LABORATORIO de INGENIERÍA de CONTROL-2

Práctica 7.

Realización programada de sistemas digitales

El paso final tras el diseño de un regulador discreto G(z) es su implementación por medio de un programa que se ejecuta en un sistema digital con acceso a convertidores A/D y D/A.

El programa es una rutina cíclica, que se dispara cada periodo T (garantizado en un entorno de tiempo real) y que debe incluir ciertas instrucciones de cálculo, en el orden adecuado.

El código contendrá como primera instrucción una lectura de la entrada 'u' y más adelante una escritura de la salida 'y'. El conjunto de instrucciones entre lectura y escritura pueden llamarse fase de *'preproceso'*, y las posteriores a la escritura, fase de *'postproceso'*.

En clase de teoría se mostró cómo hacer programas basados en formas canónicas. En esta práctica se explicarán dos recursos adicionales y se planteará un trabajo de realización.

Programación directa de la ecuación en diferencias

Aunque la realización basada en modelos de estado y formas canónicas ofrece mayor versatilidad, también puede hacerse un programa basado en transcribir directamente la ecuación en diferencias asociada a G(z). Vamos a verlo a través de un ejemplo. Considerar

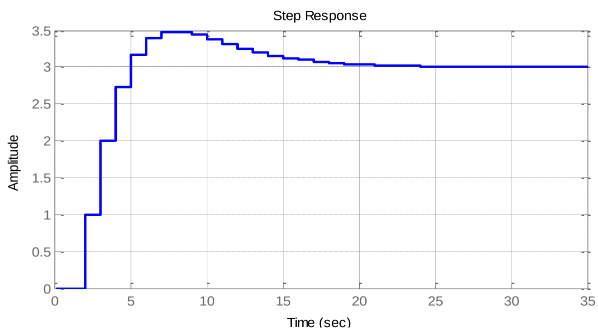
G = (z–0.85) /(z–0.75) /(z–0.6) /(z–0.5) =

$$\frac{z - 0.85}{z^3 - 1.85\,z^2 + 1.125\,z - 0.225} \qquad \frac{z^{-3}}{z^{-3}} = \frac{z^{-2} - 0'85\,z^{-3}}{1 - 1'85\,z^{-1} + 1'125\,z^{-2} - 0'225\,z^{-3}}$$

Su ecuación en diferencias es (dividir entre z^3 el numerador y denominador):

y(k) = 1.85 y(k–1) –1.125 y(k–2) + 0.225 y(k–3) + u(k–2) –0.85 u(k–3)

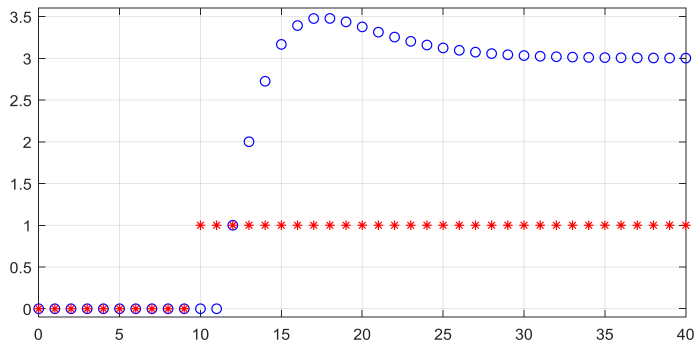
La respuesta a escalón sale (>>step(G))



La programación de la Ec.Dif. requiere usar 4 variables que representen a la salida actual y(k) junto con las 3 salidas previas y(k−1), y(k–2), y(k–3), más otras 4 variables para hacer lo mismo con la entrada u(k). Recordar que el programa debe basarse en variables escalares (no se pueden usar vectores). Un código posible sería:

```
uk1=0; uk2=0; uk3=0;  % Inicialización
yk1=0; yk2=0; yk3=0;
T=0.25; N=40; % periodo aprox. T,  N ciclos
plot(0,yk1,'oblue',0,uk1,'*red');  % gráfica
axis([0, N, -0.1, 3.6]);  grid;   hold on;
tic; k=1;          % inicio temporizador
while k<=N,
  pause(T);
  uk=  ( k>=10 );            %%%%'Lectura' Entrada
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  yk= ...                    % Cálculo Salida y(k)
   1.85*yk1 -1.125*yk2 + 0.225*yk3...
              +uk2 - 0.85*uk3 ;
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  plot(k,yk,'oblue',k,uk,'*red'); %'Escritura'Sal.
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  k=k+1;        % Actualizaciones Estados Internos
  yk3=yk2;  yk2=yk1;  yk1=yk;
  uk3=uk2;  uk2=uk1;  uk1=uk;
end;
toc        % fin temporizador
hold off;
```

La salida gráfica resultante (respuesta a escalón entre 0 y 1 aplicado en t=10) es:



Leer en detalle las instrucciones, entendiendo qué es lo que hacen:

- Temporización. ¿Cómo se aproxima el periodo T? (pause, tic, toc,…)
- Gestión de E/S ¿Cuál es la entrada? ¿Cómo es la salida gráfica?
- Fase de preproceso (cálculo de la salida)
- Fase de postproceso (actualizaciones) ¿Puede simplificarse esta fase?

La comprobación final consiste en ver que son iguales la respuesta a escalón {y(k)} obtenida por programa (última gráfica) y la respuesta a escalón del modelo (>>step(G), página anterior). Verificar las muestras del retardo inicial (2 periodos), la fase de subida, el tiempo de pico y la SO.

Programación en paralelo

En muchos casos se prefiere realizar G(z) en base a alguna descomposición suma G(z)=Ga(z)+Gb(z). Por ej., en los reguladores PID es mejor tener por separado las señales de las tres ramas en paralelo (para monitorizar o supervisar la derivada o la integral del error).

Teniendo en cuenta que los bloques en paralelo comparten la entrada u(k) y suman las salidas y(k)=ya(k)+yb(k), es fácil componer el código completo a base de códigos parciales:

Subsistema Ga(z)	Subsistema Gb(z)	Sistema total G(z)=Ga(z)+Gb(z)
Leer u	Leer u	Leer u
Cálculo ya	Cálculo yb	Cálculo ya
Escribir ya	Escribir yb	Cálculo yb
Postproceso-a	Postproceso-b	Escribir y= ya+yb
		Posproceso-a
		Posproceso-b

Para que el código completo funcione, las instrucciones de preproceso (cálculo de salida) y postproceso de cada subsistema **no deben contener variables con el mismo nombre**.

$Ga: \frac{-42}{z-0'5} = \frac{-42z^{-1}}{-0'5z^{-1}+1}$ $\left\{ \begin{array}{l} Y(k) - 0'5\,Y(k-1) = -42\,U_{k-1} \\ Y(k) = 0'5\,Y_{k-1} - 42\,U_{k-1} \end{array} \right.$

Trabajo de Laboratorio. Parte 1

Considerar el sistema G(z) = Ga(z) + Gb(z) + Gc(z) donde:

```
Ga =  -42/(z-0.5);
Gb =   50/(z-0.6);
Gc =  -8/(z-0.75);
```

$\frac{42z - 32'7}{z^2 - 1'35z + 0'45} = \frac{42z^{-1} - 32'7z^{-2}}{1 - 1'35z^{-1} + 0'45z^{-2}}$

$Y_k - 1'35\,Y_{k-1} + 0'45\,Y_{k-2} = 42\,U_{k-1} - 32'7\,U_{k-2}$

$Y_k = 1'35\,Y_{k-1} - 0'45\,Y_{k-2} + 42\,U_{k-1} - 32'7\,U_{k-2}$

Se pide hacer una realización basada en suma de un sistema de primer orden más uno de segundo orden. El primer orden se programará directamente (Ec.Dif.) y el segundo orden mediante FCC, FCO o Ec.Dif. En concreto, según salgan los enteros randómicos r1,r2:

r1=1:	Primer orden Ga	Segundo orden: Gbc= Gb + Gc
r1=2:	Primer orden Gb	Segundo orden: Gac= Ga + Gc
r1=3:	Primer orden Gc	Segundo orden: Gab= Ga + Gb

r2=1: Programar el segundo orden en base a FCC
r2=2: Programar el segundo orden en base a FCO
r2=3: Programar el segundo orden en base a la Ec.Dif.

Notación: La entrada será ‘u’
Las salidas serán ‘y’ (por ejemplo ya, ybc, etc…)
Las variables internas o auxiliares podrán nombrarse libremente
Los coeficentes deberán programarse numéricamente (no ‘ai’,’bj’, etc.)

Parte 2. Realización programada de PIDs

Considerar el controlador PID (con derivada filtrada) definido en tiempo continuo como C(s):

```
Kp=2.5;  Ki=3.5;  Kd=0.2;   Tf=0.05;
C = Kp +  Ki/s +  Kd*s/(1+ Tf*s);
```

$C = 2'5 + \frac{3'5}{s} + \frac{0'2s}{1+0'05s}$ 0'2

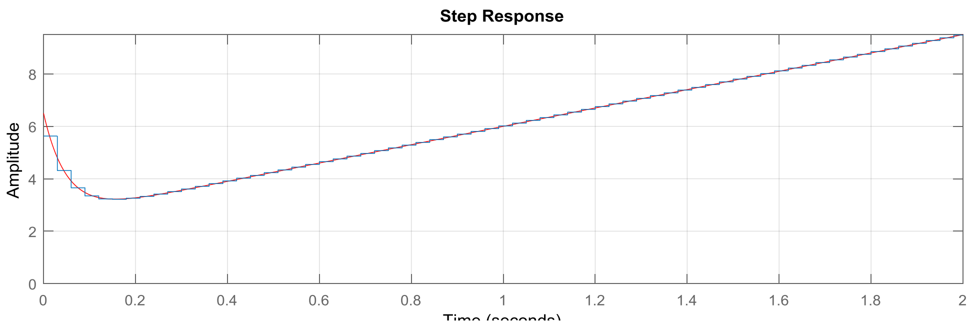
Se quiere realizar digitalmente, discretizando por separado las tres ramas en P+I+D (descomposicón en paralelo). En concreto, según sean los enteros randómicos r3,r4:

Discretización I. r3=1: Deriv. hacia atrás; r3=2: Deriv. hacia adelante; r3=3: Tustin

Discretización D. r4=1: Deriv. hacia atrás; r4=2: Deriv. hacia adelante; r4=3: Tustin

$\frac{z-1}{Tz}$

Obtener el regulador discretizado G(z), en función del periodo de muestreo Ts. Elegir un periodo pequeño, p.ej. Ts= 0.03 seg. Escribir y verificar el código de realización paralela. Aunque el PID se usa en lazo cerrado, el parecido entre C(s) y G(z) puede verificarse a través de sus respuestas a escalón en lazo abierto, en un rango finito p.ej. [0, 2]seg.:



Hoja de resultados:

Apellidos y nombre: DNI=

r1= r2= r3= r4=

Parte 1. Entregar el código programado y una figura aproximada a mano, con la respuesta escalón del programa.

Parte 2. Entregar el código programado y una figura aproximada a mano con la respuesta a escalón del programa. Incluir también la expresión del PID discreto G(z), en forma P+I+D, como función de Ts

(máximo 2 carillas)