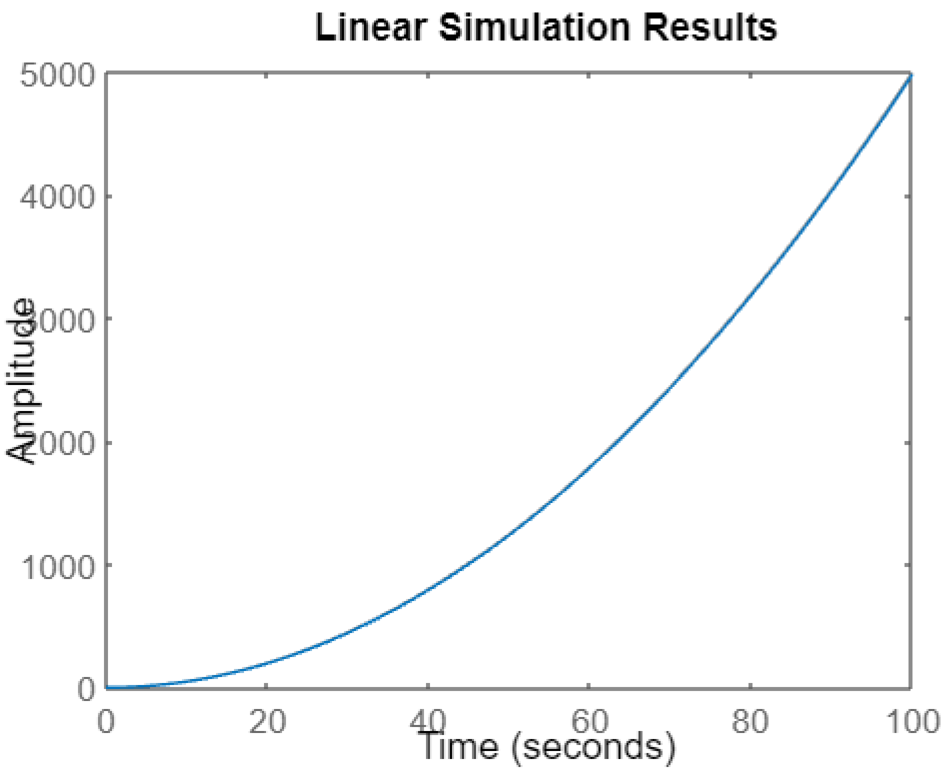
Ahora, la entrada es una rampa con un offset de valor 2. En principio, parece que no se obtiene error, por lo que apunta a un sistema tipo 2. Lo comprobamos en la segunda parte del ejercicio.

Se define la parábola:

```
parabola=(1/2)*time.^2;
```

Se llama a una nueva figura y se grafica la respuesta ante la parábola:

```
figure; lsim(G,parabola,time)
```



Se guardan los valores salida y tiempo, para posteriormente calcular los errores.

```
[y,t]=lsim(G,parabola,time);
ess_vel_absoluto=parabola(end)-y(end)
```

```
ess_vel_absoluto = 20.0670
```

```
ess_vel_relativa=(parabola(end)-y(end))*100/parabola(end)
```

```
ess_vel_relativa = 0.4013
```

En efecto, se tiene un sistema tipo 2, con un error finito muy bajo ante la entrada parábola.