Ejemplo de estimación estocástica de funciones de transferencia

Parte 1: Estimación de un sistema con datos obtenidos de una simulación

Este ejemplo mostrará la forma de uso de la función de estimación *stochastic2* [1] y la forma de evaluar si la estructura del modelo estimado es adecuada. El método se basa en la estimación de un modelo ARX [2] por mínimos cuadrados (LS) y uso de la varianza del error de los parámetros para la evaluación de resultados.

Por tratarse de un ejemplo en *Notebook* de Matlab [3], se puede cambiar el sistema de entrada; el periodo de muestreo; la forma de la señal de excitación de entrada; la duración de la simulación, etc. y recalcular el libro completamente o por partes.

```
clear all
warning off
```

Definimos un sistema ejemplo para la estimación

```
s = zpk('s');
sys1 = 1/((s+2)*(s*0.5))
```

```
Zero/pole/gain:
   2
-------
s (s+2)
```

Creamos el vector de tiempo con una duración de 20 segundos y tiempo de muestreo de 0.05s.

```
T = 0.05;
t = 0: T : 20;
```

Creamos una señal de entrada, con una duración igual a 20 segundos, o varios escalones para simular una señal estocástica

```
escalon = ones(1,length(t));
escalon(1) = 0;
for i = 100:150 escalon(i) = 0; end
for i = 220:250 escalon(i) = 0; end
for i = 290:360 escalon(i) = 0; end
```

Simulamos el sistema con la señal de entrada estocástica y graficamos la entrada y la salida simulada

```
y = lsim(sys1,escalon,t);
plot(t,escalon,t,y)
y = y';
```

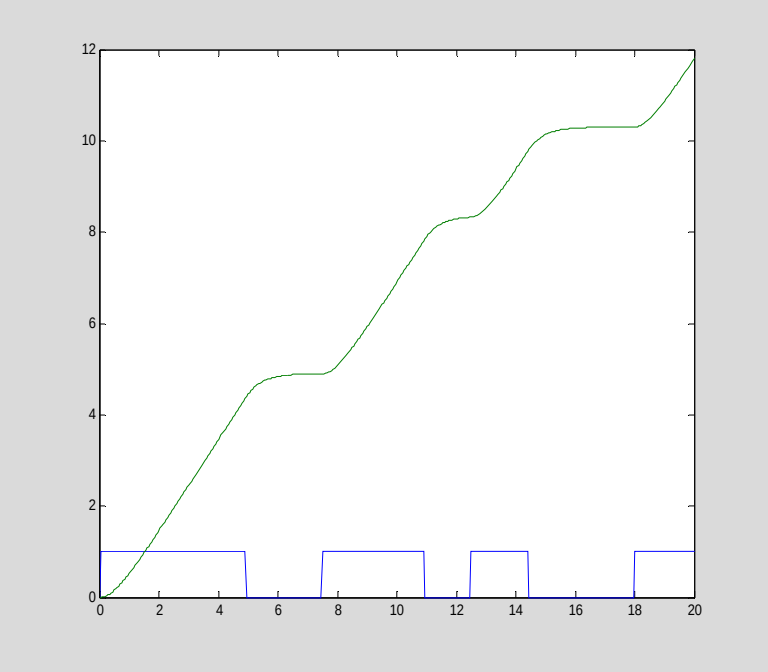

Figura 1: Señal de excitación y respuesta obtenida

Utilizamos el estimador para estimar los parámetros con una estructura del modelo de orden 2; pero, primero actualizamos la ruta hacia la carpeta conteniendo los programas.

```
path('D:\Matlab\R2006a\work\stochastic2',path);

[be,d,funcion2] = stochastic2(y,escalon,T,2,2);

La matriz varianza-covarianza obtenida es:
  1.0e-017 *
    0.3179   -0.3181    0.0016   -0.0135
   -0.3181    0.3184   -0.0016    0.0135
    0.0016   -0.0016    0.0029   -0.0029
   -0.0135    0.0135   -0.0029    0.0034

Los coeficientes estimados +/- su incertidumbre:
     1.9048e+000     3.5051e-009
  -904.8374e-003     3.5077e-009
     2.4187e-003   331.8989e-012
     2.3394e-003   364.2544e-012

La función de transferencia obtenida es:

Zero/pole/gain:
0.0024187 (z+0.9672)
--------------------
  (z-1) (z-0.9048)

Sampling time: 0.05
```

En el modelo estimado podemos observar que es tipo 1, como la planta original. Simulamos el sistema original y el sistema estimado para comparar resultados. Como

puede observarse en la figura 2, ambas curvas se superponen completamente, por lo que el modelo estimado es satisfactorio, algo que podíamos esperar al examinar la incertidumbre de los coeficientes estimados que está por debajo de $1x10^{-8}$.

```
close
step(sys1,funcion2,20)
```

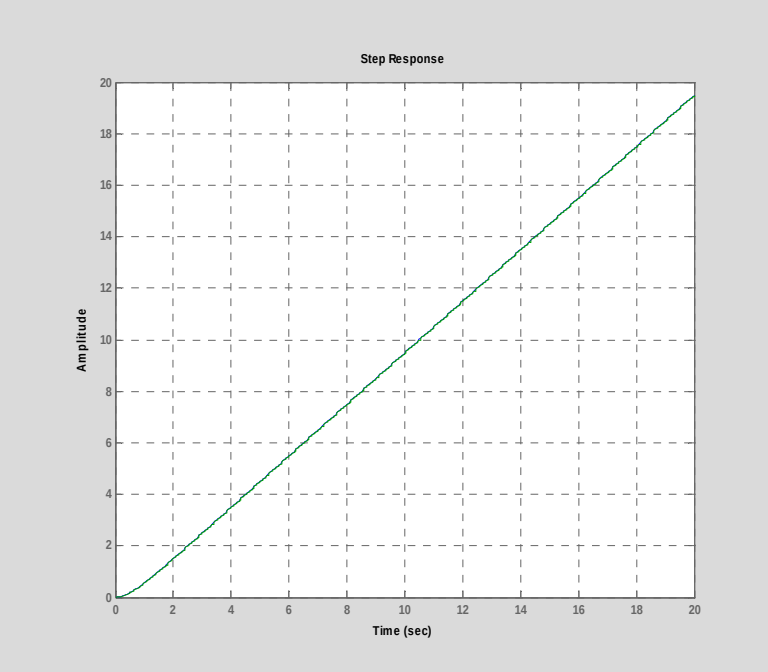

Figura 2: Respuesta ante escalón del sistema original y del sistema estimado orden 2

Una simulación de 0 a 2 segundos confirma también que las características dinámicas rápidas fueron estimadas adecuadamente. Podemos observar en la figura 3 que la respuesta del modelo discreto estimado es exactamente la misma que correspondería a un modelo discreto obtenido por la aproximación con retenedor de orden cero.

```
close
step(sys1,funcion2,2)
```

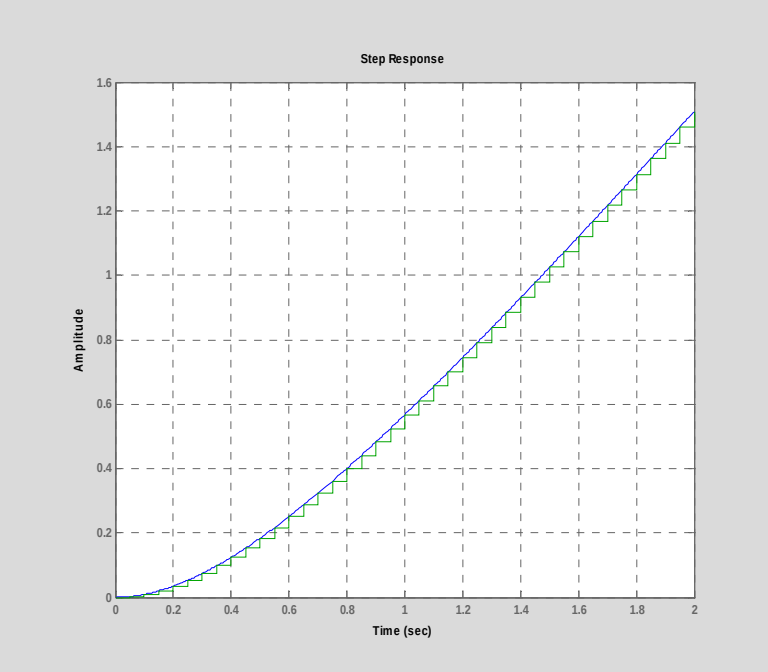

Figura 3: Respuesta transitoria entre 0 y 2 segundos del sistema estimado

Convertimos el modelo en tiempo discreto obtenido por el estimador, en un modelo continuo y lo arreglamos cambiando el cero en " menos infinito" por su ganancia estática.

```
sys1e = zpk(d2c(funcion2))
```

```
Zero/pole/gain:
1.9272e-010 (s+1.038e010)
--------------------------
         s (s+2)
```

```
sys1e = sys1e.K * -sys1e.z{1}/((s+2)*s)
```

```
Zero/pole/gain:
   2
-------
s (s+2)
```

Como podemos observar, el resultado es exacto con el sistema creado para la simulación. Pero, que este resultado no nos haga pensar que siempre será así, hay que recordar que hemos hecho la estimación en ausencia de ruido de medición y de cualquier otro ruido, algo que en la realidad será poco probable.

Ahora estimaremos el sistema con otros órdenes para evaluar las diferentes aproximaciones, primero con orden 1 y luego con orden 3.

```
[be,d,funcion1]=stochastic2(y,escalon,T,1,1);
```

```
La matriz varianza-covarianza obtenida es:
  4.3246e-009 -2.5256e-008
 -2.5256e-008  3.9463e-007

Los coeficientes estimados +/- su incertidumbre:
     1.0005e+000   129.2821e-006
    41.7091e-003     1.2350e-003

La función de transferencia obtenida es:

Zero/pole/gain:
  0.041709
-------------
z^7 (z-1.001)

Sampling time: 0.05
```

El resultado para este modelo de orden 1 nos muestra que el estimador encontró de manera bastante aproximada el polo del integrador en z = -1; y que modela la dinámica del segundo polo como un retardo de 7 periodos de muestreo. Examinando la incertidumbre de los coeficientes del modelo estimado, vemos que se encuentra alredor o por debajo de $1x10^{-3}$.

Para realizar una comparación en simulación de ambos modelos debemos "arreglar" el modelo obtenido ajustando el polo a su valor -1.

```
funcion1.p{1}(8) = 1


Zero/pole/gain:
0.041709
---------
z^7 (z-1)

Sampling time: 0.05

[be,d,funcion3]=stochastic2(y,escalon,0.05,3,2);

La matriz varianza-covarianza obtenida es:
  2.3006e-008 -4.4035e-008  2.1029e-008 -5.4775e-013 -9.9817e-011
 -4.4035e-008  8.4296e-008 -4.0261e-008   1.101e-012  1.9065e-010
  2.1029e-008 -4.0261e-008  1.9232e-008 -5.5349e-013 -9.0833e-011
 -5.4775e-013   1.101e-012 -5.5349e-013  1.0454e-013 -1.0199e-013
 -9.9817e-011  1.9065e-010 -9.0833e-011 -1.0199e-013   5.549e-013

Los coeficientes estimados +/- su incertidumbre:
     1.9048e+000   298.1938e-006
  -904.8405e-003   570.7970e-006
     1.3076e-006   272.6406e-006
     2.4187e-003   635.6424e-009
     2.3394e-003     1.4645e-006

La función de transferencia obtenida es:

Zero/pole/gain:
0.0024187 (z+0.9672)
--------------------
 z (z-1) (z-0.9049)
```

```
Sampling time: 0.05
```

El resultado muestra únicamente una variación muy pequeña respecto a nuestro modelo "exacto" de segundo orden, al introducir un polo en z = 0, que equivale a un retardo de un tiempo de muestreo, para obtener el modelo de tercer orden.

```
close
step(funcion1,funcion2,funcion3,sys1,20)
legend show
```

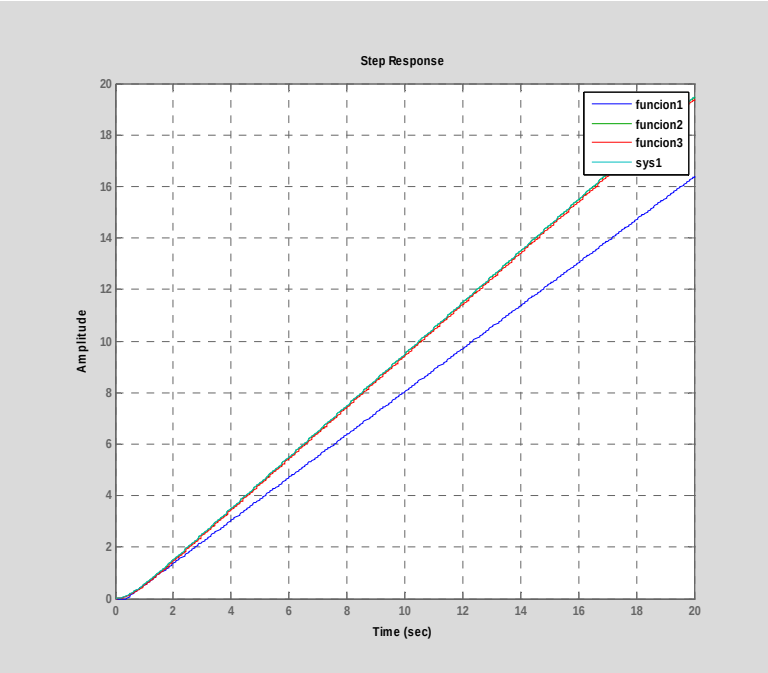

Figura 4: Respuesta ante escalón comparativa de los modelos obtenidos

Aparentemente del resultado de la simulación, el modelo de primer orden con tiempo muerto tiene una pequeña diferencia en la ganancia estimada. Los otros modelos coinciden casi exactamente con el sistema original.

Para comparar entre si la respuesta dinámica rápida de los sistemas estimados contra el original, los simulamos de 0 a 2 segundos.

```
close
step(funcion1,funcion2,funcion3,sys1,2)
legend show
```

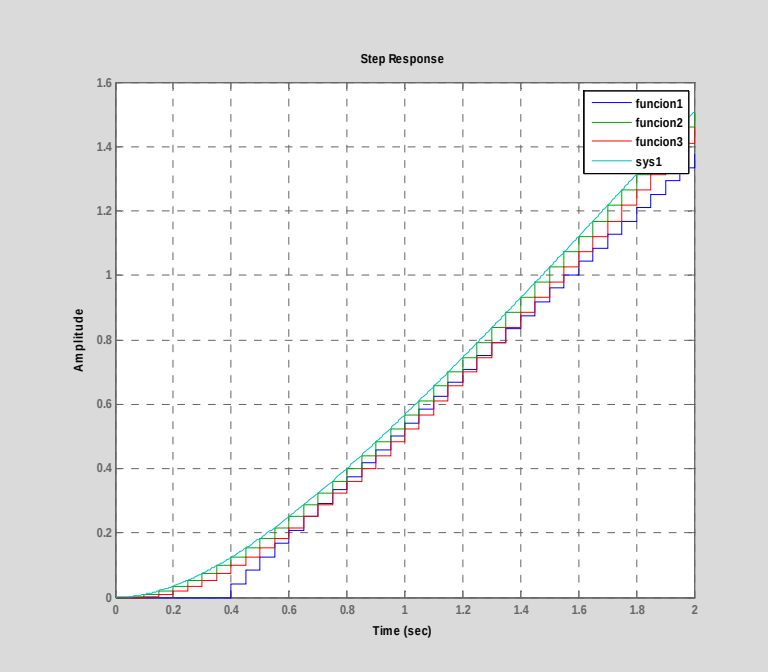

Figura 5: Comparación de la respuesta de los modelos estimados cerca del tiempo cero

Esta simulación muestra que de nuevo, el modelo de primer orden con tiempo muerto (en azul), difiere de los otros modelos en la respuesta inicial, debido al retardo de 7 periodos de muestreo. Por lo que este modelo, a pesar de su simplicidad parece ser menos adecuado que el modelo de segundo orden.

Respecto al modelo de tercer orden (en rojo), o de segundo orden con tiempo muerto; como era de esperarse, tiene solamente un retardo de un periodo de muestreo respecto al modelo de segundo orden "exacto", y por lo tanto la complejidad agregada no se compensa con una exactitud mejorada. Como conclusión, el modelo de segundo orden es el seleccionado.

Para terminar, en este caso, contábamos con un modelo real para comparar; algo que en la práctica no tendremos; por lo que las decisiones acerca de cuál modelo utilizar deben basarse en criterios como la incertidumbre de los coeficientes estimados, la complejidad esperada del modelo, orden uno, dos o tres, o la existencia o no de integradores; todo esto de acuerdo a nuestra experiencia y al conocimiento *a priori* que tengamos de la planta.

Parte 2: Reducción del modelo

En esta sección mostraremos el procesamiento de resultados obtenidos de forma experimental, no mostraremos la estimación del modelo, sino su reducción para poder así tener un modelo más simple; pero que conserve la dinámica original del sistema.

Declararemos el sistema estimado en tiempo discreto, y la variable z con el periodo de muestreo utilizado. Para evitar problemas con Matlab, definimos los polos conjugados complejos en forma de polinomio, obtenido del producto (z - p)(z - p*).

```
Ts = 0.1;
z = zpk('z',Ts);
sysz=0.08*((z+0.38)*(z-0.29))/((z-0.9)*(z^2 - 0.44*z + 0.3509))
```

```
Zero/pole/gain:
    0.08 (z+0.38) (z-0.29)
-------------------------------
(z-0.9) (z^2  - 0.44z + 0.3509)

Sampling time: 0.1
```

Como primer paso para la reducción, obtenemos un modelo balanceado del sistema.

```
[sysb,g]=balreal(sysz);
```

Examinando los valores singulares de ***Hankel***, que definen la contribución o "energía" de cada estado a la salida del sistema, definimos cuáles estados conservar y cuáles eliminar. Manteniendo, durante la reducción los estados de mayor energía de un sistema se preserva la mayoría de sus características en términos de estabilidad y respuestas de frecuencia y tiempo.

```
disp(g)
```

```
    0.4454
    0.0451
    0.0393
```

El primer estado del sistema balanceado tiene una energía aproximadamente 10 veces mayor que los estados 2 y 3; por lo que solamente conservaremos el estado 1. Utilizaremos el método por defecto de *modred* que garantiza el valor CD del modelo reducido.

```
sysr =zpk(modred(sysb,2:3))
```

```
Zero/pole/gain:
0.01577 (z+4.648)
-----------------
   (z-0.8965)

Sampling time: 0.1
```

En la función de transferencia obtenida observamos que existe un cero cuyo valor absoluto es relativamente grande el cual podemos simplificar, sustituyéndolo por su ganancia estática, para así simplificar aun más el modelo.

```
sysr=sysr.k*(1-sysr.z{1})/(z-sysr.p{1})
```

```
Zero/pole/gain:
 0.089072
----------
(z-0.8965)

Sampling time: 0.1
```

Si no deseamos trabajar en tiempo discreto, podemos convertir el modelo a tiempo continuo, recordando que esta transformación solamente devuelve resultados en la franja primaria del plano s debido a la periodicidad en $\omega_S = 2\pi/T$ que posee la transformación del plano s al plano z [4].

```
syss=d2c(sysr)
```

```
Zero/pole/gain:
 0.94027
---------
(s+1.093)
```

Finalmente una simulación nos permitirá validar nuestros modelos. Primero comparamos las respuestas del modelo en tiempo discreto original con el modelo reducido también en tiempo discreto. Observamos de los resultados en la figura 6 que ambos modelos son prácticamente idénticos.

```
close
step(sysz,sysr)
legend show
```

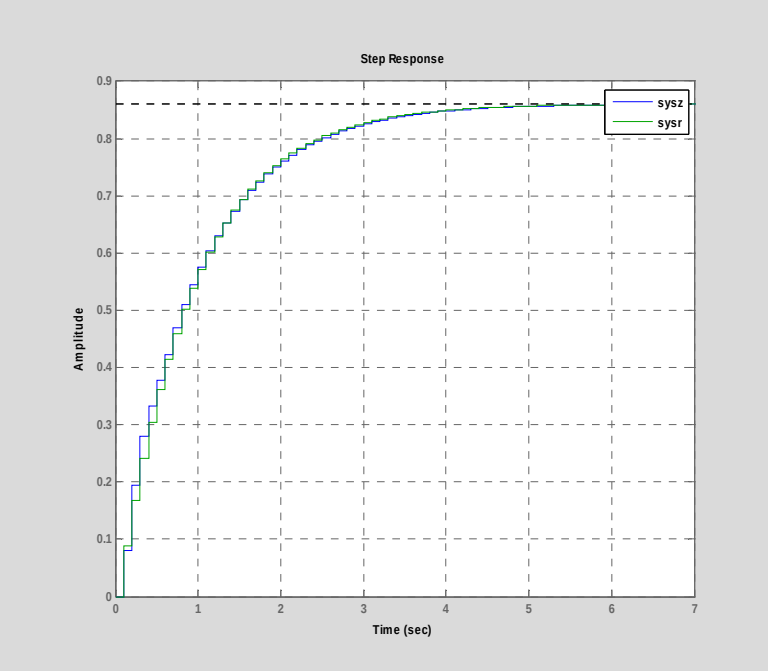

Figura 6: Comparación de la respuesta ante escalón del modelo original y el reducido

La comparación del la respuesta ante escalón del modelo original con la del modelo reducido en tiempo continuo nos confirma que también son prácticamente idénticas por lo que concluimos que cualquiera de ambas representaciones reducidas puede ser utilizada confiadamente.

```
close
step(sysz,syss)
legend show
```

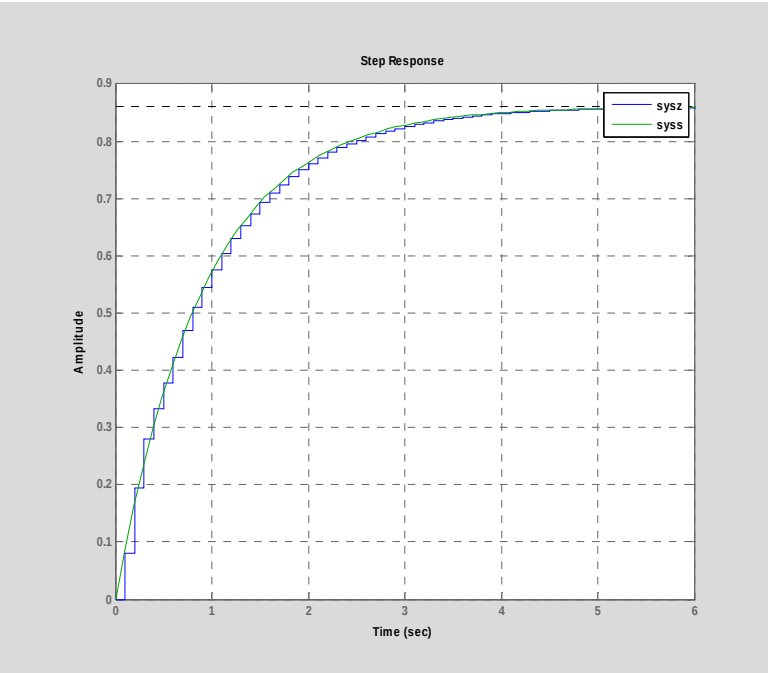

Figura 7: Comparación de la respuesta ante escalón del modelo original en tiempo discreto y el modelo reducido en tiempo continuo.

Referencias:

[1] http://www.ie.itcr.ac.cr/einteriano/control/trabajosmatlab/stochastic2.zip
[2] http://www.ie.itcr.ac.cr/einteriano/control/clase/3.6ModeladoEstocastico.pdf
[3] http://www.mathworks.com
[4] http://www.ie.itcr.ac.cr/einteriano/control/clase/3.4Clase22RlocusDiscreto.pdf

EIS/eis
2009