Diseño de controlador IMC para implementación discreta (II): diseño directo en tiempo discreto

Presentación en vídeo: http://personales.upv.es/asala/YT/V/imcdt2.html

Este código funcionó correctamente con Matlab **R2021b**

Objetivo: Comparar varias opciones de diseño de reguladores IMC para ser implementados por computador (tiempo discreto).

Tabla de Contenidos

Modelo

```
Ac=[0 1 ;-3 -1.1]; Bc=[0;3]; C=[1 0];
sysc=ss(Ac,Bc,C,0);sysc.InputName='ureal';sysc.OutputName='y';
gan=dcgain(sysc)
```

```
gan = 1
```

```
tf(sysc)
```

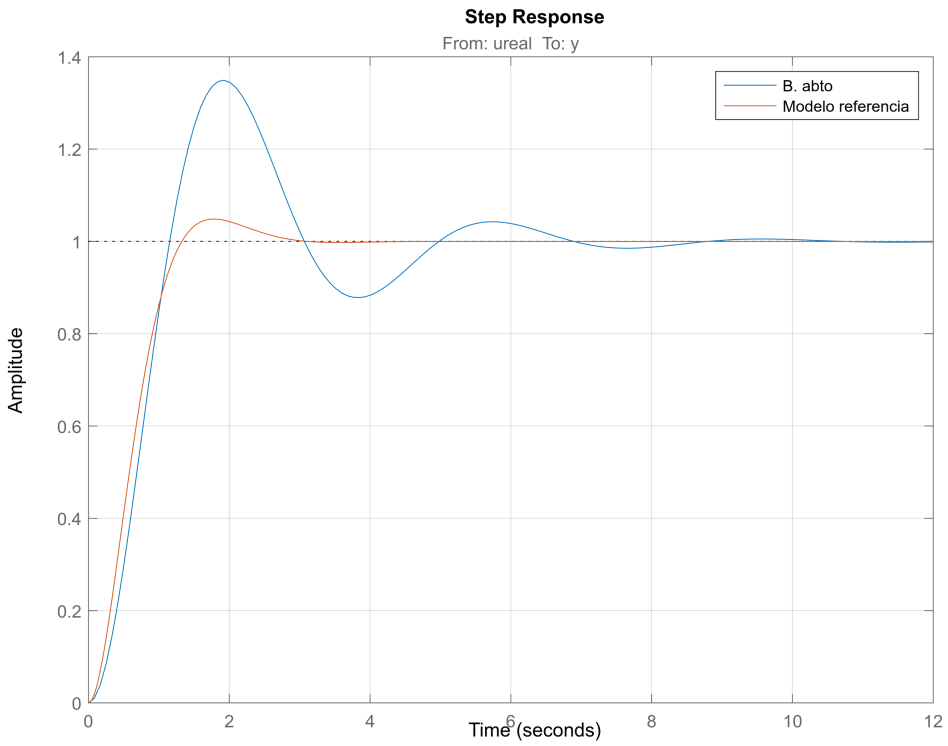
```
ans =

  From input "ureal" to output "y":
         3
  ---------------
  s^2 + 1.1 s + 3

Continuous-time transfer function.
```

Objetivos de control

```
s=tf('s');
ModeloRefContinuo=6/(s^2+3.4*s+6);
step(sysc,ModeloRefContinuo), grid on, legend("B. abto","Modelo referencia")
```



Diseño de referencia: control IMC en tiempo contínuo

```
Qc=minreal(ModeloRefContinuo/sysc); zpk(Qc) %Q cancela la planta y la sustituye por otr
```

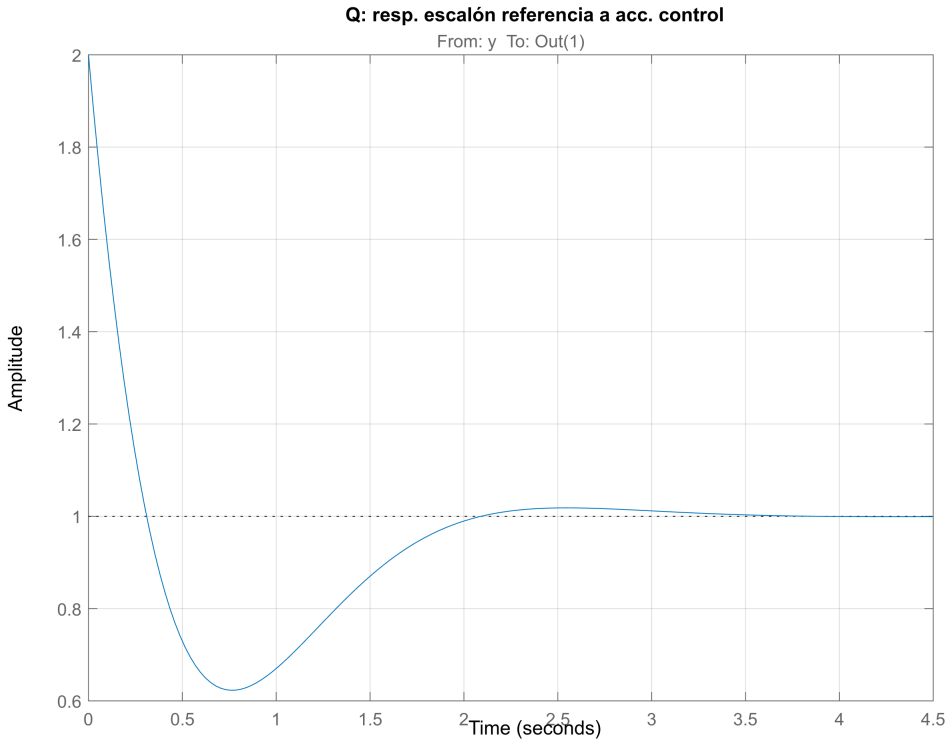
```
3 states removed.

ans =

  From input "y" to output:
  2 (s^2 + 1.1s + 3)
  ------------------
    (s^2 + 3.4s + 6)

Continuous-time zero/pole/gain model.
```

```
step(Qc), grid on, title("Q: resp. escalón referencia a acc. control")
```



A partir de Q (parámetro de Youla), formamos el regulador "convencional":

```
Kc=minreal(feedback(Qc,-sysc));
```

```
2 states removed.
```

```
Kc.InputName='e'; Kc.OutputName='u';
zpk(Kc)
```

```
ans =

  From input "e" to output "u":
  2 (s^2 + 1.1s + 3)
  ------------------
      s (s+3.4)

Continuous-time zero/pole/gain model.
```

Selección período de muestreo

Nuestro objetivo será $t_{est} \approx 2$ s, $\omega_{propia} = 1.76$ rad/s, $\omega_n = \sqrt{6} = 2.45$ rad/s.

Períodos bien elegidos serían:

```
%Ts=0.1;
```

● todo sale bien, quizás elegir esto si vamos a usar discretización de diseño continuo, por ir "a lo seguro" o si hubiera mucho "ruido de alta frecuencia" en las lecturas de sensores (incluso podría ser necesario un T_s menor para filtrar mejor si hay mucho ruido). No es el caso aquí. Comprueba tú mismo que funciona.

```
Ts=0.25;
```

●también razonable, hasta aquí llegaríamos con las reglas heurísticas de "andar por casa" usuales con un sensor "suficientemente limpio": $\geq$10 o 15 muestras en t_{est}, $\geq$ 6 u 8 muestras en ω_{propia}, ≥ 5 muestras en t_{subida}, resonancias de bucle abto y bucle cerrado a 1/5 de frecuencia de Nyquist π/T_s, ...

Otras opciones menos aconsejables "intuitivamente" (porque hay "pocas muestras" en tiempo de subida/establecimento/frecuencia propia, porque se "mide poco" y las perturbaciones de alta frecuencia darán "aliasing", etc...), aunque en simulación "ideal" sin ruido ni error de modelado, todo va bien en discreto, pero seleccionar estos períodos de muestreo es seguramente un "ejercicio teórico" que no sería aconsejable en la práctica del control de procesos industriales reales.

```
%Ts=0.8;
%Ts=2.4;
```

Diseño IMC discreto directo

Discreticemos el proceso:

```
sysd=c2d(sysc,Ts,'zoh');
zpksysd=zpk(sysd)
```

```
zpksysd =

  From input "ureal" to output "y":
    0.08441 (z+0.9119)
  -----------------------
  (z^2 - 1.598z + 0.7596)

Sample time: 0.25 seconds
Discrete-time zero/pole/gain model.
```

El cero "de muestreo" (z+0....) será importante.

```

```

Y pensemos en un modelo de referencia, también con zoh:

```
ModeloRef1=c2d(ModeloRefContinuo,Ts,'zoh');
mzpkr=zpk(ModeloRef1)
```

```
mzpkr =

    0.13981 (z+0.7517)
  -----------------------
  (z^2 - 1.183z + 0.4274)

Sample time: 0.25 seconds
Discrete-time zero/pole/gain model.
```

```

```

Cancelar el cero de muestreo del proceso G no es satisfactorio (oscilaciones en "u"), y cambiarlo desde `sysd` a donde está en ModeloRef lo hará $Q = G^{-1}M$. Planteamos mantener el cero de muestreo original de `sysd`, y no introducir el de `ModeloRef1`:

```
z=tf('z',Ts);
Corrector=tf(1)*(z-zpksysd.Z{:})/(z-mzpkr.Z{:}) %probar con/sin este "Corrector"
```

```
Corrector =

  z + 0.9119
  ----------
  z + 0.7517

Sample time: 0.25 seconds
Discrete-time transfer function.
```

*A períodos grandes, los ceros de muestreo no están tan cerca del "borde" del círculo unidad y, por tanto, no son una resonancia tan "patente"... podría prescindirse del "Corrector".

```
ModeloRef=minreal(ModeloRef1*Corrector/dcgain(Corrector));
dcgain(ModeloRef)
```

```
ans = 1.0000
```

```
zpk(ModeloRef)
```

```
ans =

    0.12809 (z+0.9119)
  -----------------------
  (z^2 - 1.183z + 0.4274)

Sample time: 0.25 seconds
Discrete-time zero/pole/gain model.
```

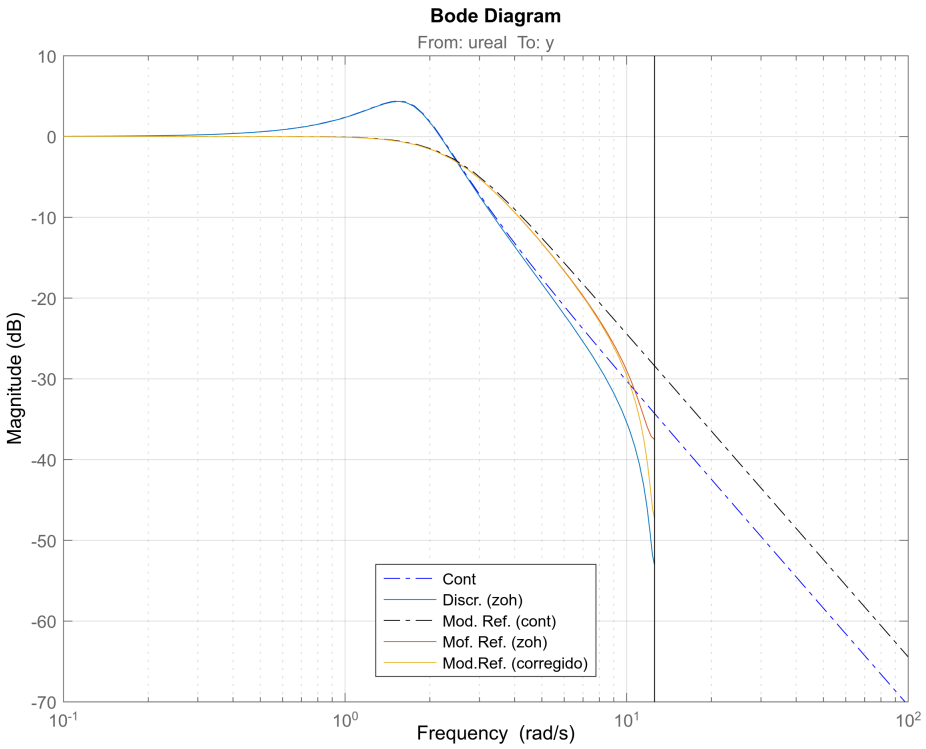
Hemos añadido un cero a "casi" la frecuencia de Nyquist para asegurar que no hay actividad significativa a esa frecuencia, para no cancelar el cero "de muestreo" del proceso.

```
pole(ModeloRef)
```

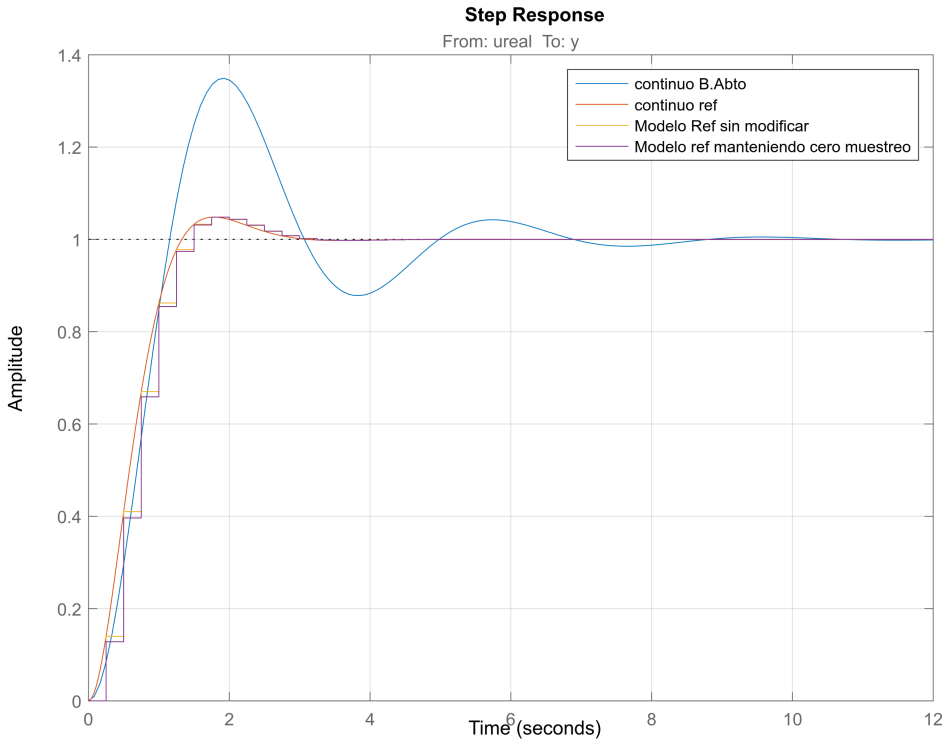
```
ans = 2×1 complex
   0.5913 + 0.2790i
   0.5913 - 0.2790i
```

Veamos una comparativa de todos los modelos en bucle abierto y los objetivos de bucle cerrado

```
bodemag(sysc,'-.',sysd,ModeloRefContinuo,'-.k',ModeloRef1,ModeloRef), grid on, legend('
```




```
step(sysc,ModeloRefContinuo,ModeloRef1,ModeloRef), grid on, legend("continuo B.Abto","c
```




```
Q=minreal(ModeloRef/sysd);
```

```
3 states removed.
```

```
zpk(Q)
```

```
ans =

  From input "y" to output:
  1.5175 (z^2 - 1.598z + 0.7596)
  ------------------------------
     (z^2 - 1.183z + 0.4274)

Sample time: 0.25 seconds
Discrete-time zero/pole/gain model.
```

El regulador IMC será:

```
K_IMC=minreal(feedback(Q,-sysd));size(K_IMC)
```

```
2 states removed.
State-space model with 1 outputs, 1 inputs, and 2 states.
```

```
zpk(K_IMC)
```

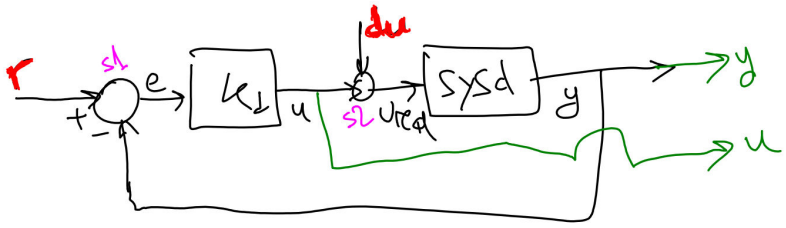
```
ans =

  From input "y" to output:
  1.5175 (z^2 - 1.598z + 0.7596)
  ------------------------------
         (z-1) (z-0.3106)
```

```
Sample time: 0.25 seconds
Discrete-time zero/pole/gain model.
```

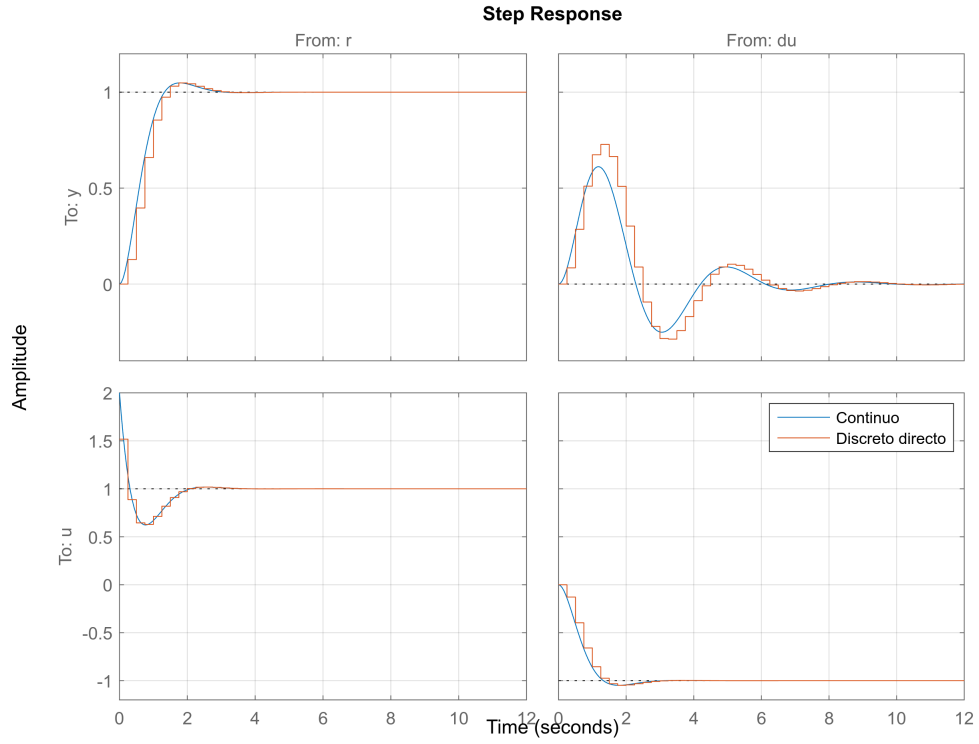
Prestaciones en bucle cerrado

Prestaciones en tiempo discreto

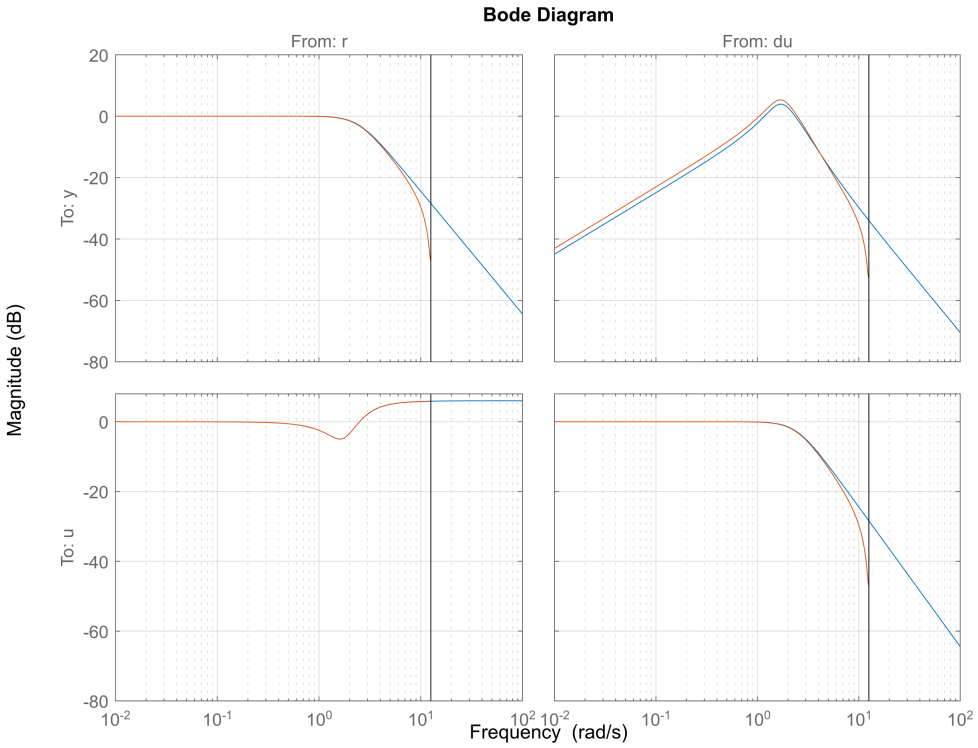


Cerramos bucle con "connect", y podemos hacer `step`, `bode`, `lsim`, `initial`...

```
K_IMC.InputName='e';
K_IMC.OutputName='u';
s1=sumblk('e=r-y');s2=sumblk('ureal=u+du');
BCcont=connect(s1,s2,Kc,sysc,{'r','du'},{'y','u'});
BC=connect(s1,s2,K_IMC,sysd,{'r','du'},{'y','u'});
step(BCcont,BC), grid on, legend("Continuo","Discreto directo")
```

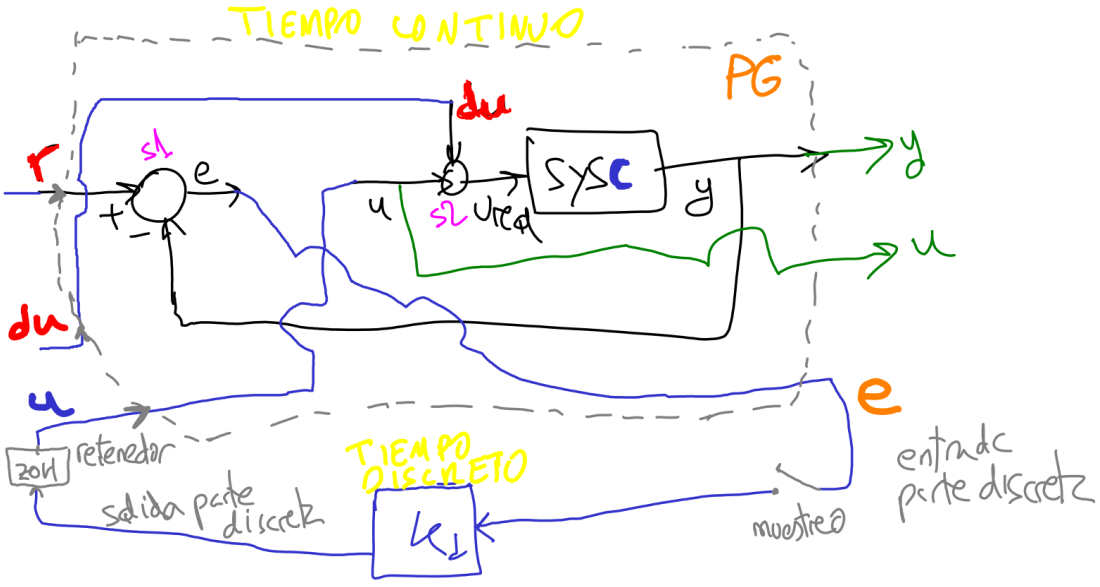



```
bodemag(BCcont,BC), grid on
```

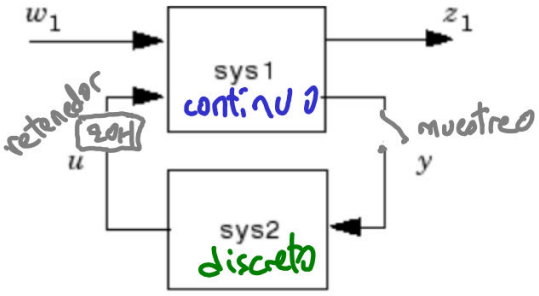


Prestaciones intermuestreo

El proceso controlado es realmente continuo, `sdlsim` puede simular respuesta intermuestreo.



de modo que es una interconexión "lower Linear Fractional Transformation (LFT)" entre un sistema contínuo y uno discreto:



Siendo $w_1 = (r, d_u)$ y $z_1 = (y, u)$... aunque, como u es realmente un tren de escalones (es una señal discreta retenida), no necesitamos sacarlo con sdlsim porque será idéntica a la simulación discreta (*ante entrada escalón)... por tanto, en lo que sigue dejaremos $z_1 \equiv y$.

El sistems "2" tiene un único elemento (el regulador discreto), el sistema 1 tiene un diagrama de bloques, que podemos simplificar con el comando "`connect`":

```
PG=connect(s1,s2,sysc,{'r','du','u'},{'y','u','e'});
size(PG)
```

```
State-space model with 3 outputs, 3 inputs, and 2 states.
```

Discretizamos zoh para simulación discreta (aproximada porque las entradas son continuas, sólo exacta si r o d_u son constantes entre muestreos)

```
PGd=c2d(PG,Ts,'zoh');%para simulación puramente discreta, si queremos...
```

La simulación de 10 segundos es:

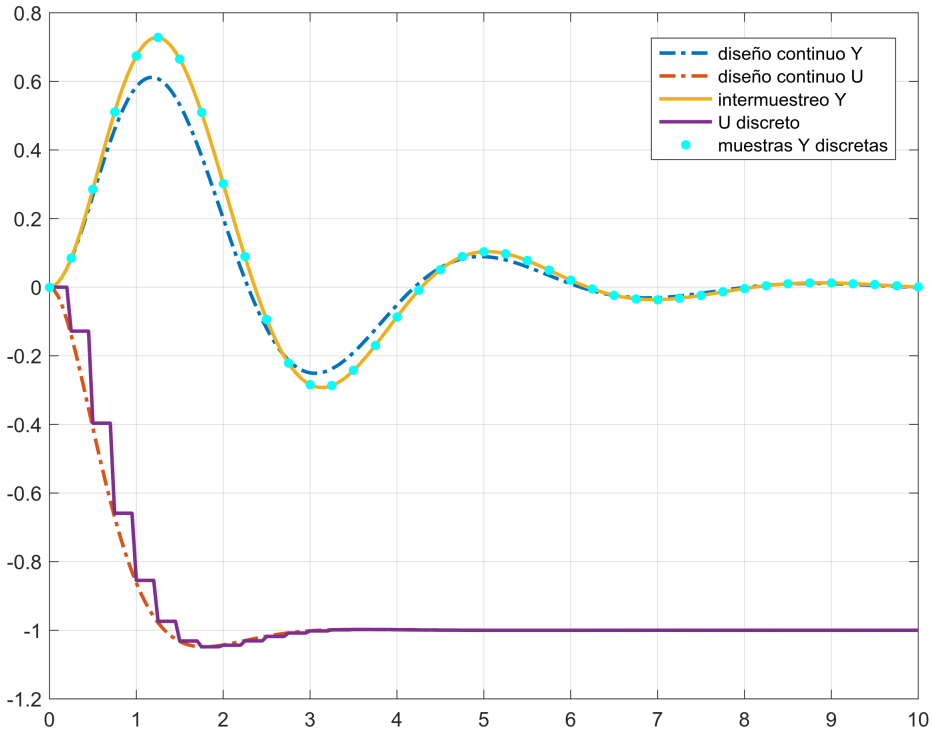
```
Tsimc=0:0.05:10; %tiempos de "interpolación" continuos deseados
Tsimd=0:Ts:10; %tiemos de muestreo discretos
```

Entrada escalón:

```
UUc=ones(length(Tsimc),1)*[0 1]; %entradas generalizadas continuas (interpoladas): no r
UUd=ones(length(Tsimd),1)*[0 1]; %entradas sólo en instantes muestreo para simulación d

Y0=lsim(lft(PG,Kc),UUc,Tsimc);   %simulación todo en tiempo continuo
Y1=sdlsim(PG,K_IMC,UUd,Tsimd);   %simulación HÍBRIDA (contínuo+discreto: sampled-data)
Y1d=lsim(lft(PGd,K_IMC),UUd,Tsimd); %simulación tiempo DISCRETO

plot(Tsimc,Y0,'-.',LineWidth=2), grid on, hold on
plot(Y1{1},Y1{2},LineWidth=2),
plot(Tsimd,Y1d(:,1),'.c',MarkerSize=16), hold off
legend("diseño continuo Y","diseño continuo U","intermuestreo Y","U discreto","muestras
```



*Ante perturbaciones en la entrada que no sean "constantes entre muestreos" podría haber efecto "alias" apreciable a períodos de muestreo grande, no objetivo de este material.