Métodos de Identificación dinámica

Autores: Dr. Pedro Arafet Padilla
Dr. Francisco Chang Mumañ
MSc. Miguel Torres Alberto
MSc. Hugo Dominguez Abreu

Facultad de Ingeniería Eléctrica
Universidad de Oriente
Junio 2008

Tabla de contenidos

Nota de los autores

Los métodos de identificación están muy dispersos en la literatura y se hace muy difícil para los profesionales la selección y posterior utilización de algunos de ellos. Por otro lado, en la docencia de pregrado no siempre se cuenta con programas adecuados para realizar alguna identificación práctica usando métodos sencillos.

Es la intención de los autores, con esta Monografía, ofrecerle a los profesionales, que laboran tanto en la industria como en la investigación y la docencia, una herramienta de trabajo que les permita tener acceso a los métodos de identificación experimental más comúnmente utilizados, brindando el fundamento teórico mínimo necesario, el algoritmo de cálculo para la comprensión de los mismos, así como, en algunos casos, el programa Matlab y ejemplos resueltos.

Se mencionan algunos de los métodos contenidos en el *toolbox* de identificación de Matlab, brindando sencillos programas para ello.

Se incluyen además las características específicas de cada método, lo que ayuda en gran medida a la selección más adecuada del mismo, dadas las condiciones particulares del proceso que se quiere identificar. Constituyen una importante ayuda las recomendaciones dadas para implementar en la práctica la identificación del sistema.

Se muestran, fundamentalmente, métodos de identificación de sistemas SISO (Single Input Single Output) y la identificación de un sistema MIMO (Multiple Input Multiple Output) de dos entradas y dos salidas con el Toolbox de Matlab. Así mismo se presenta la identificación de un sistema sencillo usando las Redes Neuronales

1. El problema de la identificación

Como se sabe, en muchas ocasiones, es muy útil poseer el modelo de un sistema para su análisis, y en particular, para el control, porque la inmensa mayoría de los métodos de diseño se basan en su conocimiento. A la determinación de dicho modelo, a partir de tener algún conocimiento previo sobre el proceso y de experiencias prácticas, se le conoce como identificación.

Teóricamente, para llegar a obtener un modelo podrían adoptarse dos enfoques diferentes:

- Por la vía analítica: determinar las ecuaciones y parámetros que intervienen siguiendo exclusivamente las leyes generales de la Física.
- Por la vía experimental: en la cual se considera el sistema como una "caja negra", con determinadas entradas y salidas, como se ilustra en la Figura 1.1. En esta situación se realizaría un conjunto de experimentos que proporcionarían pares de medidas de las entradas y salidas durante la evolución del sistema hacia el estado estacionario, a partir de los cuales se trataría de determinar el modelo del sistema.

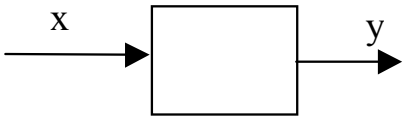


Fig. 1.1 Sistema como "caja negra".

Con respecto al primer enfoque hay que tener en cuenta que normalmente es extremadamente difícil considerar todas las leyes físicas que intervienen y que, aún suponiendo que esto fuera posible, el modelo resultante pudiera ser muy complejo, y por consiguiente, difícilmente manejable por las técnicas de diseño de sistemas de control. Por otra parte, en la práctica, las tolerancias de los elementos, desgastes, fuentes de ruido no consideradas, etc., hacen que el comportamiento real nunca sea el comportamiento previsto.

Por lo que respecta al segundo enfoque, es evidente que la resolución del problema de identificación sin adoptar hipótesis sobre las características del sistema puede ser muy

difícil.

En la práctica se combinan ambos enfoques, actuando en dos etapas:

- Etapa de análisis, en la cual se tienen en cuenta las leyes físicas y las condiciones particulares de trabajo para establecer hipótesis sobre la estructura y propiedades del modelo que se pretende identificar.

- Etapa experimental, en la cual se adoptan las hipótesis establecidas anteriormente y se tienen en cuenta las mediciones para determinar el modelo.

En el análisis hay que tener en cuenta que aunque el sistema sea no lineal, puede ser conveniente adoptar un modelo lineal con objeto de estudiar su comportamiento ante variaciones relativamente pequeñas sobre un punto de trabajo. Así mismo, pueden usarse hipótesis simplificadoras para describir el comportamiento del sistema mediante un modelo de orden reducido, más fácil de identificar y, posteriormente, de utilizar. Por otra parte, en sistemas lineales con múltiples entradas, es posible aplicar el principio de superposición, considerando cada salida como suma de salidas elementales correspondientes a una sola entrada. La situación se ilustra en la figura 1.2.

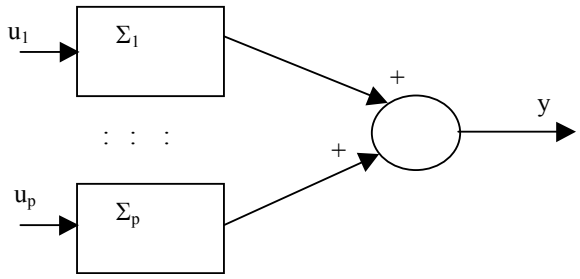


Fig. 1.2. Aplicación del principio de superposición

Un factor a tener en cuenta en el análisis es la determinación del tiempo de las experiencias, ya que pueden existir parámetros que varíen en función de perturbaciones lentas no medibles, o bien pueden aparecer no linealidades que no están presentes en un transitorio

alrededor de un punto de trabajo.

Otro aspecto importante, dentro del marco del control de los procesos industriales, es que no es lo mismo identificar un modelo de un sistema que trabajará a lazo abierto o a lazo cerrado. Está claro que en el primero se requiere mayor precisión.

El modelo de un sistema, como representación de sus aspectos fundamentales en la forma más conveniente para la finalidad a que está destinado, puede quedar expresado en forma de un conjunto de ecuaciones, tablas, gráficos o incluso de reglas que describen su operación.

Respecto al orden del modelo existen dos alternativas: imponer el orden o dejarlo libre a determinar.

Identificación experimental según las mediciones a procesar:

1.- Utilizando la respuesta ante señales de ensayo sobre el sistema.
2.- Procesando mediciones históricas de funcionamiento de la planta.
3.- Identificación en línea. Su aplicación es posible sin perturbar significativamente las condiciones de trabajo del sistema.
4.- Identificación en tiempo real.

Se puede establecer una clasificación según las características del modelo que se pretende obtener:

1.- De modelo paramétrico. Se pretenden obtener los valores de los coeficientes de las funciones o matrices de transferencia, o los elementos de las matrices de representación en el espacio de estado.
2.-De modelo no paramétrico. Sería el caso de las gráficas de módulo y fase en las respuestas frecuenciales, en los cuales se usarían los diagramas de Bode, Nyquist, Nichols y las respuestas a impulso o escalón.

En todo caso nótese que, a partir de un modelo paramétrico es muy fácil y rápido obtener respuestas en frecuencia y que un modelo no paramétrico puede parametrizarse empleando coeficientes tales como márgenes de fase y de ganancia, ancho de banda, etc.

Otra clasificación muy conocida es la de métodos frecuenciales y temporales, dependiendo del dominio (frecuencial o temporal) que se utilice.

Métodos frecuenciales

Se sabe que, si el sistema es lineal, la respuesta a una sinusoide es una sinusoide de la misma frecuencia y, en general, de diferente amplitud y fase a estado estacionario.

Excitando el sistema con sinusoides de diferentes frecuencias, especialmente en el rango de trabajo del sistema, podemos obtener las características de amplitud y fase y conocemos las contribuciones de los términos s, $(1 + \tau s)$, $(s^2 + as + b)$, en que pueden descomponerse el numerador y el denominador.

Otra clasificación posible, según el procesamiento de las mediciones, es la de métodos gráficos y analíticos.

En los métodos gráficos se obtienen los parámetros del sistema de manera gráfica, mientras que en los analíticos se obtienen producto de cálculos numéricos.

Es bueno destacar que con el desarrollo de la computación, muchos de estos métodos gráficos han pasado a ser analíticos, pero gráficos en su esencia.

Precisemos algunos conceptos:

¿Qué es la identificación experimental de sistemas?
La identificación de sistemas consiste en la determinación de un modelo que represente lo

más fielmente posible al sistema dinámico, a partir del conocimiento previo sobre éste y de los datos medidos.

¿Cómo se hace esto?
Esencialmente ajustando los parámetros obtenidos hasta que la salida del modelo coincida, lo mejor posible, con la salida medida.

¿Cómo saber si un modelo es bueno?
Una buena prueba es comparar la salida del modelo con datos medidos que no se usaron en la estimación de los parámetros.

¿Tenemos que asumir un modelo particular?
Para modelos paramétricos tenemos que asumir una estructura. Si asumimos que el sistema es lineal, podemos estimar directamente la respuesta a impulso o a escalón usando análisis de correlación o su respuesta frecuencial usando análisis espectral. Se usan mucho para comparar con otros modelos.

¿Es una gran limitación trabajar sólo con modelos lineales?
No, actualmente no. Generalmente se estiman los parámetros de las partes lineales y haciendo uso de la pericia física se le añaden al modelo las no linealidades.

Finalmente debemos entender que el modelo obtenido es un reflejo alejado de la realidad, pero que, sorprendentemente, es suficiente para tomar las decisiones necesarias.

2. Métodos gráficos

Estos métodos se caracterizan por determinar los parámetros del modelo de una forma gráfica, y por mucho tiempo se utilizaron de esta forma a pesar de las imprecisiones a que conllevan.
No obstante, con la ayuda de la computadora, muchos métodos gráficos se han programado mediante algoritmos analíticos.

2.1 Métodos basados en la respuesta a escalón.

El escalón es la señal de prueba más utilizada, en la práctica sólo puede lograrse de forma aproximada ya que es imposible lograr un cambio brusco de una variable en un tiempo infinitesimal, no obstante se considera válido si la constante de tiempo de la señal real es menor que la décima parte de la menor constante de tiempo que se quiere determinar en la identificación.

El uso de esta señal tiene la ventaja de la sencillez en su generación y que el tiempo de experimentación es corto. Como desventaja se puede mencionar la introducción de una alteración relativamente grande en el comportamiento del sistema, lo cual no siempre es permisible.

El procedimiento para obtener los parámetros del modelo estará en dependencia del modelo propuesto para la identificación, a partir de la respuesta del sistema a esta señal de estímulo.

2.1.1 Modelo de primer orden.

Para un sistema del tipo

$$G(s) = \frac{K}{\tau s + 1}$$

se necesitan estimar la ganancia (K) y la constante de tiempo(τ).

Para mayor generalidad, se excita al sistema con un escalón a la entrada de amplitud r_1-r, a partir de cualquier estado estacionario del sistema, obteniéndose una respuesta como se muestra en la figura 2.1.
La ganancia (K) se calcula como

$$\frac{c_1 - c}{r_1 - r} = \frac{\Delta c}{\Delta r}$$

y la constante de tiempo τ se calcula gráficamente como se muestra o tomando el valor de t para el cual $k = c + 0.63\Delta c$, o sea, que la respuesta c(t) ha alcanzado el 63.2% de su variación total.

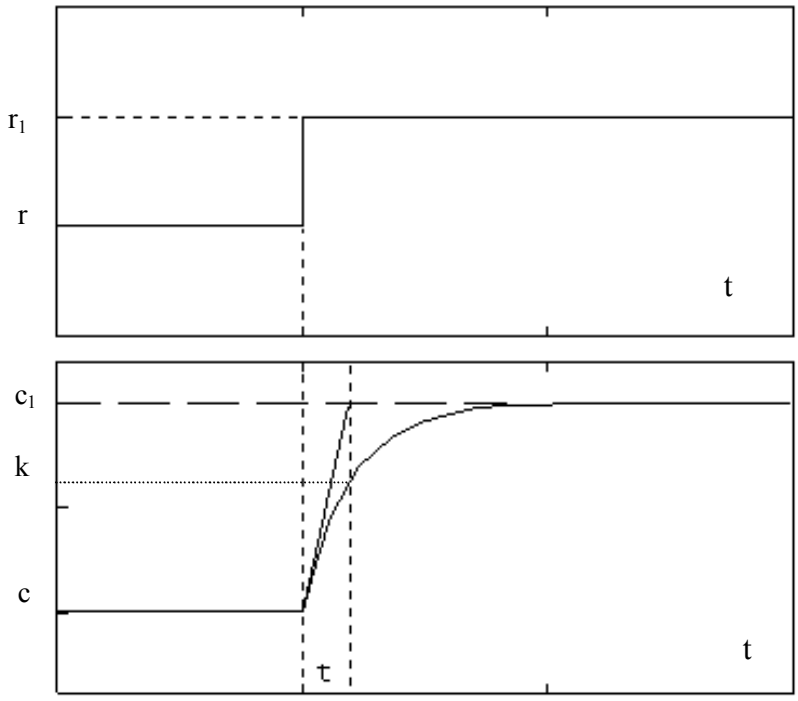


Fig. 2.1 Escalón de entrada y respuesta del sistema de primer orden.

2.1.2 Método de Oldenbourg – Sartorius

Se usa para sistemas de segundo orden no oscilatorio. La ganancia se calcula igual al caso de primer orden: $K = \Delta c / \Delta r$ (2.1)

Suponemos: $$F(s) = \frac{K}{(T_1 s + 1)(T_2 s + 1)} \quad (2.2)$$

Para calcular las constantes de tiempo T_1 y T_2 se usan las relaciones entre los tiempos T_A y T_C, definidos cuando se traza una tangente por el punto de inflexión de la curva que representa la respuesta de un sistema de segundo orden a un escalón, según muestra la Fig. 2.2

$$T_A = T_1 (\frac{T_2}{T_1})^n \qquad \text{donde} \quad n = \frac{T_2}{T_1 - T_2} \quad (2.3)$$

$$T_C = T_1 + T_2 \quad (2.4)$$

los valores de T_A y T_C se determinan gráficamente de la representación de la respuesta del sistema a un escalón.

La solución analítica de las ecuaciones resultantes al sustituir T_A y T_C en las expresiones (2.3) y (2.4) es muy compleja, por lo que resulta más conveniente aplicar un procedimiento gráfico.

Estas expresiones pueden escribirse como:

$$1 = \frac{T_1}{T_A} (\frac{T_2 / T_A}{T_1 / T_A})^n \quad (2.5)$$

$$\frac{T_C}{T_A} = \frac{T_1}{T_A} + \frac{T_2}{T_A} \quad (2.6)$$

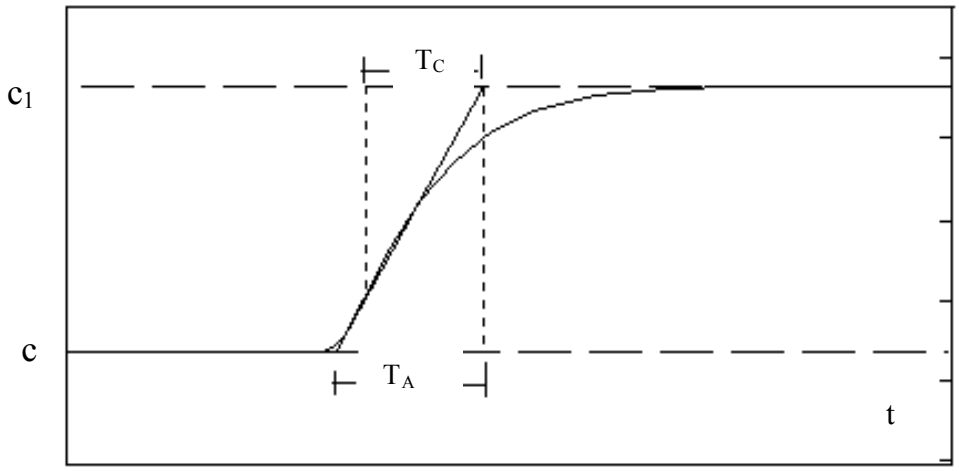


Fig. 2.2 Respuesta de un sistema de segundo orden a un escalón

De la primera ecuación se puede obtener la siguiente tabla:

T_1/T_A	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
T_2/T_A	1	0.73	0.57	0.44	0.34	0.25	0.18	0.12	0.07	0.03	0

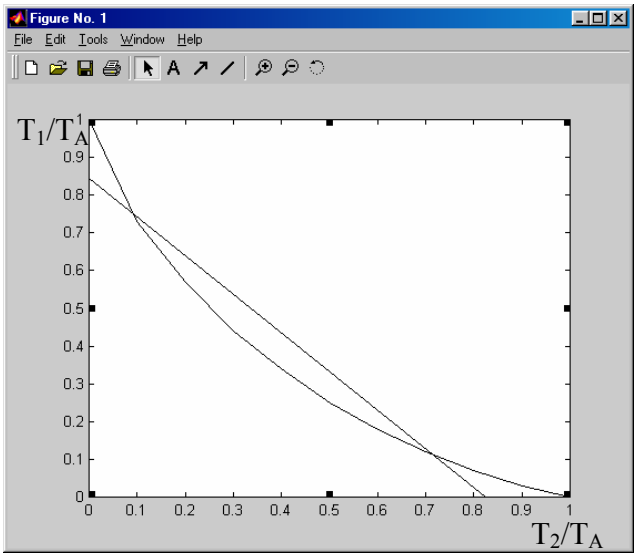


Fig. 2.3 Curvas para la aplicación del Método de Oldenbourg- Sartorius

Representamos ambas en una gráfica como se muestra en la fig. 2.3 .

Obteniéndose las constantes de tiempo en las intersecciones de ambas curvas.

Es importante señalar que si $T_C/T_A = 0.736$, la recta que representa la expresión es tangente a la curva, lo que significa que T_1 y T_2 son iguales. Si la relación $T_C/T_A < 0.736$ no hay intersección entre la recta y la curva y significa que estamos en presencia de un sistema de orden superior al segundo, caso para el cual este método no es aplicable.

Uno de los más graves problemas de estos métodos es el trazado de la tangente. Sin embargo esto se puede, hoy en día, simplificar usando Matlab.
En el Departamento de Informática de la Facultad de Ingeniería Eléctrica se elaboró un programa que resuelve el problema con el siguiente algoritmo:

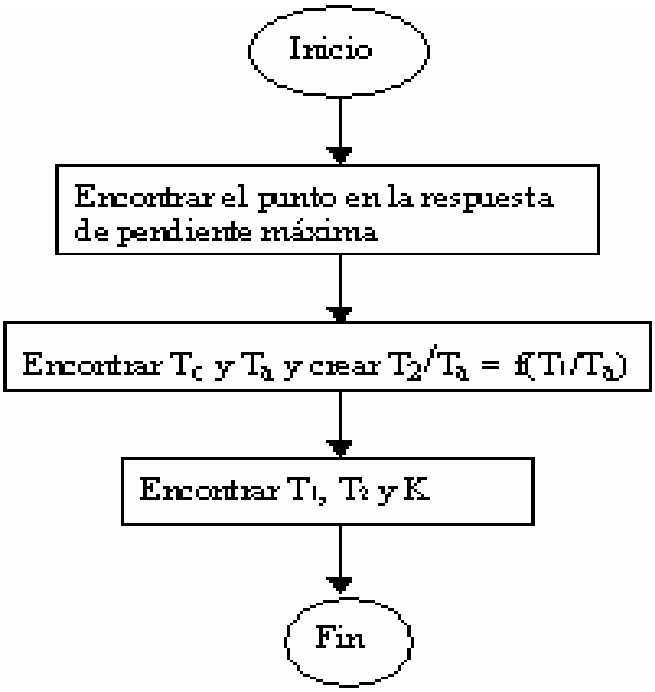


Fig. 2.4 Algoritmo de solución por Matlab

Se elaboró el programa os que se presenta en el anexo # 1.

2.1.3 Método de Anderson. Segundo orden

Si se tiene un sistema de segundo orden cuya función de transferencia es de la forma:

$$\frac{C(s)}{R(s)} = \frac{K}{(sT_1 + 1)(sT_2 + 1)}$$

la respuesta a un escalón de amplitud A sería:

$$C(t) = AK(1 + \frac{T_1 e^{-\frac{t}{T_1}} - T_2 e^{-\frac{t}{T_2}}}{T_2 - T_1}) \tag{2.7}$$

que puede escribirse:

$$C(t) = AK + K_1 e^{-\frac{t}{T_1}} - K_2 e^{-\frac{t}{T_2}} \tag{2.8}$$

donde K_1 y K_2 son positivos si $T_2 > T_1$. La diferencia entre las respuestas a estado estacionario y transitoria sería:

$$C_L(t) = AK - C(t) = -K_1 e^{-\frac{t}{T_1}} + K_2 e^{-\frac{t}{T_2}} \tag{2.9}$$

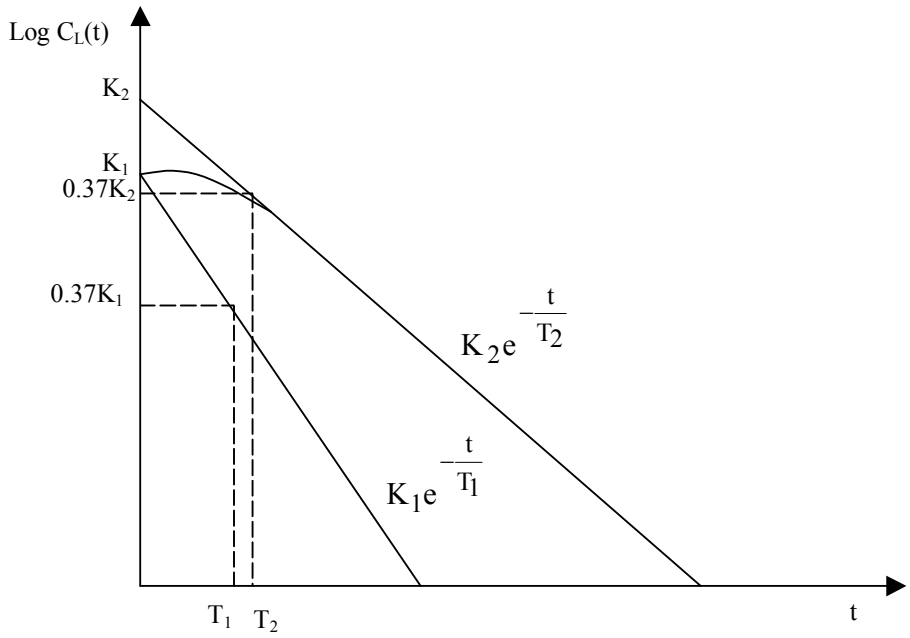


Fig. 2.5 Representación gráfica del Método de Anderson

y como T_1 y T_2 son reales, C(t) es no oscilatoria y $C_L(t)$ es positiva.

Como $T_2 > T_1$, el primer término de $C_L(t)$ disminuye más rápidamente que el segundo, luego, para valores altos de t, se puede decir que:

$C_L(t) \approx K_2 e^{-\frac{t}{T_2}}$ y tomando logaritmo se tiene que:

$$\log C_L(t) \approx \log K_2 - (\frac{1}{T_2} \log e)t \qquad (2.10)$$

es decir, si se construye una gráfica de $C_L(t)$, usando papel semilogarítmico con el eje logarítmico para $C_L(t)$ y el eje lineal para t, se obtiene, para valores altos de t, una línea recta que interseca al eje logarítmico (t = 0) en K_2.

También se sabe que un término exponencial de la forma $K_2 e^{-\frac{t}{T_2}}$ alcanza un 36.8% de su valor inicial cuando $t = T_2$. De acuerdo con esto se puede establecer la gráfica anterior y determinar la constante de tiempo mayor.

Para la determinación de la otra constante de tiempo, obsérvese que:

$$K_2 e^{-\frac{t}{T_2}} - C_L(t) = K_1 e^{-\frac{t}{T_1}} \qquad (2.11)$$

Esto significa, por un razonamiento similar, que si se traza la representación de la diferencia mencionada en un gráfico semilogarítmico, se obtiene una línea recta que corresponde a la representación de $K_1 e^{-\frac{t}{T_1}}$, dicha línea recta, para t = 0 tiene por ordenada k_1 y para $t = T_1$ tiene por ordenada 0,368 k_1 .

En resumen, se deben de seguir los siguientes pasos:

- Se traza la curva correspondiente a la diferencia del valor a estado estacionario y el transitorio de la respuesta del sistema a un escalón en función del tiempo, en un papel semilogarítmico, situando en el eje lineal los valores del tiempo.

- Se prolonga la parte recta de la curva anterior, correspondiente a los valores altos de t, hasta intersecar el eje vertical (K_2).

- Se determina la mayor constante de tiempo como el valor del tiempo donde la recta anterior alcanza el 0.368 de su valor inicial.

- Se repite el procedimiento anterior para los valores resultantes de la diferencia entre la curva original $C_L(t)$ y la recta que corresponde a valores altos de t, con lo que se determina la menor constante de tiempo.

Aunque teóricamente este método se puede utilizar para sistemas de orden superior, dado su carácter gráfico, en la práctica no se pueden determinar más de dos constantes de tiempo.

2.1.4 Respuesta a escalón para sistemas oscilatorios.

Es conocido que la función de transferencia de un sistema oscilatorio de segundo orden tiene la forma:

$$G(s) = \frac{K\omega_n^2}{s^2 + 2\delta\omega_n s + \omega_n^2} \tag{2.12}$$

estos sistemas se pueden identificar determinando gráficamente los valores del pico de sobrepaso M_p y del período de las oscilaciones amortiguadas d, de su respuesta ante una señal escalón en su entrada, Fig. 2.6.

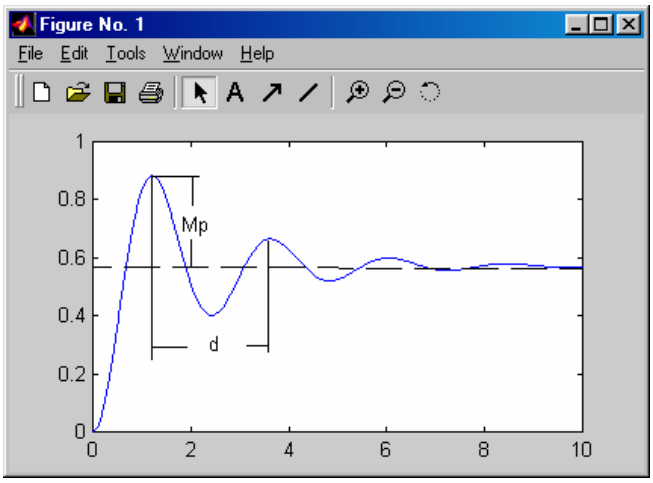


Fig. 2.6 Respuesta a un escalón de un sistema oscilatorio de segundo orden

El problema consiste en estimar K, δ y ω. El valor de la ganancia K se determina según (2.1), a partir de M_p puede calcularse el coeficiente de amortiguamiento δ mediante la expresión:

$$M_p = e^{-\frac{\delta\pi}{\sqrt{1-\delta^2}}} \tag{2.13}$$

Para obtener la frecuencia natural no amortiguada ω_n se emplean las expresiones:

$$\omega_n = \frac{\omega}{\sqrt{1-\delta^2}} \qquad \omega = \frac{2\pi}{d} \tag{2.14}$$

siendo M_p el pico de sobrepaso y d el período de oscilación amortiguada.

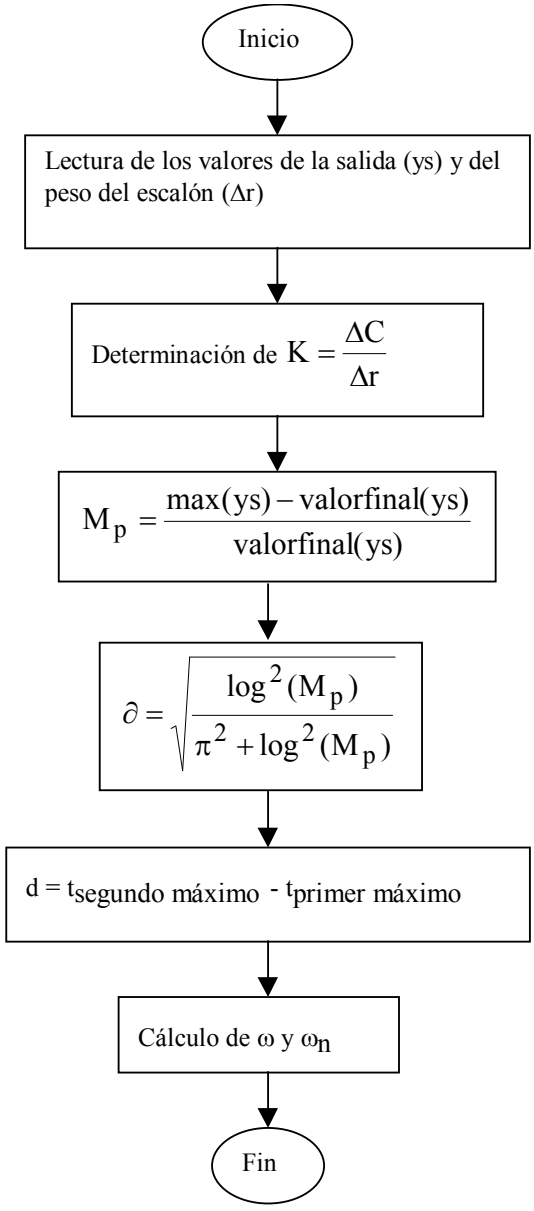


Fig. 2.7 Algoritmo de solución para este método

Para este método se confeccionó un programa Matlab y se probó con un ejemplo en Simulink.

Los parámetros de la simulación fueron los siguientes:
Un escalón unitario a la entrada
Un paso fijo de 0.15 unidades de tiempo
Tiempo de simulación 25 unidades de tiempo
Método de solución: ode5 (Dormand-Prince)

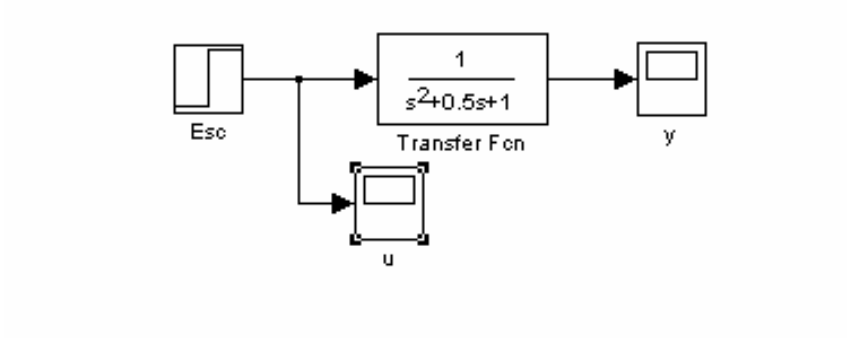


Fig. 2.8 Esquema de simulación

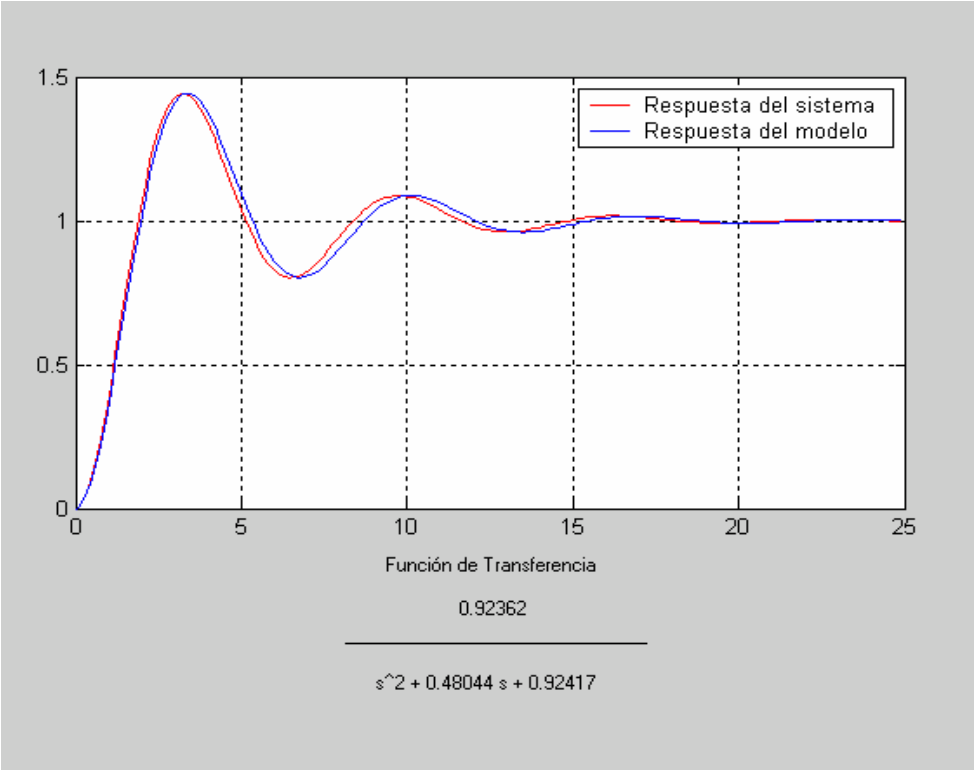


Fig. 2.9 Resultado de la identificación

La precisión pudo haber sido mejor si se hubiera disminuido el tiempo de muestreo, que en nuestro caso fue de 0.15 unidades de tiempo.

El listado del programa se encuentra en el anexo # 2

2.1.5 Método de Strejc. Modelo de orden n.

Se usa para modelos del tipo:

$G(s) = \dfrac{K}{(Ts+1)^n}$, es decir, se deben estimar K, T y n a partir de la respuesta a un escalón.

La ganancia K se determina como hasta ahora, en base a $\dfrac{\Delta c}{\Delta r}$. Se traza la curva que representa la respuesta del sistema y se traza la tangente por el punto de inflexión, determinando los tiempos T_L y T_A, Fig. 2.10.

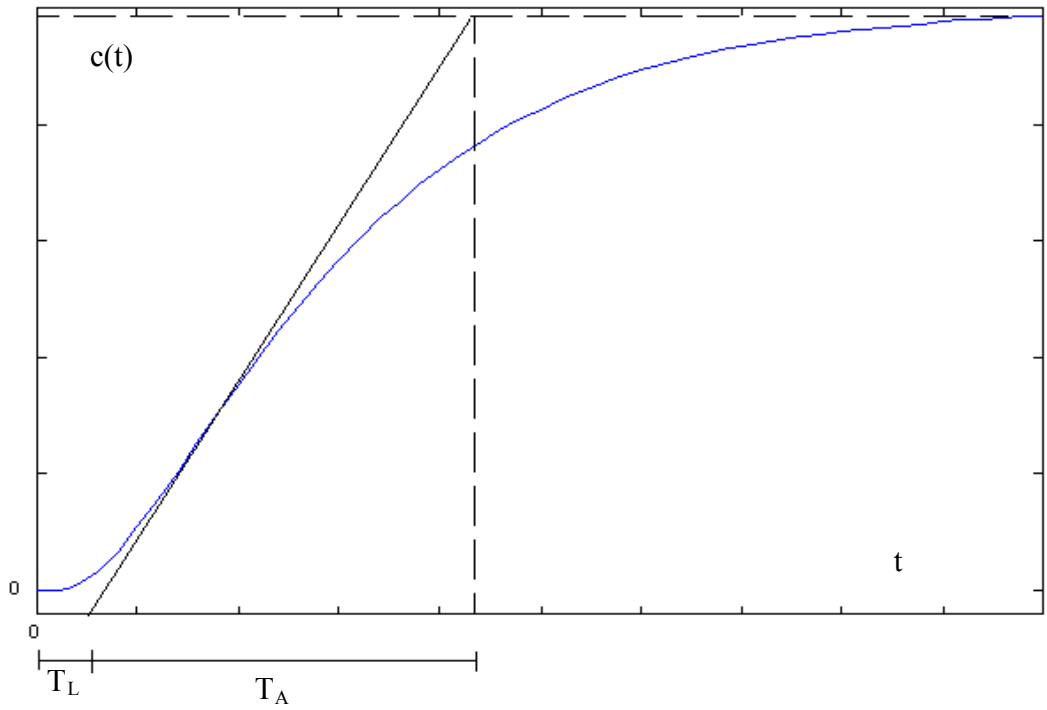


Fig.2.10 Determinación de T_A y T_L para la aplicación del Método de Strejc

La relación entre T_L y T_A es una función creciente de n, y T_A y T_L son proporcionales a T, los factores de proporcionalidad dependen de n, como se observa en la Tabla 2.1.

n	T_L/T_A	T_A/T	T_L/T
2	0.104	2.718	0.282
3	0.218	3.695	0.805
4	0.319	4.463	1.425
5	0.410	5.119	2.106
6	0.493	5.7	2.811

Tabla 2.1 Relaciones entre T_A, T_L y T en función de n.

El procedimiento sería:

- Determinar T_L y T_A a partir del gráfico de la respuesta del sistema al escalón.
- Con la relación T_L/T_A se determina el valor de n por medio de la Tabla 2.1.
- Con el valor T_A/T de la Tabla 2.1, correspondiente al valor de n determinado anteriormente y el de T_A, se calcula T. De forma similar se puede usar la relación T_L/T y el tiempo T_L.

Se debe tener en cuenta que cuando T_L/T_A está entre dos valores de n, se toma el menor.

2.1.6 Modelos con retraso de transporte

Es característico en control de procesos la presencia de retrasos puros o retrasos de transporte (L), por lo que es importante la consideración de éstos en los modelos propuestos:

$$\frac{Ke^{-sL}}{Ts+1} \qquad \frac{Ke^{-sL}}{(T_1s+1)(T_2s+1)} \qquad \frac{Ke^{-sL}}{(Ts+1)^n} \tag{2.15}$$

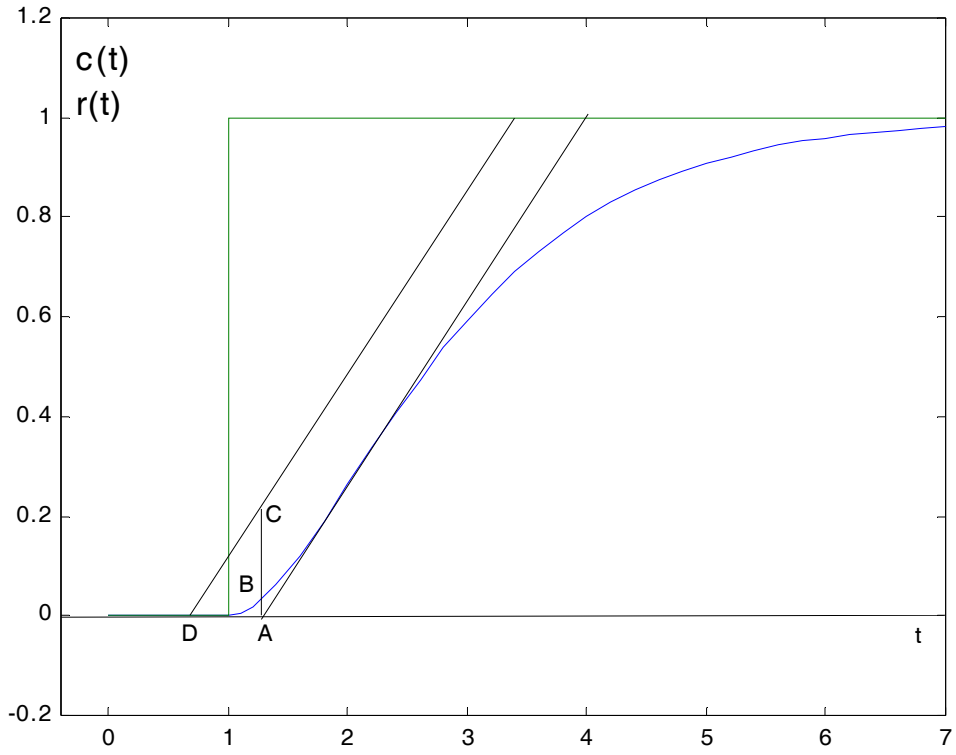


Fig. 2.11 Determinación del tiempo de retardo L.

El proceso de identificación consta de dos partes:

- Determinación del retraso de transporte L.
- Determinación de los restantes parámetros por los métodos descritos.

Para determinar L es posible:

a) Tomar el tiempo para el cual se obtiene 0.05Δc, es decir, el 5% de Δc.
b) Se traza la tangente en el punto de inflexión y se levanta una perpendicular en A hasta C, de manera que AC = 2.718 AB. Por C se traza una paralela a la tangente hasta encontrar D. Desde 0 hasta D es el retraso de transporte.

2.1.7 Método del retraso puro efectivo o tiempo muerto efectivo

Es frecuente en la identificación de procesos químicos la existencia de sistemas cuya respuesta a un estímulo escalón tiene generalmente la forma de S. En estos casos un método bastante efectivo y simple consiste en descomponer dicha respuesta en un retraso puro y un sistema de primer orden.

De manera que se obtiene:

$$G(s) = \frac{k_p e^{-\tau s}}{Ts + 1} \qquad (2.16)$$

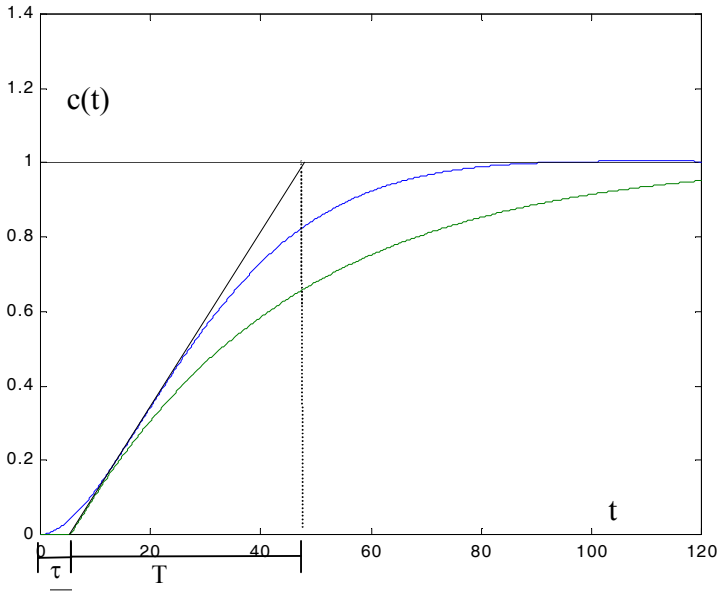


Fig. 2.12 Obtención de la constante de tiempo T a partir de la respuesta a escalón.

Como se sabe, la ganancia del sistema se obtiene con $k_p = \frac{y_s(\infty) - y_s(0)}{\Delta x_e}$

Donde:

y_s – salida del sistema

Δx_e – amplitud del escalón en la entrada del sistema

La constante de tiempo T se obtiene gráficamente de la tangente al punto de inflexión como se muestra en la Fig. 2.12.

2.2 Métodos basados en la respuesta a un pulso

La utilización de un pulso como señal de prueba en la identificación de sistemas, permite lograr tiempos de experimentación cortos sin introducir grandes perturbaciones en el comportamiento del sistema objeto de estudio, a expensas de mayores exigencias en la exactitud de las mediciones a realizar.

2.2.1 Método de pulso para obtener la respuesta a escalón

Si se aplica a un sistema un pulso rectangular de duración T_p y se registra la respuesta a dicho pulso, desplazando éste muchas veces un tiempo igual a T_p y sumando las respuestas resultantes, se obtiene la respuesta del sistema a un escalón; lo cual es posible ya que si se considera el sistema lineal se cumple el principio de superposición.

Obsérvese que se obtiene la respuesta del sistema a un escalón sin necesidad de aplicar esta señal a su entrada, por lo que el sistema a identificar sólo sufre la perturbación correspondiente al pulso rectangular.

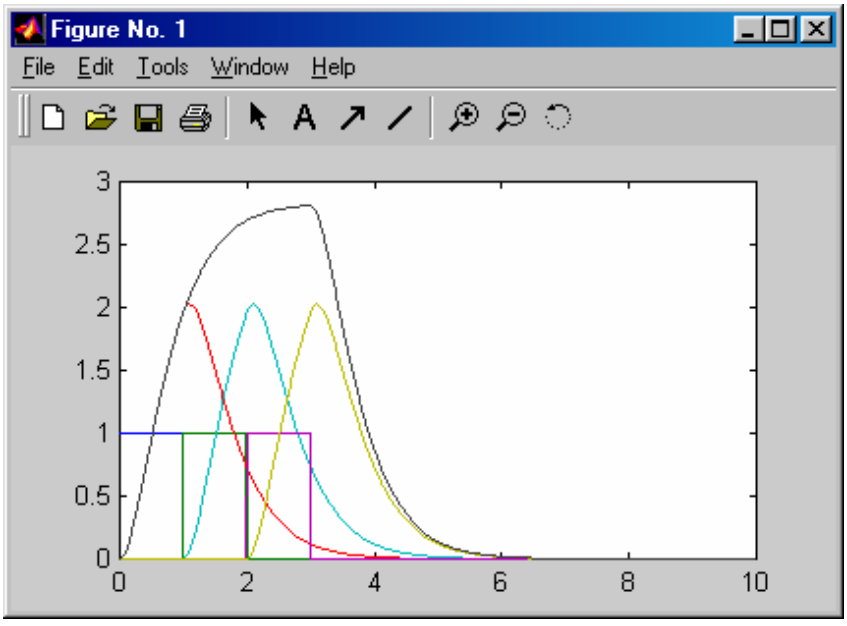


Fig. 2.13 Gráfica de la respuesta a un escalón a partir de la respuesta a un pulso.

Aunque teóricamente un escalón es igual a la suma de un número infinito de pulsos rectangulares, en la práctica es suficiente un número finito de éstos hasta que la suma de las respuestas desplazadas muestre un valor constante, el cual corresponde al valor a estado estacionario producto del escalón equivalente. Si n es el número de respuestas que cumple con lo anterior, tiene que ocurrir que n * T_p sea mayor que el tiempo para el cual el sistema alcanza prácticamente el estado estacionario, o sea, n * T_p > 5 T; donde T es la mayor constante de tiempo del sistema analizado.

Este método es de los llamados no paramétricos, pues lo que se obtiene es la respuesta a escalón en lugar de los parámetros del modelo. En el Departamento de Informática de la Facultad de Ing. Eléctrica de la Universidad de Oriente existe un programa para este método.

Se confeccionó un programa Matlab (anexo # 3) y se probó con un ejemplo en Simulink.

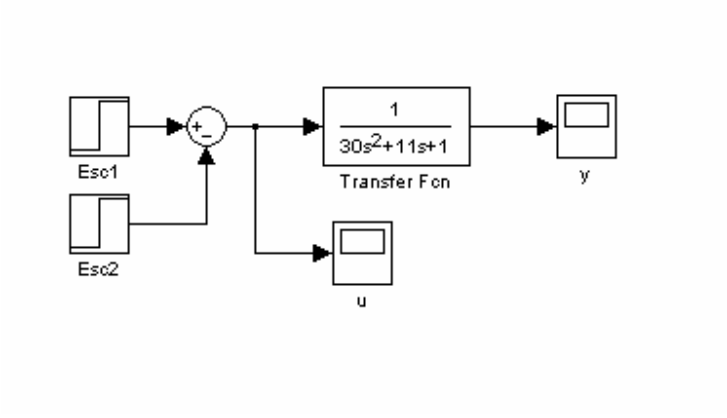


Fig. 2.14 Esquema confeccionado en Simulink

Los parámetros de simulación fueron los siguientes:
Un pulso unitario de duración 2 unidades de tiempo.
Tiempo de simulación: 40 unidades de tiempo.
Paso fijo de 0.01 unidades de tiempo.
Método de solución ode5 (Dormand-Prince)

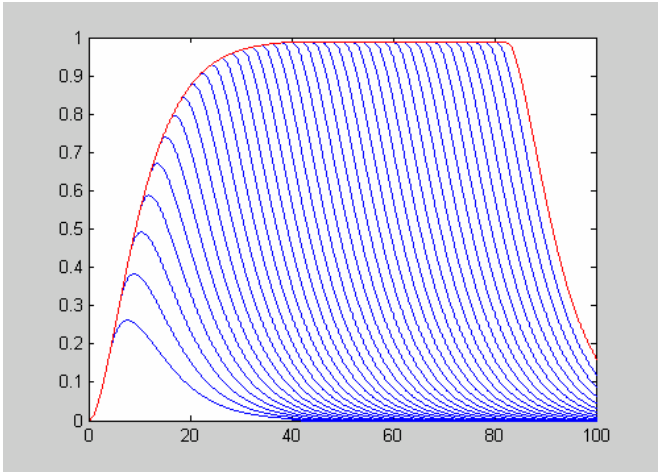


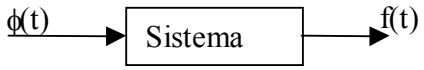
Fig. 2.15 Resultados de la identificación

3. Métodos analíticos

Los métodos analíticos son los más comunes y existe una gran variedad en la literatura. A continuación se presenta una muestra de ellos.

3.1 Método de pulso para obtener la respuesta frecuencial

Se trata de un método no paramétrico en el cual se descomponen las mediciones de la entrada y la salida del sistema en pulsos rectangulares de igual ancho y, de esa forma, calcular la función transferencial sinusoidal. Como se sabe de ésta se pueden determinar las relaciones de amplitud y fase, o sea, cuando se sustituye s = jω en la función de transferencia. Sea el siguiente sistema:



Si f(t) y Φ(t) son transformables por Fourier:

$$\text{Entonces } F(j\omega) = \frac{\int_{-\infty}^{+\infty} f(t)e^{-j\omega t}dt}{\int_{-\infty}^{+\infty} \Phi(t)e^{-j\omega t}dt} \qquad (3.1)$$

Dado el hecho de que es muy difícil calcular la integral de la entrada y la salida, usaremos un método gráfico para evaluar las mismas, descomponiendo los transcursos de la entrada y la salida en pulsos rectangulares de igual ancho. Como suponemos que $t \geq 0$, se pueden expresar los límites de las integrales de 0 a ∞.

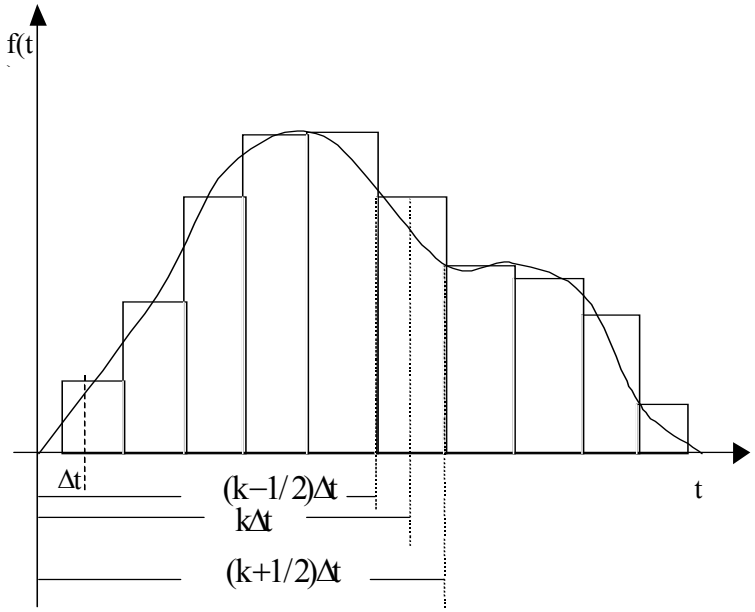


Fig. 3.1. Descomposición en pulsos ancho Δt de la respuesta del sistema

De manera que:

$$\int_0^{\infty} f(t)e^{-j\omega t}dt = \sum_{k=0}^{\infty} \int_{k-\frac{1}{2}\Delta t}^{k+\frac{1}{2}\Delta t} f(t)e^{-j\omega t}dt \tag{3.2}$$

Según la gráfica anterior f(kΔt) es el valor de la función en el k-ésimo intervalo, por lo que la ecuación anterior puede escribirse como:

$$\int_0^{\infty} f(t)e^{-j\omega t}dt = \sum_{k=0}^{\infty} f(k\Delta t) \int_{k-\frac{1}{2}\Delta t}^{k+\frac{1}{2}\Delta t} e^{-j\omega t}dt \tag{3.3}$$

y la integral se puede calcular según:

$$\int_{k-\frac{1}{2}\Delta t}^{k+\frac{1}{2}\Delta t} e^{-j\omega t}dt = -\frac{1}{j\omega} \int_{k-\frac{1}{2}\Delta t}^{k+\frac{1}{2}\Delta t} -j\omega e^{-j\omega t}dt = \frac{e^{-j\omega(k-\frac{1}{2})\Delta t} - e^{-j\omega(k+\frac{1}{2})\Delta t}}{j\omega} \tag{3.4}$$

por tanto:

$$\int_0^{\infty} f(t)e^{-j\omega t}dt = \sum_{k=0}^{\infty} f(k\Delta t)\frac{e^{-j\omega(k-\frac{1}{2})\Delta t} - e^{-j\omega(k+\frac{1}{2})\Delta t}}{j\omega} \tag{3.5}$$

finalmente quedaría:

$$\int_0^{\infty} f(t)e^{-j\omega t}dt = \frac{e^{j\omega\frac{\Delta t}{2}} - e^{-j\omega\frac{\Delta t}{2}}}{j\omega}\sum_{k=0}^{\infty} f(k\Delta t)e^{-j\omega(k\Delta t)} \tag{3.6}$$

Procesando la señal estímulo (cualquiera que sea su forma), siguiendo el procedimiento anterior para igual valor de Δt y suponiendo n intervalos, se tiene:

$$F(j\omega) = \frac{\sum_{k=1}^{n} f(k\Delta t)e^{-j\omega(k\Delta t)}}{\sum_{k=1}^{n} \Phi(k\Delta t)e^{-j\omega(k\Delta t)}} \tag{3.7}$$

y sabiendo que $e^{-j\theta} = \cos(\theta) - j\,\mathrm{sen}(\theta)$ (3.8)

se puede escribir la ecuación anterior según:

$$F(j\omega) = \frac{\sum_{k=1}^{n} f(k\Delta t)\cos(k\omega\Delta t) - j\sum_{k=1}^{n} f(k\Delta t)\,\mathrm{sen}(k\omega\Delta t)}{\sum_{k=1}^{n} \Phi(k\Delta t)\cos(k\omega\Delta t) - j\sum_{k=1}^{n} \Phi(k\Delta t)\,\mathrm{sen}(k\omega\Delta t)} \tag{3.9}$$

simplificando esta ecuación:

$$F(j\omega) = \frac{A(\omega) + jB(\omega)}{C(\omega) + jD(\omega)} \tag{3.10}$$

que, para el análisis frecuencial será:

$$RA = \frac{\sqrt{A^2 + B^2}}{\sqrt{C^2 + D^2}} \qquad \text{y} \qquad \varphi = \tan^{-1}\frac{B}{A} - \tan^{-1}\frac{D}{C} \tag{3.11}$$

El término $\omega\Delta t$ no debe exceder de $\frac{\pi}{2} = 1.57$ radianes para una aproximación satisfactoria de la transformada de Fourier. Si esto sucede debemos disminuir Δt.

El algoritmo de solución es evidente.

3.2 Método de correlación. Forma continua

Estos métodos, no paramétricos, se basan en la aplicación de señales aleatorias a la entrada y tienen una gran importancia en la actualidad.

Se deben conocer dos conceptos importantes:

- La función de correlación cruzada entre dos variables aleatorias r(t) y c(t) se designa por:

$$R_{rc}(t_2) = \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} r(t)c(t+t_2)dt \qquad (3.12)$$

esta función da una medida de la dependencia de la variable c en el instante $t+t_2$ y el valor de la variable r en t.

- La función de autocorrelación de una variable aleatoria se designa por:

$$R_{rr}(t_2) = \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} r(t)r(t+t_2)dt \qquad (3.13)$$

esta función da una medida de la dependencia del valor de una variable en $t+t_2$ con respecto al valor de la misma variable en t.

Se sabe que si r(t) y c(t) son las variables de entrada y salida de un sistema, cuya respuesta a un impulso o función pesante es g(t), se puede plantear que:

$$c(t) = \int_{-\infty}^{\infty} r(t-t_1)g(t_1)dt_1 \qquad (3.14)$$

de manera que $c(t+t_2)$ está dada por:

$$c(t+t_2)=\int_{-\infty}^{\infty} r(t+t_2-t_1)g(t_1)dt_1 \tag{3.15}$$

Si se sustituye $c(t+t_2)$ en la correspondiente a la función de correlación cruzada, se tiene que:

$$R_{rc}(t_2)=\lim_{T\to\infty}\frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} r(t)dt\int_{-\infty}^{\infty} r(t+t_2-t_1)g(t_1)dt_1 \tag{3.16}$$

y al intercambiar el orden de integración se obtiene:

$$R_{rc}(t_2)=\int_{-\infty}^{\infty} g(t_1)dt_1\left[\lim_{T\to\infty}\frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} r(t)r(t+t_2-t_1)dt\right] \tag{3.17}$$

Se puede observar que la función entre corchetes corresponde a la función de autocorrelación de r(t) en $t = t_2 - t_1$, o sea, que la función de correlación cruzada puede escribirse:

$$R_{rc}=(t_2)=\int_{-\infty}^{\infty} R_{rr}(t_2-t_1)g(t_1)dt_1 \tag{3.18}$$

Si se tiene que la señal de entrada aplicada a un sistema tiene función de autocorrelación proporcional a una función impulso, entonces, $R_{rr}(t_2)=k\partial(t_2)$ y sustituyendo en la expresión anterior, se tiene que:

$$R_{rc}=kg(t_2)\int_{-\infty}^{\infty}\partial(t_2-t_1)g(t_1)dt_1 \tag{3.19}$$

En esta expresión el integrando es diferente de cero solo cuando $t_2 = t_1$, luego puede escribirse:

$$R_{rc}(t_2)=kg(t_2)\int_{-\infty}^{\infty}\partial(t_2-t_1)dt_1 \tag{3.20}$$

y finalmente, dado que : $\int_{-\infty}^{\infty} \partial(t_2 - t_1)dt_1 = 1,$ se llega a:

$$R_{rc} = kg(t_2) \tag{3.21}$$

Esto significa que si a un sistema se le aplica como estímulo una señal aleatoria con función de autocorrelación dada por $k\partial(t)$, el resultado de calcular la función de correlación cruzada de la variable de salida y el estímulo correspondiente es proporcional a la respuesta a impulso. Como esta función caracteriza al sistema se puede decir que hemos identificado el sistema.

En sistemas industriales es posible, a veces, usar las propias perturbaciones aleatorias que sufren las variables de entrada o generarlas.

3.3 Método de correlación cruzada. Forma discreta

Considérese un sistema con secuencia de ponderación g(j) sometido a ruidos aditivos a la salida. En este caso puede escribirse:

$$y(i) = \sum_{j=0}^{\infty} g(j)u(i-j) + \nu(i) \tag{3.22}$$

La correlación cruzada entre la salida y la entrada se define mediante:

$$\Phi_{yu}(k) = \lim_{n \to \infty} \frac{1}{2n+1} \sum_{i=-n}^{n} y(i)u(i-k) \tag{3.23}$$

Sustituyendo (3.22) en (3.23) se obtiene:

$$\Phi_{yu}(k) = \lim_{n \to \infty} \frac{1}{2n+1} \sum_{i=-n}^{n} u(i-k) \left[\sum_{j=0}^{\infty} g(j)u(i-j) + \nu(i) \right] \tag{3.24}$$

El término delante del corchete se puede pasar para adentro de la sumatoria en j:

$$\Phi_{yu}(k) = \sum_{j=0}^{\infty} g(j) \lim_{n\to\infty} \frac{1}{2n+1} \sum_{i=-n}^{n} u(i-k)u(i-j) + \lim_{n\to\infty} \frac{1}{2n+1} \sum_{i=-n}^{n} u(i-k)\nu(i) \tag{3.25}$$

Haciendo el cambio de variables t = i – k, se obtiene:

$$\Phi_{yu}(k) = \sum_{j=0}^{\infty} g(j) \lim_{n\to\infty} \frac{1}{2n+1} \sum_{i=-n}^{n} u(t)u(t-j+k) + \lim_{n\to\infty} \frac{1}{2n+1} \sum_{i=-n}^{n} u(i-k)\nu(i) \tag{3.26}$$

Y como la autocorrelación de la entrada se define como:

$$\Phi_{uu}(k) = \lim_{n \to \infty} \frac{1}{2n+1} \sum_{i=-n}^{n} u(i)u(i-k) \tag{3.27}$$

sustituyendo ésta en la anterior se obtiene:

$$\Phi_{yu}(k) = \sum_{j=0}^{\infty} g(j)\Phi_{uu}(j-k) + \Phi_{\nu u}(k) \tag{3.28}$$

Si la entrada y el ruido no están correlacionados, entonces

$$\Phi_{yu}(k) = \sum_{j=0}^{\infty} g(j)\Phi_{uu}(j-k) \tag{3.29}$$

Si la entrada es un ruido blanco, entonces para toda $k \neq j$ se obtiene:

$$\Phi_{uu}(k) = 0 \quad y \quad \Phi_{uu}(0) = d \tag{3.30}$$

de manera que : $$\Phi_{yu}(k) = d\, g(k) \tag{3.31}$$

Siendo d una constante, concretamente el peso del impulso de la autocorrelación de la entrada. Por lo que se demuestra que se puede encontrar la respuesta a impulso sin más que determinar la correlación cruzada entre la entrada y la salida.

Este método se puede aplicar en línea. En efecto, considérese un sistema con entrada r(k) a la cual se le añade un ruido blanco u(k). En este caso la entrada real que se aplica al sistema es:

$\tilde{u}(k) = r(k) + u(k)$ de manera que $y(k) = y_r(k) + y_u(k)$, sustituyendo esto en (3.23) se obtiene:

$$\Phi_{yu}(k) = \lim_{n \to \infty} \frac{1}{2n+1} \sum_{i=-n}^{n} y_r(i)u(i-k) + \lim_{n \to \infty} \frac{1}{2n+1} \sum_{i=-n}^{n} y_u(i)u(i-k) \tag{3.32}$$

Dado el hecho que r(k) y u(k) no están correlacionadas, tampoco lo estarán $y_r(k)$ y u(k), por lo que se puede despreciar el primer término.

Si se consideran los valores a partir de i = 0, se obtiene:

$$\Phi_{yu}(k) = \frac{1}{n+1} \sum_{i=0}^{n} y(i)u(i-k) \tag{3.32}$$

Se confeccionó un programa Matlab (anexo # 4), que contiene la función CRA que resuelve el problema, y se probó con un ejemplo en Simulink.

Los parámetros de simulación fueron los siguientes:

Muestreo de 0.1 unidades de tiempo

Método de solución ode5

Tiempo final 100 unidades de tiempo

Un escalón a la entrada de peso 1

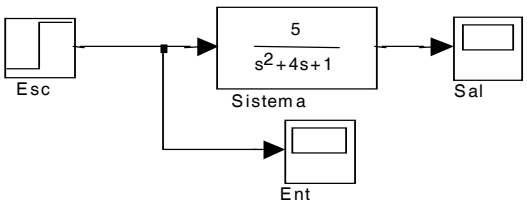


Fig. 3.2 Sistema simulado

Los resultados obtenidos se muestran en la gráfica siguiente:

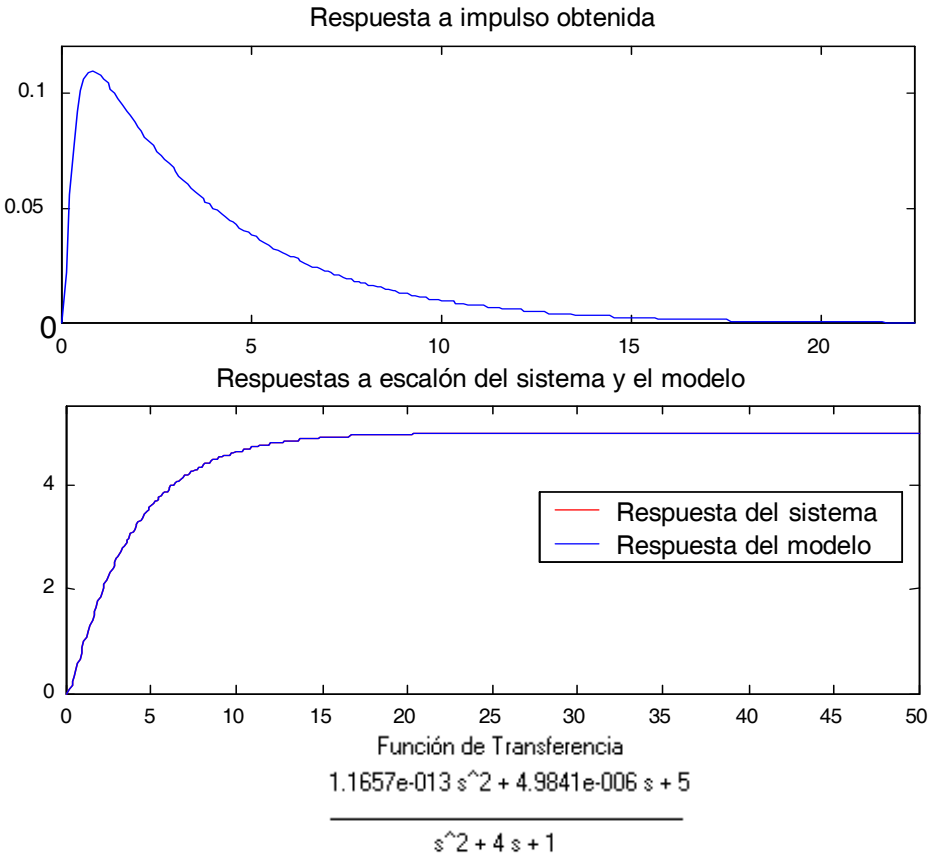


Fig. 3.3 Resultados de la identificación

3.4 Método de deconvolución

Consideremos un sistema lineal, estacionario y estable, con una entrada y una salida, como se muestra en la Fig. 3.4.

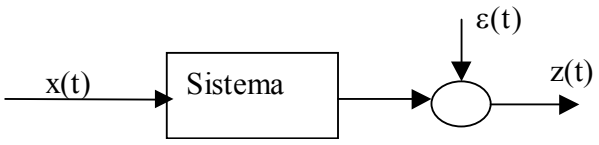


Fig. 3.4 Sistema considerado

Tomando como modelo del sistema la respuesta a impulso se puede escribir:

$$z(t) = \int_0^{T_s} h(\tau)x(t-\tau)d\tau + \varepsilon(t) \tag{3.33}$$

donde T_s es el tiempo de establecimiento del sistema, $h(\tau)$ la respuesta a impulso y $\varepsilon(t)$ el ruido aditivo. El problema se trata ahora de estimar la respuesta a impulso a partir de las mediciones de entrada y salida.

Como paso previo a la solución del problema de estimación de h(t) realicemos la discretización de la ecuación (3.33).

$$z(j\Delta) = \sum_{k=1}^{N} x[(j-k)\Delta] \bullet \Delta \bullet h(k\Delta) + \varepsilon(j\Delta) \quad \text{para } j = 1,2,....M \tag{3.34}$$

donde:

M – Número de puntos de las mediciones de la salida

$T_M = M\Delta$ - tiempo de observación de la salida

N – Número de puntos de la respuesta a impulso

$T_s = N\Delta$ - Tiempo de establecimiento

Ilustremos cómo se toman los valores de z y x, alrededor de un punto de operación, que aparecen en Fig. 3.5.

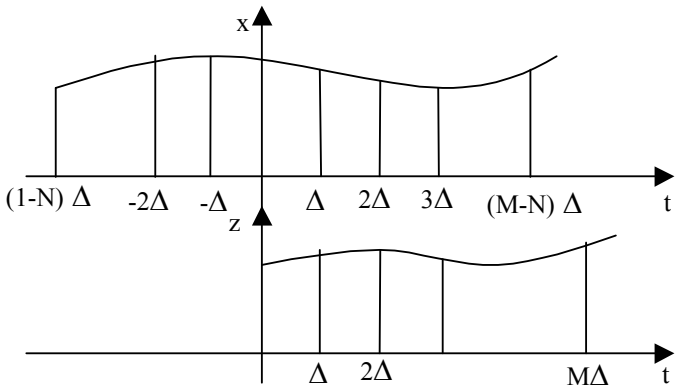


Fig. 3.5 Mediciones de la entrada x(t) y la salida z(t)

Teniendo en cuenta que Δ es constante, la ecuación (3.34) se puede escribir en una forma más sencilla:

$$z_j = \sum_{k=1}^{N} x_{j-k} h_k \quad \text{para } j = 1, 2, ..., M \quad \text{donde consideramos que } x = x * \Delta \tag{3.35}$$

Es importante aclarar que, en general, las mediciones son en desviaciones.

En forma desarrollada tenemos el siguiente sistema de ecuaciones:

$$\begin{aligned}
z_1 &= x_0 h_1 + x_{-1} h2 + x_{-2} h_3 + \cdots + x_{1-N} h_N \\
z_2 &= x_1 h_1 + x_0 h_2 + x_{-1} h_3 + \cdots + x_{2-N} h_N \\
z_3 &= x_2 h_1 + x_1 h_2 + x_0 h_3 + \cdots + x_{3-N} h_N \\
&\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
z_M &= x_{M-1} h_1 + x_{M-2} h_2 + x_{M-3} h_3 + \cdots + x_{M-N} h_N
\end{aligned} \tag{3.36}$$

y en forma matricial se obtiene:

$$z = Ah \tag{3.37}$$

donde :

$$A_{M*N} = \begin{bmatrix} x_0 & x_{-1} & x_{-2} & \cdots & x_{1-N} \\ x_1 & x_0 & x_{-1} & \cdots & x_{2-N} \\ x_2 & x_1 & x_0 & \cdots & x_{3-N} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{M-1} & x_{M-2} & x_{M-3} & \cdots & x_{M-N} \end{bmatrix} \tag{3.38}$$

con $z = [z_1 \;\; z_2 \;\; z_3 \;\; \cdots \;\; z_M]^T \qquad h = [h_1 \;\; h_2 \;\; h_3 \;\; \cdots \;\; h_N]^T$

Este sistema de ecuaciones se puede resolver de varias formas, por supuesto se recomienda usar métodos de mínimos cuadrados (Matlab resuelve por esta vía usando h=A\z). Sin embargo, si suponemos que tomamos las mediciones desde t = 0, que es el caso en el cual se excita al sistema en el instante en que se toma como referencia el tiempo y β_i el estimado de h_i, el sistema de ecuaciones toma la forma sencilla:

$$\begin{aligned} z_1 &= x_0\beta_1 \\ z_2 &= x_1\beta_1 + x_0\beta_2 \\ z_3 &= x_2\beta_1 + x_1\beta_2 + x_0\beta_3 \\ &\cdots\cdots\cdots\cdots\cdots\cdots \\ z_n &= x_{n-1}\beta_1 + x_{n-2}\beta_2 + \cdots + x_0\beta_n \end{aligned} \tag{3.39}$$

Si bien es cierto que se recomienda el uso de mínimos cuadrados para la solución de este problema por la presencia de ruido en el sistema así como por el error de discretización, se puede intentar (si el ruido es pequeño) la solución de una forma muy sencilla:

$$\begin{aligned} \beta_1 &= \frac{z_1}{x_0} \\ \beta_2 &= \frac{z_2 - x_1\beta_1}{x_0} \\ \beta_3 &= \frac{z_3 - x_2\beta_1 - x_1\beta_2}{x_0} \\ &\cdots\cdots\cdots\cdots\cdots \end{aligned} \tag{3.40}$$

$$\beta_i = \frac{z_i - \left(\sum_{j=1}^{i-1} x_{i-j}\beta_j \right)}{x_0}$$

. . . . . .

$$\beta_n = \frac{z_n - \left(\sum_{j=1}^{n-1} x_{n-j}\beta_j \right)}{x_0} \qquad (3.41)$$

Por el método de los mínimos cuadrados se necesitaría primero crear la matriz A y luego resolver el sistema z = Ah. El inconveniente que tiene es que la matriz A tiene el orden igual a la cantidad de valores de la entrada, es decir, puede ser grande.

Los algoritmos de solución en ambos casos son sencillos y evidentes. La experiencia es que se obtienen buenos resultados con ambos procedimientos.

Se confeccionó un programa Matlab (anexo # 5) y se probó con un ejemplo en Simulink.

Los parámetros de simulación fueron los siguientes:

Muestreo de 0.5 unidades de tiempo
Método de solución ode5
Tiempo final 20 unidades de tiempo
Una señal en la entrada compuesta por una sinusoide de amplitud 0.2 y frecuencia 0.1 sumada a un escalón unitario.

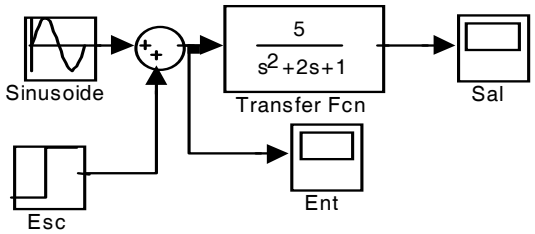


Fig. 3.6 Esquema de simulación

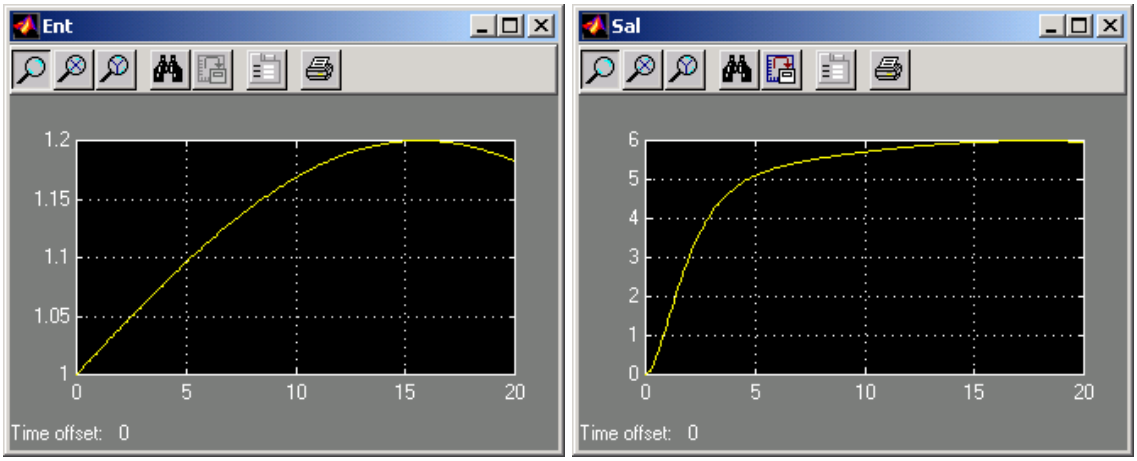


Fig. 3.7 Entrada y Salida del sistema

Los resultados del programa fueron los siguientes:

Transfer function: Transfer function:

0.2343 z + 0.2188 Sampling time: 0.1 0.000263 s + 5.003

---------- ----------

z^2 - 1.81 z + 0.8188 s^2 + 1.999 s + 0.9999

Está claro que se pudo aumentar la precisión disminuyendo el tiempo de muestreo.

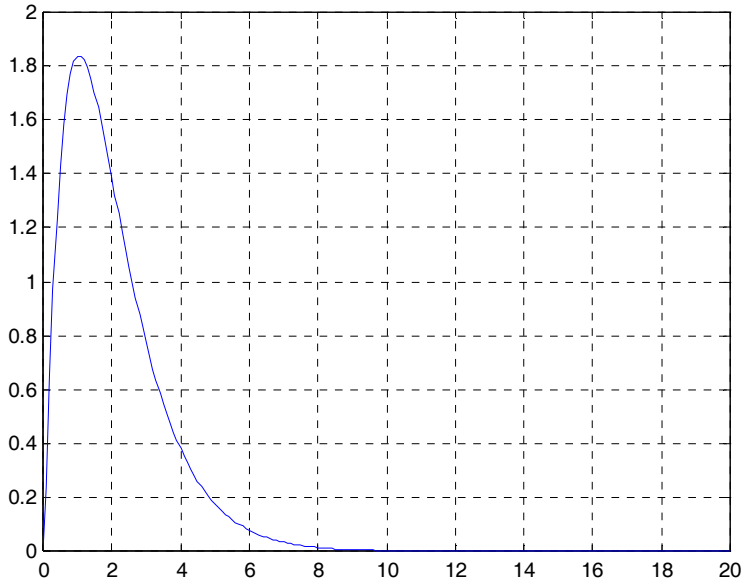


Fig. 3.8 Respuesta a impulso del sistema

3.5 Método de integración gradual.

Para la obtención del modelo matemático que describe el comportamiento de un sistema, es necesario excitar al mismo en su entrada con señales de naturaleza conocida. Algunas señales muy útiles para este fin son aquéllas que se originan y terminan en estado estacionario, como las que se muestran en la figura 3.9.

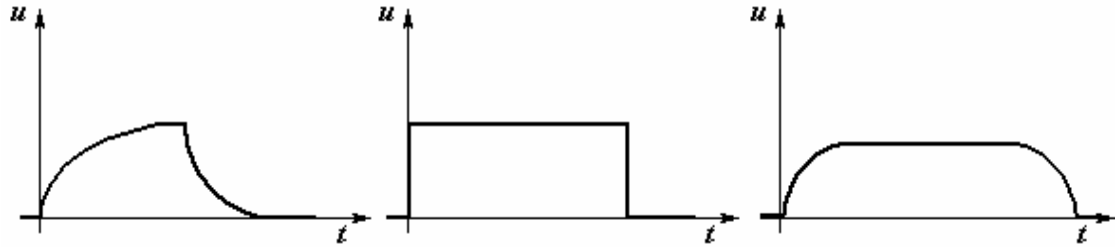


Figura 3.9. Señales con origen y terminación en estado estacionario.

Suponiendo estado estacionario antes y después de las mediciones, entonces se cumplen las siguientes condiciones iniciales para la entrada *u(t)* y la salida *y(t)*.

$$\begin{aligned} y(0) = y(\infty);\ \dot{y}(0) = \dot{y}(\infty) = \cdots = y^{(n)}(0) = 0 \\ u(0) = u(\infty);\ \dot{u}(0) = \dot{u}(0) = \cdots = u^{(m)}(0) = 0 \end{aligned} \tag{3.42}$$

Supongamos el siguiente modelo:

$$a_n y^{(n)}(t) + a_{n-1} y^{(n-1)}(t) + \cdots + a_1 \dot{y}(t) + a_0 y(t) = u(t) \tag{3.43}$$

El problema principal consistiría en determinar los parámetros a_0, a_1, a_2,…,a_n. El método de integración gradual lo resuelve mediante la integración. Así, integrando la ecuación (3.43) de 0 a ∞, suponiendo que vale (3.42), se obtiene:

$$a_0 \int_0^\infty y\,dt = \int_0^\infty u\,dt \Rightarrow a_0 = \frac{\int_0^\infty u\,dt}{\int_0^\infty y\,dt} \tag{3.44}$$

Integrando la ecuación (3.42) primero de *t* a ∞ y luego de 0 a ∞ se obtiene a_1:

$$a_1 \int_0^{\infty}\int_t^{\infty} \dot{y}dt^2 + a_0 \int_0^{\infty}\int_t^{\infty} ydt^2 = \int_0^{\infty}\int_t^{\infty} udt^2$$

$$a_1 = \frac{1}{\int_0^{\infty} ydt}\left[a_0 \int_0^{\infty}\int_t^{\infty} ydt^2 - \int_0^{\infty}\int_t^{\infty} udt^2 \right] \tag{3.45}$$

El cálculo del resto de los coeficientes es similar. Por ejemplo, a_2 se puede calcular como

$$a_2 = \frac{1}{\int_0^{\infty} ydt}\left[\int_0^{\infty}\int_t^{\infty}\int_t^{\infty} udt^3 - a_0 \int_0^{\infty}\int_t^{\infty}\int_t^{\infty} ydt^3 + a_1 \int_0^{\infty}\int_t^{\infty} ydt^2 \right] \tag{3.46}$$

En la Figura 3.10 se presenta la interpretación gráfica de la señal de salida.

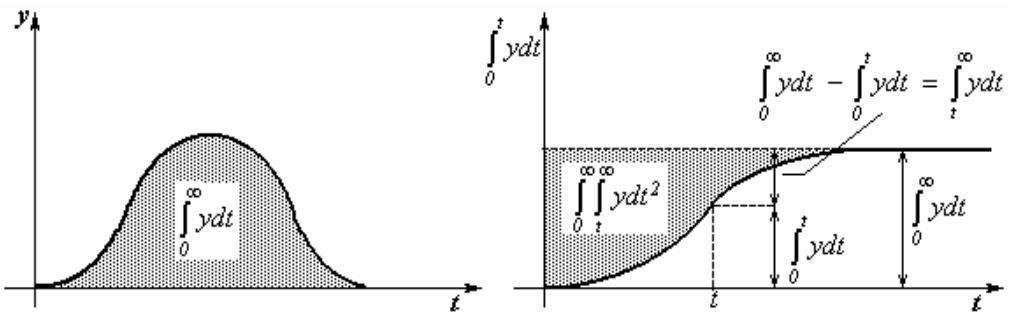


Figura 3.10 Interpretación gráfica de la señal de salida

En el caso en que la señal de entrada comience y termine en estado estacionario, pero que no se cumpla que y(0)= y(∞), u(0)= u(∞), como son los casos que se muestran en la Figura 3.11, se procede de la siguiente forma:

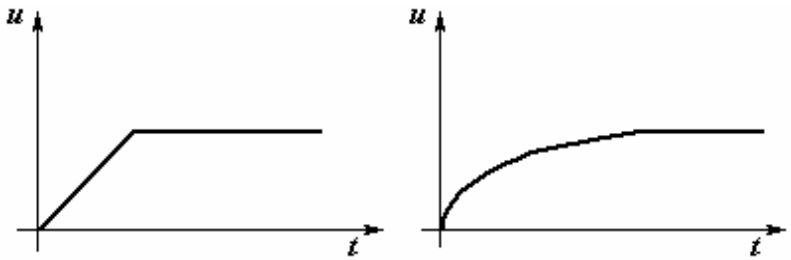


Figura 3.11. Señales de entradas para las cuales $u(0) \neq u(\infty)$.

De la ecuación (3.43) se resta la ecuación para estado estacionario y se obtiene

$$a_n \overset{(n)}{y}(t) + \cdots + a_2 \dot{y}(t) + a_1 \dot{y}(t) + a_0 [y(\infty) - y(t)] = -[u(\infty) - u(t)] \tag{3.47}$$

Luego de integrar de 0 a ∞ se puede obtener:

$$a_1 = \frac{a_0 \int_0^\infty [y(\infty) - y(t)]dt - \int_0^\infty [u(\infty) - u(t)]dt}{y(\infty)} \tag{3.48}$$

y, luego de una integración doble se obtiene la ecuación (3.49):

$$a_2 = \frac{1}{y(\infty)} \left(a_0 \int_0^\infty \int_t^\infty [y(\infty) - y(t)]dt^2 - a_1 \int_0^\infty [y(\infty) - y(t)]dt - \int_0^\infty \int_t^\infty [u(\infty) - u(t)]dt^2 \right) \tag{3.49}$$

En el caso de que la señal de entrada fuera un escalón, el cálculo de estos parámetros se simplificaría, pues

$$\int_0^\infty [u(\infty) - u(t)]dt = 0 \tag{3.50}$$

y

$$\int_0^\infty \int_t^\infty [u(\infty) - u(t)]dt^2 = 0 \tag{3.51}$$

debido a que $u(\infty) = u(t)$ para $t>0$. a_0 se determina directamente de la relación:

$$a_0 = \frac{u(\infty)}{y(\infty)} \tag{3.52}$$

3.5.1 Integración gradual para un conjunto de datos de entrada-salida.

Se supone el modelo:

$$y'' + a_1 y' + a_0 y = b_0 u + b_1 u' \tag{3.53}$$

Si se poseen k datos para el tiempo, así como los respectivos valores de entrada y salida, después de integrar de 0 a t_i, i=1,…,k, se obtiene:

$$
\begin{aligned}
&a_1 \int_0^{t_1} y\,dt + a_0 \int_0^{t_1}\int_0^{\tau} y\,dt\,d\tau - b_0 \int_0^{t_1}\int_0^{\tau} u\,dt\,d\tau - b_1 \int_0^{t_1} u\,dt = -y(t_1) \\
&\vdots \\
&a_1 \int_0^{t_k} y\,dt + a_0 \int_0^{t_k}\int_0^{\tau} y\,dt\,d\tau - b_0 \int_0^{t_k}\int_0^{\tau} u\,dt\,d\tau - b_1 \int_0^{t_k} u\,dt = -y(t_k)
\end{aligned}
\qquad (3.54)
$$

es decir,

$$
\begin{bmatrix}
\int_0^{t_1} y\,dt & \int_0^{t_1}\int_0^{\tau} y\,dt\,d\tau & -\int_0^{t_1}\int_0^{\tau} u\,dt\,d\tau & -\int_0^{t_1} u\,dt \\
\vdots & \vdots & \vdots & \vdots \\
\int_0^{t_k} y\,dt & \int_0^{t_k}\int_0^{\tau} y\,dt\,d\tau & -\int_0^{t_k}\int_0^{\tau} u\,dt\,d\tau & -\int_0^{t_k} u\,dt
\end{bmatrix}
\begin{bmatrix} a_1 \\ a_0 \\ b_0 \\ b_1 \end{bmatrix}
=
\begin{bmatrix} -y(t_1) \\ : \\ : \\ -y(t_k) \end{bmatrix}
\qquad (3.55)
$$

Con estructura matricial-vectorial:

$$A\,p = y \qquad (3.56)$$

donde ***A***[k,m]; **p**[m,l]; **y**[k,l]. Las columnas de la matriz **A** se pueden calcular con la ayuda de la matriz de integración **J** según:

$$
\begin{bmatrix} \int_0^{t_1} y\,dt \\ : \\ : \\ \int_0^{t_k} y\,dt \end{bmatrix}
= \boldsymbol{J}
\begin{bmatrix} y(t_1) \\ : \\ : \\ y(t_k) \end{bmatrix}
\qquad (3.57)
$$

Para simplificar la notación supondremos ahora:

$$y(i) = y(t_i).$$

Para la integral doble vale:

$$\begin{bmatrix} \int_0^{t_1}\int_0^{\tau} y\,d\tau dt \\ \vdots \\ \int_0^{t_k}\int_0^{\tau} y\,d\tau dt \end{bmatrix} = \boldsymbol{J}\begin{bmatrix} \int_0^{t_1} y\,dt \\ \vdots \\ \int_0^{t_k} y\,dt \end{bmatrix} = \boldsymbol{J}^2\begin{bmatrix} y(1) \\ \vdots \\ y(k) \end{bmatrix} \tag{3.58}$$

3.5.2 Matriz de Simpson.

La estructura de la matriz de integración se obtiene de la siguiente forma. Se divide el transcurso de las señales de entrada y salida en segmentos iguales de tiempo $\Delta t = t_{i+1}-t_i$ (i=0,1,2,.....k). Al aplicar la llamada Regla de Simpson se obtiene, por ejemplo, para la señal de salida y:

$$\begin{aligned} &\int_0^{\Delta t} y\,dt \approx \frac{\Delta t}{12}\left(5y(0)-8y(1)-y(2)\right) \\ &\int_0^{2\Delta t} y\,dt \approx \frac{\Delta t}{12}\left(4y(0)+16y(1)+4y(2)\right) \\ &\int_0^{k\Delta t} y\,dt \approx \frac{\Delta t}{12}\left(4y(0)+16y(1)+8y(2)+16y(3)+8y(4)+....+4y(k)\right) \end{aligned} \tag{3.59}$$

Sobre la base de estas relaciones y sabiendo que y(0) = 0, es posible establecer la matriz de integración ***J*** según:

$$\boldsymbol{J} = \frac{\Delta t}{12}\begin{array}{c} \begin{matrix} y_1 & y_2 & y_3 & y_4 & \cdot & \cdot & y_k \end{matrix} \\ \begin{bmatrix} 8 & -1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ 16 & 4 & \cdot & \cdot & \cdot & \cdot & \cdot \\ 16 & 9 & 8 & -1 & \cdot & \cdot & \cdot \\ 16 & 8 & 16 & 4 & \cdot & \cdot & \cdot \\ 16 & 8 & 16 & 9 & 8 & -1 & \cdot \end{bmatrix} \end{array} \tag{3.60}$$

La elaboración de las filas de la matriz de Simpson demanda un tratamiento diferenciado para el caso de las filas pares e impares. Dado que el método de Simpson asume que el intervalo de integración sea subdivido en un número par k de sub-intervalos de integración, en el caso de que se disponga de un número impar k-1 de sub-intervalos, se necesita hacer

uso de una fórmula de integración auxiliar que, combinada con la de Simpson, proporcione un resultado cuya precisión esté en correspondencia con la precisión dada por las fórmulas estándar de integración de Simpson.

De acuerdo con lo expresado, para la obtención de las filas pares de la matriz de Simpson se emplea la siguiente fórmula:

$$\int_{t_0}^{t_k} y(t)dt = \frac{h}{12}\left[4y_0 + 16\left(y_1 + y_3 + \cdots y_{k-1}\right) + 8\left(y_2 + y_4 + \cdots y_{k-2}\right) + 4y_k\right] - \frac{t_k - t_0}{180} h^4 \overset{iv}{y}(\varepsilon)$$

$$\varepsilon \in \left[t_0, t_k\right] \qquad (3.61)$$

donde y(t): expresión genérica, que admite hasta cuarta derivada, para las ordenadas de entrada y salida del proceso que se identifica

t_0 y t_k: instantes de tiempo inicial y final del registro de las ordenadas de las entrada y salida del proceso que se identifica.

$h = \Delta t = t_{i+1} - t_{i:}$ magnitud del paso de integración.

y_i: ordenadas de la señal de entrada y salida del proceso en experimentación.

$\overset{iv}{y}(\varepsilon)$: cuarta derivada de la expresión genérica *y(t)* evaluada en algún punto ε del intervalo de integración $[t_0, t_k]$.

$\frac{t_k - t_0}{180} h^4 \overset{iv}{y}(\varepsilon)$: término de error de la fórmula de Simpson.

:

Para los elementos de las filas impares de la matriz de Simpson se utiliza la fórmula auxiliar:

$$\int_{t_k - h}^{t_k} y(t)dt = \frac{h}{12}\left[5y_k + 8y_{k-1} - y_{k-2}\right] - \frac{h^4}{24} y'''(\varepsilon) \qquad (3.62)$$

donde:

$\frac{h^4}{24} y'''(\varepsilon)$: término de error de la fórmula auxiliar.

Esta fórmula resulta de integrar al polinomio de interpolación inverso de Newton de tercer

orden en correspondencia con lo estipulado para obtener la fórmula de integración canónica de Simpson que es básica para obtener la ecuación (3.42). El cálculo de la integral para el caso de las filas impares se efectúa según la expresión:

$$\int_{t_0}^{t_k-h} y(t)dt = \int_{t_0}^{t_k} y(t)dt = \int_{t_k-h}^{t_k} y(t)dt \tag{3.63}$$

Sustituyendo (3.61) y (3.62) en (3.63), se obtiene:

$$\int_{t_0}^{t_k-h} y(t)dt = \frac{h}{12}\left[4y_0 + 16\left(y_1 + y_3 + .. + y_{k-3}\right) + 8y_{k-1} + 8\left(y_2 + y_4 + .. + y_{k-4}\right) + 9y_{k-2} - y_k\right] \tag{3.64}$$

Para obtener la primera fila de la matriz de Simpson se parte de las siguientes expresiones:

$$\int_{t_0}^{t_2} y(t)dt = \frac{h}{12}\left[4y_0 + 16y_1 + 4y_2\right] \text{ (Fórmula canónica de Simpson)} \tag{3.65}$$

$$\int_{t_2-h}^{t_2} y(t)dt = \frac{h}{12}\left[-y_0 + 8y_1 + 5y_2\right] \text{ (Fórmula auxiliar)} \tag{3.66}$$

de cuya resta se obtiene:

$$\int_{t_0}^{t_2-h} y(t)dt = \frac{h}{12}\left[5y_0 + 8y_1 - y_2\right] \tag{3.67}$$

Las ecuaciones (3.61), (3.64) y (3.67) permiten formar los elementos para las filas pares e impares de la matriz de Simpson requerida por el método de integración gradual.

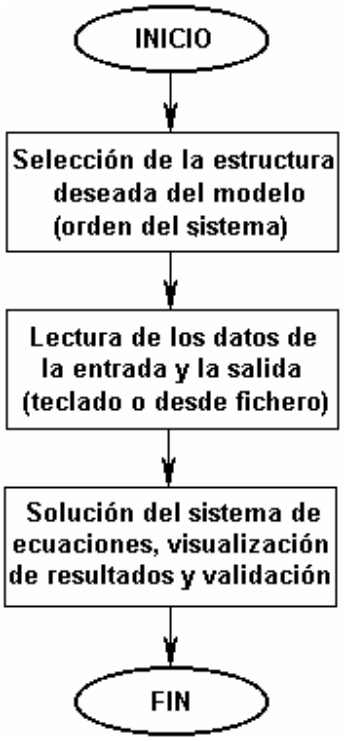


Figura 3.12. Algoritmo de cálculo.

En el caso de trabajar con un modelo de la forma:

$$y' + a_0 y = b_0 u \tag{3.68}$$

el sistema de ecuaciones quedaría de la forma siguiente:

$$\begin{bmatrix} \int_0^{t_1} y dt & -\int_0^{t_1} u dt \\ \vdots & \vdots \\ \int_0^{t_k} y dt & -\int_0^{t_k} u dt \end{bmatrix} \begin{bmatrix} a_0 \\ b_0 \end{bmatrix} = \begin{bmatrix} -y(t_1) \\ : \\ : \\ -y(t_k) \end{bmatrix} \tag{3.69}$$

Para el caso en que se desee trabajar con un modelo de 2do. orden de la forma:

$$y'' + a_1 y' + a_0 y = b_0 u + b_1 u' \tag{3.70}$$

El sistema sería:

$$\begin{bmatrix} \int_0^{t_1} y dt & \int_0^{t_1}\int_0^{\tau} y dt d\tau & -\int_0^{t_1}\int_0^{\tau} u dt d\tau & -\int_0^{t_1} u dt \\ \vdots & \vdots & \vdots & \vdots \\ \int_0^{t_k} y dt & \int_0^{t_k}\int_0^{\tau} y dt d\tau & -\int_0^{t_k}\int_0^{\tau} u dt d\tau & -\int_0^{t_k} u dt \end{bmatrix} \begin{bmatrix} a_1 \\ a_0 \\ b_0 \\ b_1 \end{bmatrix} = \begin{bmatrix} -y(t_1) \\ : \\ : \\ -y(t_k) \end{bmatrix} \tag{3.71}$$

El programa desarrollado (Anexo # 6), una vez resuelto el sistema de ecuaciones, según sea el caso, y teniendo como resultado los valores de los coeficientes, visualiza en la pantalla los gráficos correspondientes a la salida medida y a la salida calculada a partir de la señal de excitación aplicada en la entrada al modelo obtenido. Se visualiza la desviación estándar del error entre los valores de la señal de salida medida y la obtenida a través de la identificación. El usuario tiene la posibilidad de ajustar el modelo cambiando los coeficientes calculados con el objetivo de que la respuesta del modelo sea lo más cercana posible a la salida medida.

Conjuntamente con el programa al que se hizo referencia, se creó la función INTGR con el objetivo de añadirla al Toolbox de Identificación de sistemas dinámicos que ofrece el MATLAB. Esta función es compatible con el paquete de funciones ya existentes, entregando los resultados en la estructura estándar conocida como matriz THETA. Esta matriz contiene información acerca de la estructura del modelo, parámetros estimados, precisión de la estimación, así como la matriz de la covarianza de los parámetros calculados.

Ejemplos de uso del programa.

A continuación se tratan algunos ejemplos con el objetivo de comprobar la eficacia de este método.

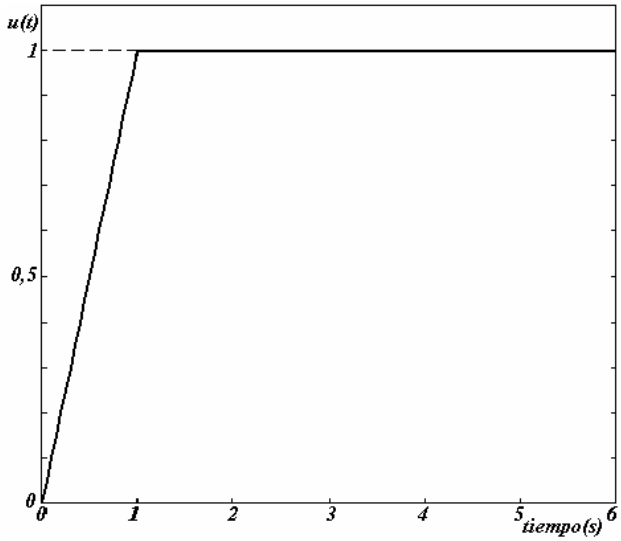


Figura 3.13. Señal de excitación.

La señal de excitación en todos los casos analizados fue la siguiente:

$$u(t) = \begin{cases} 0, & t < 0 \\ t, & 0 \le t \le 1 \\ 1, & t > 1 \end{cases}$$

y su representación gráfica se muestra en la Figura 3.13.

Sistema de 1er. orden.

La función de transferencia empleada fue:

$$F_1(s) = \frac{2}{s+2}$$

Este sistema fue excitado con la señal de entrada vista anteriormente, obteniéndose 50 juegos de valores de entrada-salida cada 0,05 s.

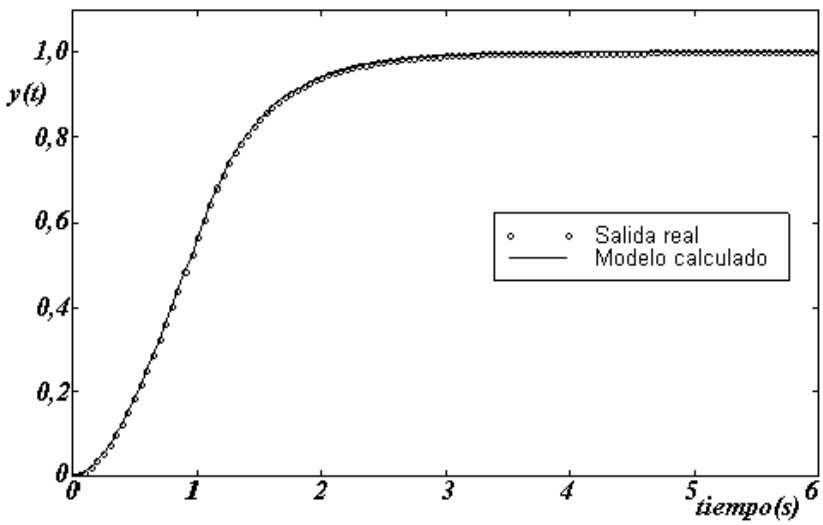


Figura 3.14. Respuesta del modelo calculado y salida real del sistema de primer orden.

Luego se realizó la identificación obteniendo la función de transferencia:

$$F(s) = \frac{1{,}9945}{s + 1{,}9932}$$

con una desviación estándar de 2,1544e-6.

Sistema de 2do. Orden.

Se tomó como ejemplo el siguiente sistema de segundo orden con raíces complejas conjugadas

$$F_2(s) = \frac{2s + 1}{s^2 + 2s + 5}$$

obteniéndose como resultado la función de transferencia:

$$F_2(s) = \frac{1{,}9981s + 0{,}9943}{s^2 + 2{,}0063s + 4{,}9888}$$

con una desviación estándar de 2,7374e-4. La respuesta se muestra en la Figura 3.15.

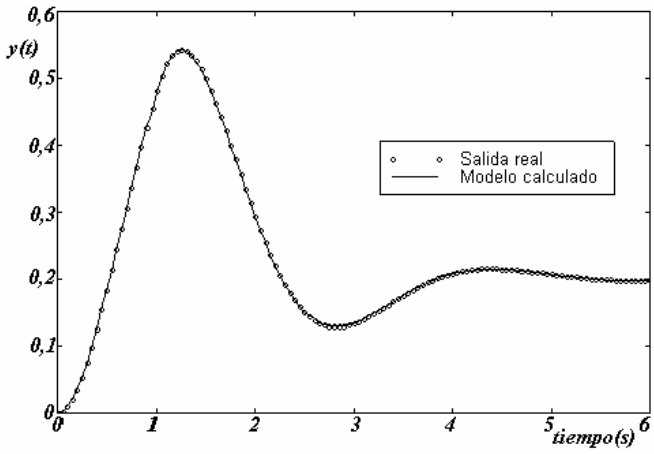


Figura 3.15 Salida real y del modelo para un sistema de segundo orden.

Comentarios finales.

- La aplicación del método en sistemas de 1er, 2do y 3er órdenes permite obtener resultados satisfactorios, obteniéndose errores promedio muy pequeños.
- Se obtienen de forma directa los coeficientes del modelo continuo de una planta.
- Es necesario una selección correcta del tiempo de muestreo.
- En el caso que se trabaje con señales muy ruidosas se recomienda darle un tratamiento preliminar o filtrado de la información que se va a procesar, de forma tal que los datos que serán analizados reflejen de la forma más exacta posible el comportamiento real del sistema.

3.6 Método de integración múltiple

El sistema, motivo de estudio, es lineal, invariante, continuo y de parámetros concentrados, de una entrada y una salida.

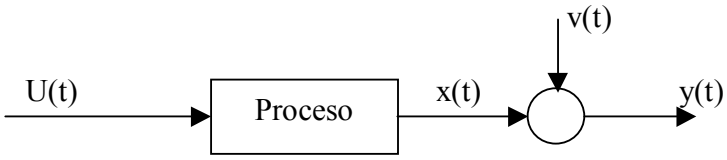


Fig. 3.16 Sistema considerado

u(t) → señal de entrada

x(t) → señal de salida ideal

y(t) → señal de salida influida por el ruido

v(t) → señal de ruido

Todas las señales son funciones continuas, reales, de variables continuas en t ε <0, ∞>.

Supondremos el modelo del sistema de la siguiente forma:

$$G(p) = \frac{1 + b_1 p + b_2 p^2 + ... + b_n p^n}{a_0 + a_1 p + a_2 p^2 + ... + a_n p^n} \qquad (3.72)$$

Como es evidente, siempre es posible llegar a esta forma de la función de transferencia.

La estructura general del modelo se presenta en la figura 3.17.

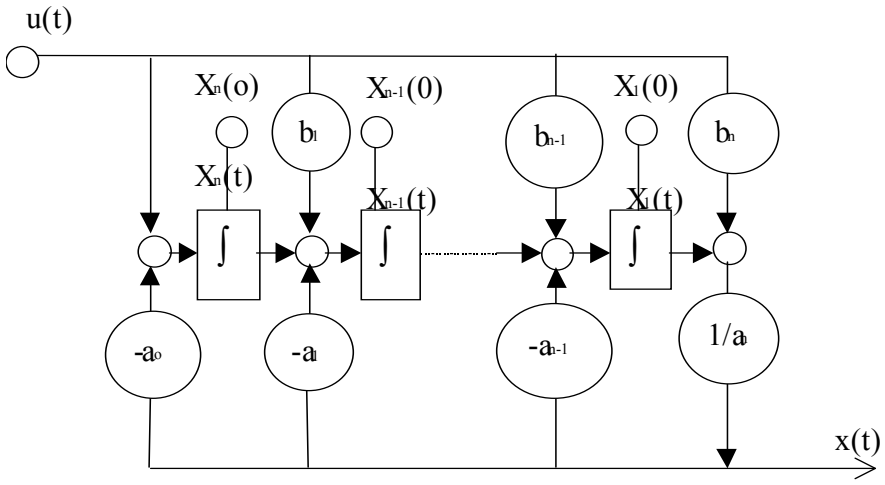


Fig. 3.17 Estructura general del modelo

La ecuación diferencial que representa el comportamiento dinámico del sistema sería:

$$b_n \frac{d^n u(t)}{dt^n} + \cdots + b_1 \frac{du(t)}{dt} + u(t) = a_n \frac{d^n x(t)}{dt^n} + \cdots a_o x(t) \tag{3.73}$$

La expresión para $x_1(t)$ para un paso de x = 0 a x = T sería:

$$x_1(T) = x_1(0) + x_2(0)T + \cdots + x_n(0)\frac{T^{n-1}}{(n-1)!} +$$

$$\sum_0^T \int \cdots \int = \begin{cases} + b_{n-1}\int_0^T u(\tau)d\tau + b_{n-2}\int_0^T\int u(\tau)d^2\tau + \cdots + \int_0^T \cdots \int u(\tau)d^n\tau - \\ - a_{n-1}\int_0^T x(\tau)d\tau - a_{n-2}\int_0^T\int x(\tau)d^2(\tau) - \cdots - a_0\int_0^T \cdots \int x(\tau)d^n(\tau) \end{cases} \tag{3.74}$$

De manera que podríamos escribir, para simplificar,

$$x_1(T) = x_1(0) + x_2(0)T + \cdots + x_n(0)\frac{T^{n-1}}{(n-1)!} + \sum_0^T \int \cdots \int$$

$$x_1(2T) = x_1(0) + x_2(0)2T + \cdots + x_n(0)\frac{(2T)^{n-1}}{(n-1)!} + \sum_0^{2T} \int \cdots \int \tag{3.75}$$

$$\vdots$$

$$x_1(nT) = x_1(0) + x_2(0)nT + \cdots + x_n(0)\frac{(nT)^{n-1}}{(n-1)!} + \sum_0^{nT} \int \cdots \int$$

O de forma matricial

$$\begin{bmatrix} x_1(T) \\ x_1(2T) \\ \vdots \\ \vdots \\ \vdots \\ x_1(nT) \end{bmatrix} = \begin{bmatrix} 1 & 1 & . & . & 1 \\ 1 & 2 & . & . & 2^{n-1} \\ . & . & . & . & . \\ . & . & . & . & . \\ 1 & n & . & . & n^{n-1} \end{bmatrix} \begin{bmatrix} x_1(0) \\ x_2(0)\,T \\ \vdots \\ \vdots \\ \vdots \\ x_n(0)\,nT \end{bmatrix} + \begin{bmatrix} \sum_0^T \int \cdots \int \\ \sum_0^{2T} \int \cdots \int \\ \vdots \\ \vdots \\ \vdots \\ \sum_0^{nT} \int \cdots \int \end{bmatrix} \tag{3.76}$$

El problema consiste ahora en eliminar los estados $x_1, x_2, \ldots, x_n$ del sistema de ecuaciones. La primera matriz del lado derecho es la conocida matriz cuadrada de orden n de Vandermond, cuyos elementos se calculan según $V_{i,j} = i^{(j-1)}$.

Supongamos ahora que $p = [p_1 \;\; p_2 \;\; p_3 \ldots \;\; p_n]$ es la primera fila de V^{-1}, entonces

$$p.V = [1 \;\; 0 \;\; \ldots\ldots \; 0]$$

y, luego de multiplicar la ecuación matricial anterior por p tenemos:

$$p\begin{bmatrix} x_1(T) \\ \vdots \\ \vdots \\ x_1(nT) \end{bmatrix} = x_1(0) + p\begin{bmatrix} \sum \int_0^T \cdots \int \\ \sum \int_0^{2T} \cdots \int \\ \vdots \\ \sum \int_0^{nT} \cdots \int \end{bmatrix} \tag{3.77}$$

De la estructura del sistema podemos escribir:

$$x_1(t) = a_n\, x(t) \; - \; b_n\, u(t)$$

de lo que sigue:

$$a_n\begin{bmatrix} -1 & p_1 & \cdots & p_n \end{bmatrix}\begin{bmatrix} x(0) \\ \vdots \\ x(nT) \end{bmatrix} - b_n\begin{bmatrix} -1 & p_1 & \cdots & p_n \end{bmatrix}\begin{bmatrix} u(0) \\ \vdots \\ u(nT) \end{bmatrix} = p\begin{bmatrix} \sum \int_0^T \cdots \int \\ \vdots \\ \sum \int_0^{nT} \cdots \int \end{bmatrix} \tag{3.78}$$

Los elementos de p se pueden encontrar según:

$$p_{i} = (-1)^{i-1}\ C_n^i \quad i = 1\ ....n$$

$$C_n^i = \frac{n!}{i!(n-i)!} \qquad C_n^n = C_n^0 = 1$$

Finalmente el modelo completo o terminado sería:

$$a_n \sum_{i=0}^{n} (-1)^i \cdot C_n^i \cdot x(iT) - b_n \sum_{i=0}^{n} (-1)^i \cdot C_n^i \cdot u(iT) = \sum_{i=1}^{n} (-1)^i \cdot C_n^i \int_0^{iT} \cdots \int u(\tau) d^n\tau +$$

$$+ \sum_{j=0}^{n-1} (a_j \cdot \sum_{i=0}^{n} (-1)^i \cdot C_n^i \cdot \underbrace{\int_0^{iT} \cdots \int}_{n-j} x(\tau) d^{n-j}\tau) + \sum_{j=1}^{n-1} (b_j \cdot \sum_{i=1}^{n} (-1)^i \cdot C_n^i \cdot \underbrace{\int_0^{iT} \cdots \int}_{n-j} u(\tau) d^{n-j}\tau) \tag{3.79}$$

De este modo, hemos obtenido una ecuación con los coeficientes del modelo dinámico como incógnitas. Ahora bastaría tomar diferentes intervalos [0, T] y obtener tantas ecuaciones como quisiera y luego resolver por mínimos cuadrados.

4. Algunos elementos sobre el *toolbox* de Matlab

El Matlab 5.3 ofrece dos *toolbox* sobre identificación: uno temporal y otro frecuencial. Pretendemos mostrar algunos pocos elementos que puedan ser útiles acerca del primero sin abordar la interfaz correspondiente por ser muy simple de usar.

4.1 Modelo lineal general

El modelo lineal general de un sistema puede ser descrito simbólicamente según:

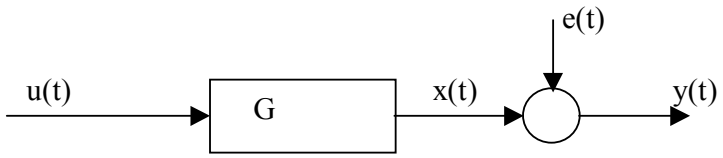


Fig. 4.1 Representación simbólica del sistema

$$y = Gu + He \tag{4.1}$$

que dice que la salida medida **y** es debida a la entrada medida **u** y al ruido **e**. Donde G denota las propiedades dinámicas del sistema, es decir, cómo la salida se forma desde la entrada. Para sistemas lineales se llama función de transferencia entre la entrada y la salida. H refiere las propiedades del ruido y se le llama también modelo del ruido, y describe cómo está formada la perturbación en la salida.

4.1.1 Representación polinomial de la función de transferencia

Una forma útil de representar G y H es en funciones racionales de q^{-1}. El modelo paramétrico ARX (Auto-Regressive eXogen) corresponde a:

$$G(q) = q^{-nk}\,\frac{B(q)}{A(q)} \qquad\qquad H(q) = \frac{1}{A(q)} \tag{4.2}$$

donde B y A son polinomios en el operador de retraso q^{-1}:

$$A(q) = 1 + a_1 q^{-1} + ... + a_{na} q^{-na} \qquad (4.3)$$

$$B(q) = b_1 + b_2 q^{-1} + ... + b_{nb} q^{-nb+1}$$

Los términos na y nb dan cuenta de los órdenes de los polinomios A(q) y B(q) respectivamente y nk es el número de retrasos de la entrada a la salida. Usualmente se escribe el modelo de la siguiente forma:

$$A(q)y(t) = B(q)u(t - nk) + e(t) \qquad (4.4)$$

4.1.2 Modelo ARX

De manera explícita el modelo ARX se expresa según:

$$y(t) + a_1 y(t-1) + ... + a_{na} y(t - na) = b_1 u(t - nk) + b_2 u(t - nk - 1) + ... + b_{nb} u(t - nk - nb + 1) \qquad (4.5)$$

Esto da lugar a un sistema de ecuaciones donde las incógnitas a y b serán los coeficientes de la función de transferencia discreta y que se obtienen según:

Mínimos Cuadrados: Minimiza la suma de los cuadrados de la parte derecha menos la parte izquierda con respecto a los coeficientes a y b. Para esto se usa la función arx del Matlab.

Variable Instrumental: Se determinan a y b de manera tal que el error entre las partes derecha e izquierda no correlaciona con alguna combinación lineal de la entrada. Para esto se usa la función iv4.

4.1.3 Modelo ARMAX

En la estructura ARMAX (AutoRegressive Moving Average eXogen) se introduce el polinomio C(q) al modelo ARX:

$$A(q)y(t) = B(q)u(t - nk) + C(q)e(t) \tag{4.6}$$

donde C se puede expresar según:

$$C(q) = 1 + c_1 q^{-1} + ... + c_{nc} q^{-nc} \tag{4.7}$$

Esto da lugar a un sistema de ecuaciones donde las incógnitas son los coeficientes del modelo discreto, cuyas soluciones (de este modelo y los posteriores) se obtienen por predicción del error con el Método de Máxima Verosimilitud. Para esto se usa la función armax.

4.1.4 Modelo OE

La estructura Output – Error (OE) se presenta de forma siguiente:

$$y(t) = \frac{B(q)}{F(q)} u(t - nk) + e(t) \tag{4.8}$$

$$\text{con} \quad F(q) = 1 + f_1 q^{-1} + ... + f_{nf} q^{-nf} \tag{4.9}$$

4.1.5 Modelo BJ

La llamada estructura de Box – Jenkins está dada por:

$$y(t) = \frac{B(q)}{F(q)} u(t - nk) + \frac{C(q)}{D(q)} e(t) \quad \text{con} \quad D(q) = 1 + d_1 q^{-1} + ... + d_{nd} q^{-nd} \tag{4.10}$$

Todos estos modelos son casos particulares de la estructura general:

$$A(q)y(t) = \frac{B(q)}{F(q)}u(t-nk) + \frac{C(q)}{D(q)}e(t) \qquad (4.11)$$

donde:

La estructura AR corresponde con nb = nf = nc = nd = 0 y además u(t) = 0.
La estructura ARX se obtiene haciendo nd = nc = nf = 0.
La estructura ARMAX corresponde a nf = nd = 0.
La estructura Output – Error (OE) con na = nc = nd = 0.
El modelo Box – Jenkins corresponde a na = 0

De las funciones existentes en Matlab solo mencionaremos algunas y en sus formas más usadas para el cálculo de la matriz THETA (se puede tener más información ejecutando **help theta** en Matlab) que provee toda la información necesaria para la identificación).

AR

Para la estructura AR se pueden usar ambos métodos:

```
% n es el orden del modelo AR
th = ar(y, n) % se calcula según mínimos cuadrados
th = ivar(y, n) % se calcula según variable instrumental
```

ARX

```
th = arx(z,[na nb nk]) % z es una matriz z = [y u], siendo y la salida y u la entrada
th = iv4(z,[na nb nk])% Por el método de Variable Instrumental
```

Para sistemas realimentados se recomienda usar variable instrumental.

ARMAX

```
th = armax(z, [na nb nc nk])
```

OE

```
th = oe(z, [nb nf nk])
```

BJ

th = bj(z, [nb nc nd nf nk])

4.1.6 Ejemplo de sistema con una entrada una salida (SISO)

Supondremos ruido a la entrada.

Con ayuda del Simulink del Matlab realizaremos la identificación del sistema lineal:

$$G(s) = \frac{s+1}{s^3 + s^2 + 5s + 1}$$

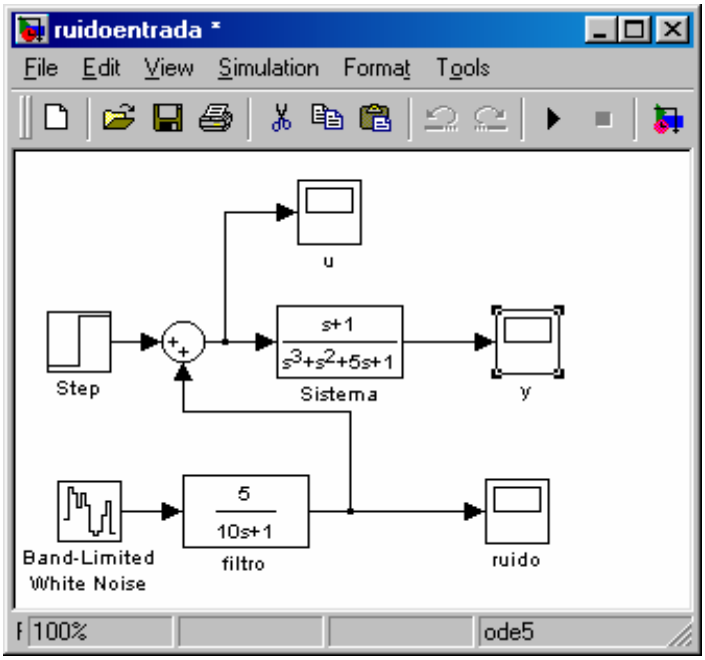


Fig. 4.2 Esquema Simulink

Se realizó una simulación en Matlab/Simulink, según se muestra en la fig 4.2

Como entrada se le aplica un escalón 2u(t-2) y se toma un intervalo de tiempo fijo (fixed step size) de 0.1 unidades de tiempo y un tiempo de simulación de 20 u. Se le adiciona

ruido a la entrada con el bloque *Band-Limited White Noise* (*Noise Power* = 0.5, *seed* = 23341) a través de una función de transferencia:

$$\frac{5}{10s+1}$$

La entrada **u** y la salida **y** se llevan al *workspace* del MATLAB.

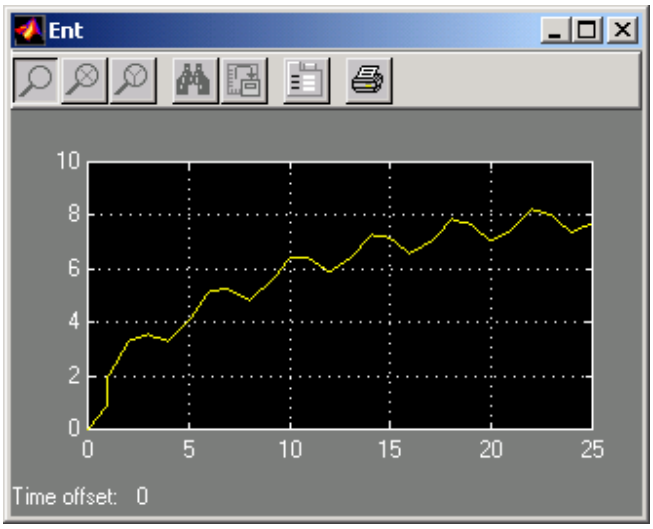


Fig. 4.3 Entrada del sistema

Probemos con un modelo ARX con na = nb = 3 y nk = 1. Para esto se preparó el programa arxpl (Anexo # 7).

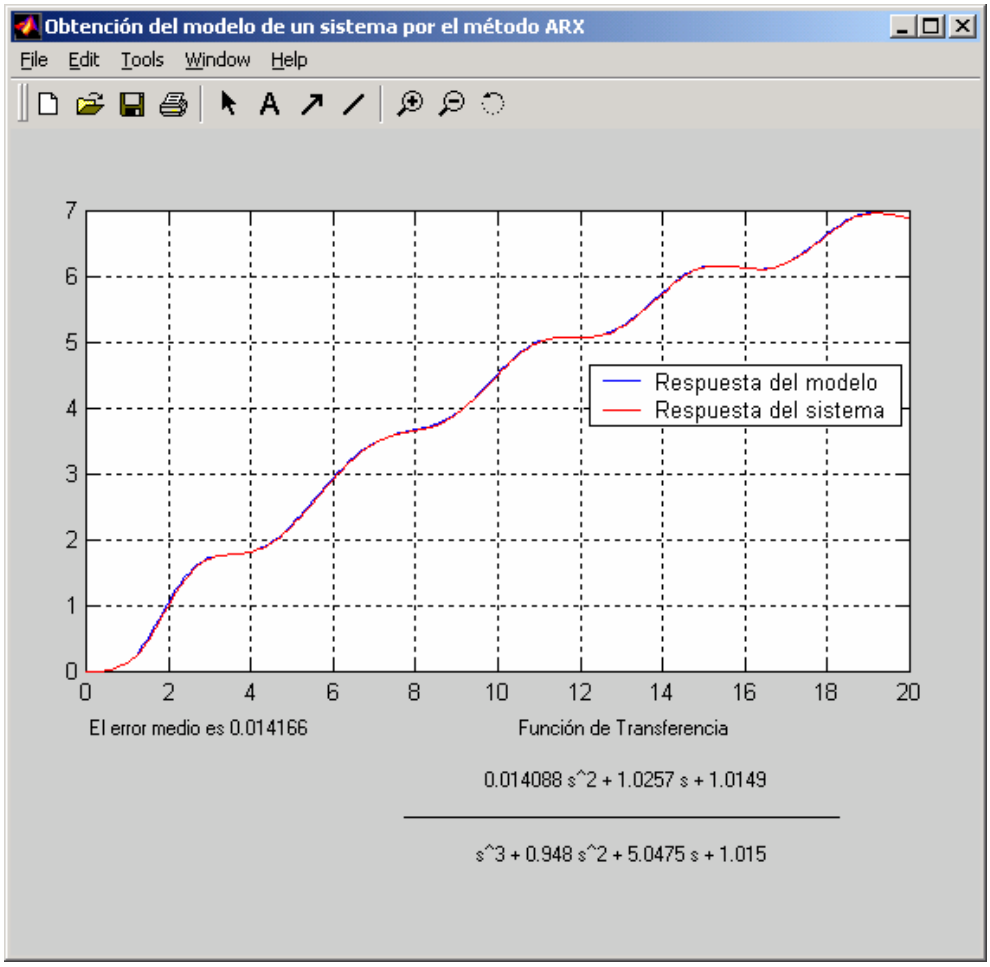


Fig. 4.4 Respuesta de la identificación

Nota: A este programa se le podrían adicionar cálculo de errores por diferentes métodos. Por otro lado se obtiene la expresión en polos y ceros con el objetivo de investigar si el orden propuesto no es mayor que el real, pues en el caso de que sea mayor se vería la posibilidad de cancelación y se podría probar con un orden reducido.

La comparación de las respuestas del modelo y el sistema se aprecia en la figura 4.4.

Ejemplo2. Supondremos ruido a la salida

Supondremos el mismo sistema adicionándole el mismo ruido a la salida, según se muestra en la figura 4.5, pero a través de la función de transferencia:

$$\frac{0.05}{s^2+s+1}$$

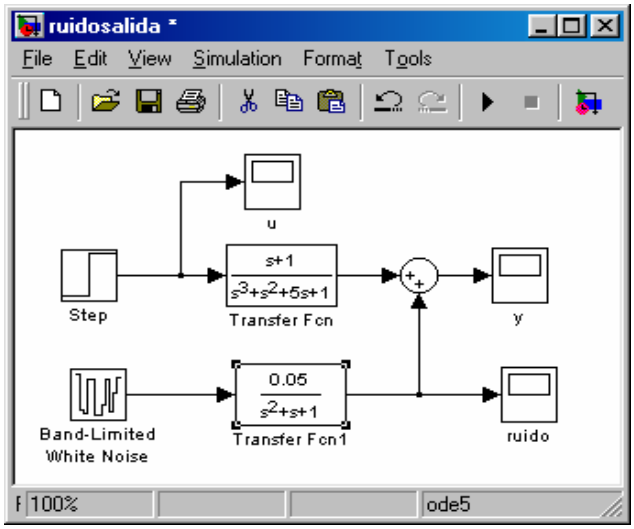


Fig. 4.5 Sistema con ruido a la salida

Probemos con el mismo programa arxpl.

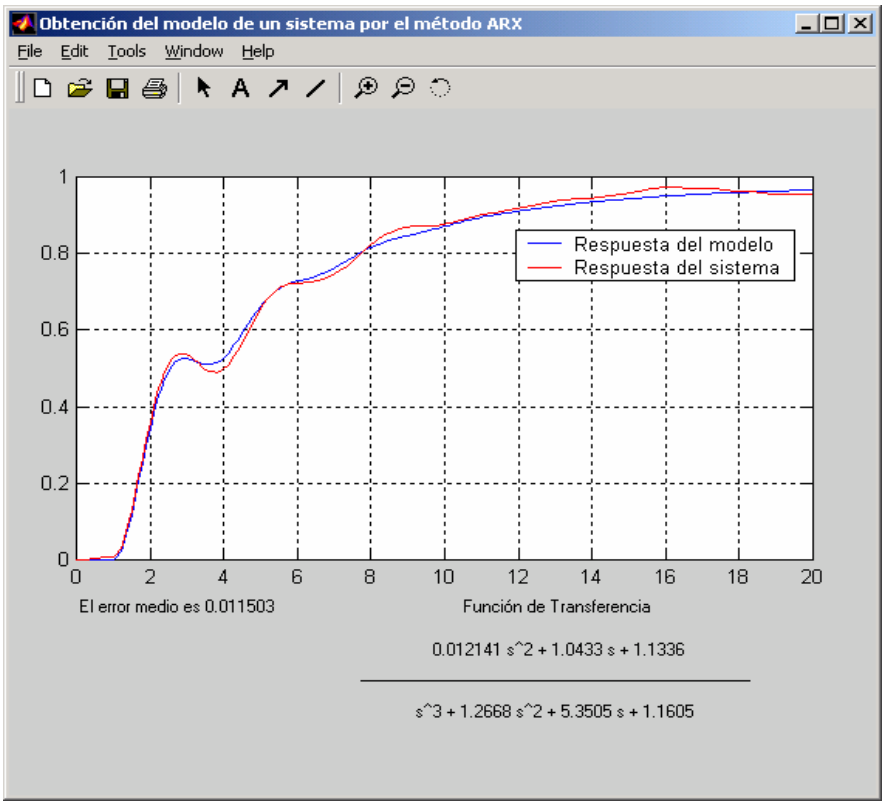


Fig. 4.6 Respuestas del modelo y del sistema

Un análisis del error medio nos diría que es realmente pequeño e igual a 0.0115.

Sin embargo pudiéramos probar con la estructura ARMAX que tiene más en cuenta el ruido.

Usaremos ahora el mismo programa arxpl cambiando th = arx (z,[3 3 1]) por th = armax (z,[3 3 3 1]).

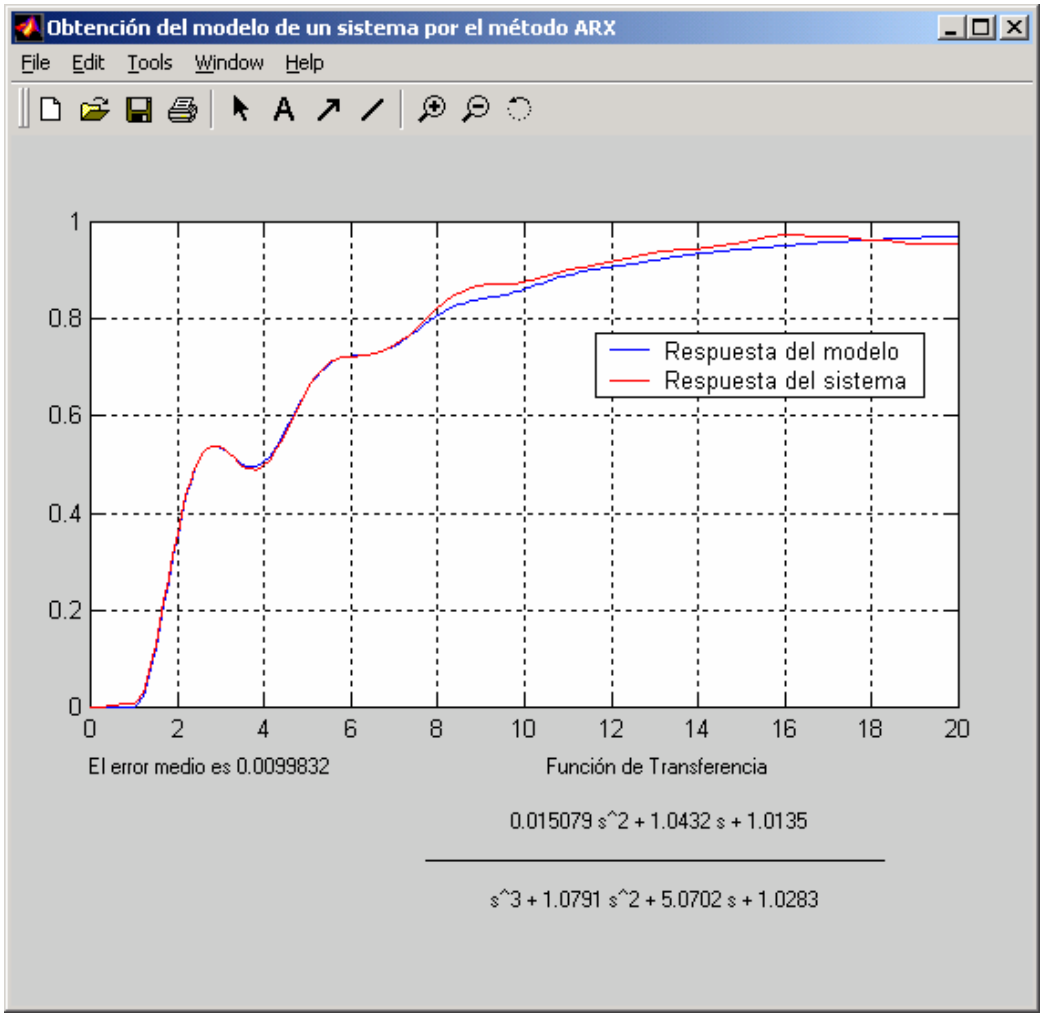


Fig. 4.7 Identificación usando la estructura armax

Ésta se ve mejor. El error medio es de 0.01.

Se pudiera probar el resto de las estructuras existentes en Matlab cambiando sólo la línea th = arx (z,[3 3 1]) del programa arxpl.

4.2 Identificación de sistemas multivariables (MIMO)

Los métodos presentados en el epígrafe anterior tienen la posibilidad de ser utilizados en sistemas multivariables. Es importante explicar cómo serían *na*, *nb* y *nk* en estos casos.
na es una matriz cuadrada cuyo orden coincide con el número de salidas.
nb y *nk* son matrices donde el número de filas corresponde con el número de salidas y el número de columnas con el número de entradas.

Mostraremos algunos ejemplos de identificación de sistemas multivariables con diferentes señales de entradas.

Ejemplo 1.1. Se trata de un sistema multivariable con dos entradas y dos salidas, descrito por las ecuaciones en el espacio de estado siguientes:

$$x' = Ax + Bu$$
$$y = Cx + Du$$

con

$$A = \begin{bmatrix} 0 & 1 \\ -25 & -4 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Este sistema, en el dominio s, se puede representar de la forma siguiente:

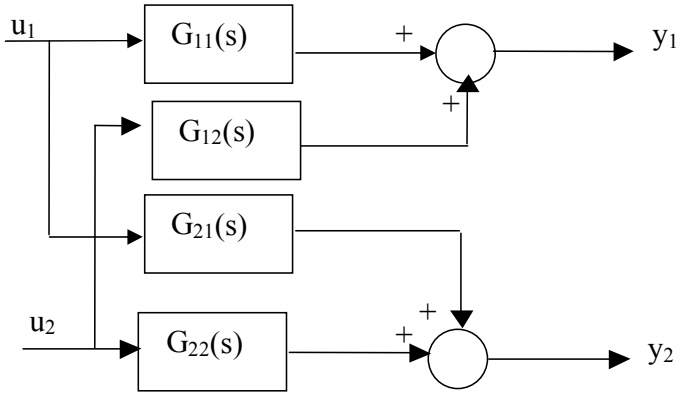


Fig. 4.8 Diagrama de bloques del sistema

Donde:

$$G_{11}(s) = \frac{s+4}{s^2+4s+25} \qquad G_{12}(s) = \frac{s+5}{s^2+4s+25}$$

$$G_{21}(s) = \frac{-25}{s^2+4s+25} \qquad G_{22}(s) = \frac{s-25}{s^2+4s+25}$$

Se trata, en particular, de un sistema de fase no mínima.

Se realizó la simulación con ayuda del *Simulink* del Matlab según el siguiente esquema para obtener los datos necesarios para la identificación:

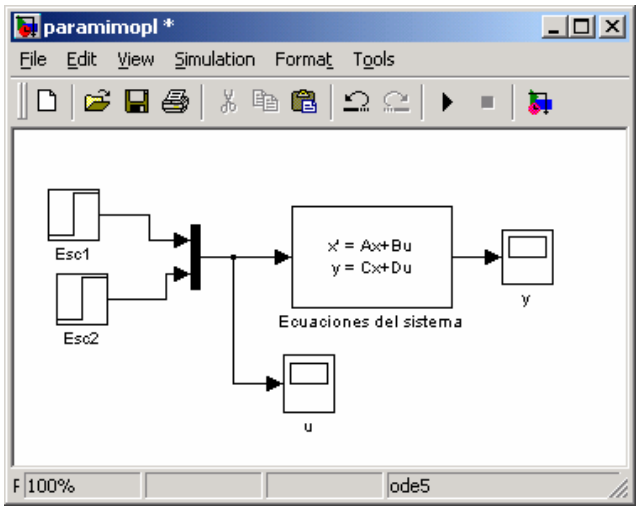


Fig. 4.9 Esquema de simulación

Los parámetros de simulación fueron los siguientes:

Tiempo total	10 u
Tiempo de muestreo	0.02 u
Método de solución	Ode5 (Dormand-Prince)
u_1 y u_2	Escalón unitario

Se realizó la identificación con el programa mimopllsim (Anexo # 8) y se obtuvieron lo siguientes resultados:

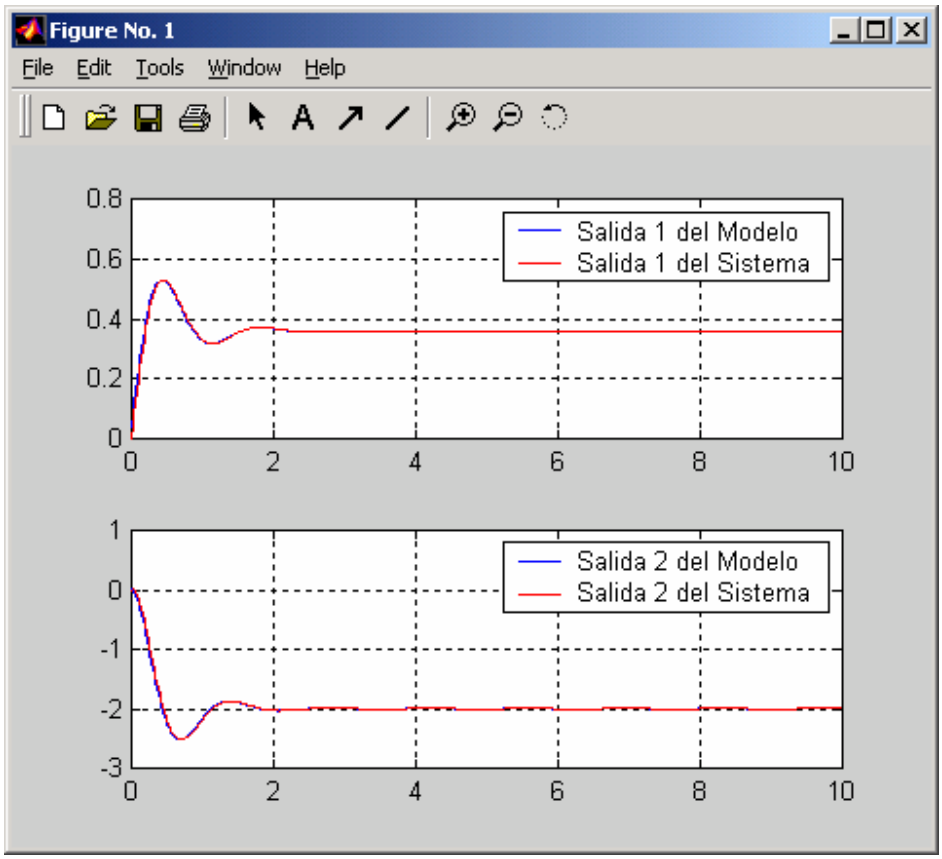


Fig. 4.10 Comparación de las salidas del modelo y del sistema

El criterio de error fue el error medio absoluto, es decir, el valor medio de los valores absolutos de las diferencias entre ambas salidas.

El error en la salida 1 fue de 0.0016231 u.

El error en la salida 2 fue de 0.0067574 u.

Se obtuvo la matriz A del modelo:

$$A = \begin{bmatrix} -13.7720 & 16.0693 \\ -9.9307 & 9.7720 \end{bmatrix}$$

diferente a la del sistema, sin embargo, ambas matrices tienen iguales valores propios hasta la cuarta cifra decimal.

Ejemplo 1.2. Mostraremos también los resultados del programa utilizando el mismo sistema con entrada ruido blanco de banda limitada (Band-Limited White Noise).

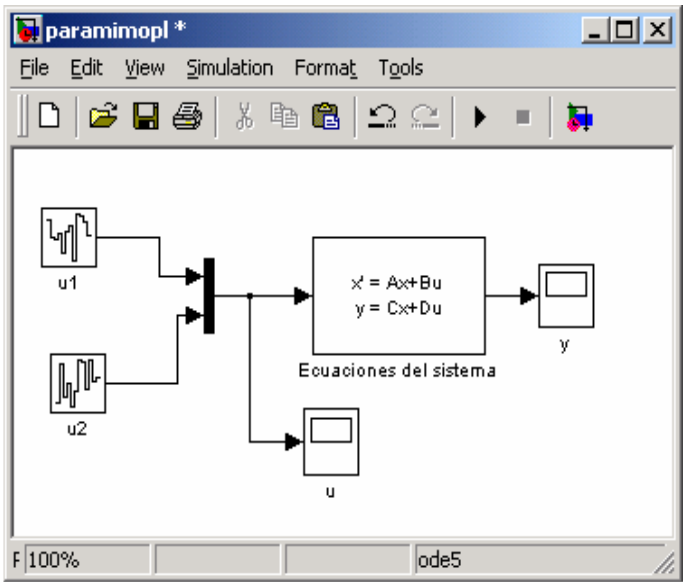


Fig. 4.11 Esquema del mismo sistema con entrada ruido

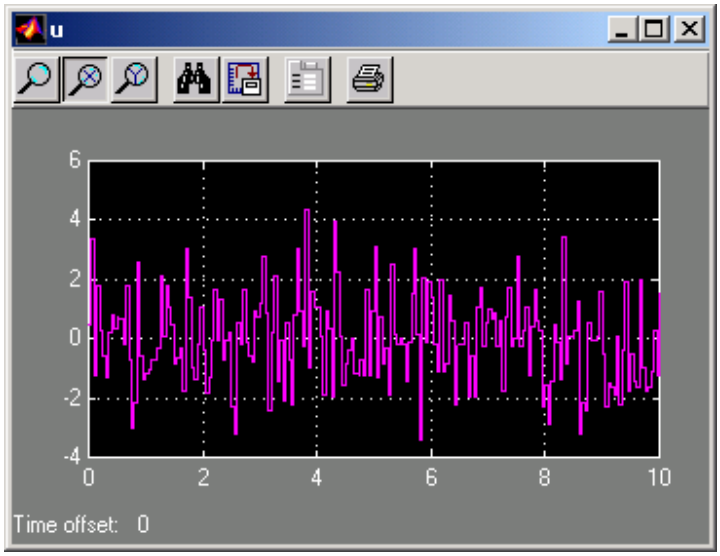


Fig. 4.12 Señal utilizada para las entradas u_1 y u_2

Los parámetros de simulación fueron los siguientes:

Tiempo total	10 u
Tiempo de muestreo	0.1 u
Método de solución	Ode5 (Dormand-Prince)

Se utilizó la misma señal de ruido blanco de banda limitada para ambas entradas simultáneamente con los siguientes parámetros:

Potencia del ruido	1
Tiempo de muestreo	0.1
Semilla (*seed*)	23341

Se realizó la identificación con el mismo programa mimopllsim y se obtuvieron los siguientes resultados:

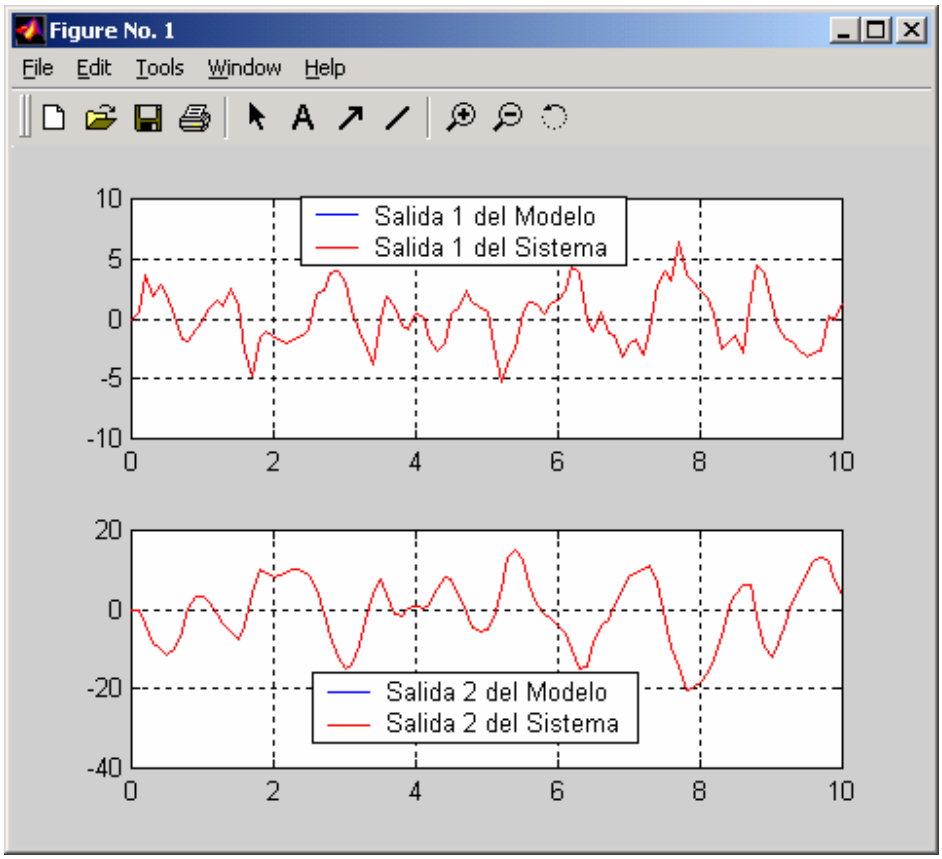


Fig. 4.13 Comparación de las salidas del modelo y del sistema

El error en la salida 1 fue de 4.1934e-015 u.

El error en la salida 2 fue de 1.758e-014 u

Se obtuvo la matriz A del modelo:

$$A = \begin{bmatrix} -9.1786 & 21.6521 \\ -3.3499 & 5.1787 \end{bmatrix}$$

pero con los mismos valores propios que la matriz del sistema hasta la cuarta cifra decimal.

Ejemplo 2. Se presenta otro sistema de dos entradas y dos salidas. Para la identificación de este sistema se elaboró el programa mimopllsim22 (Anexo # 9).

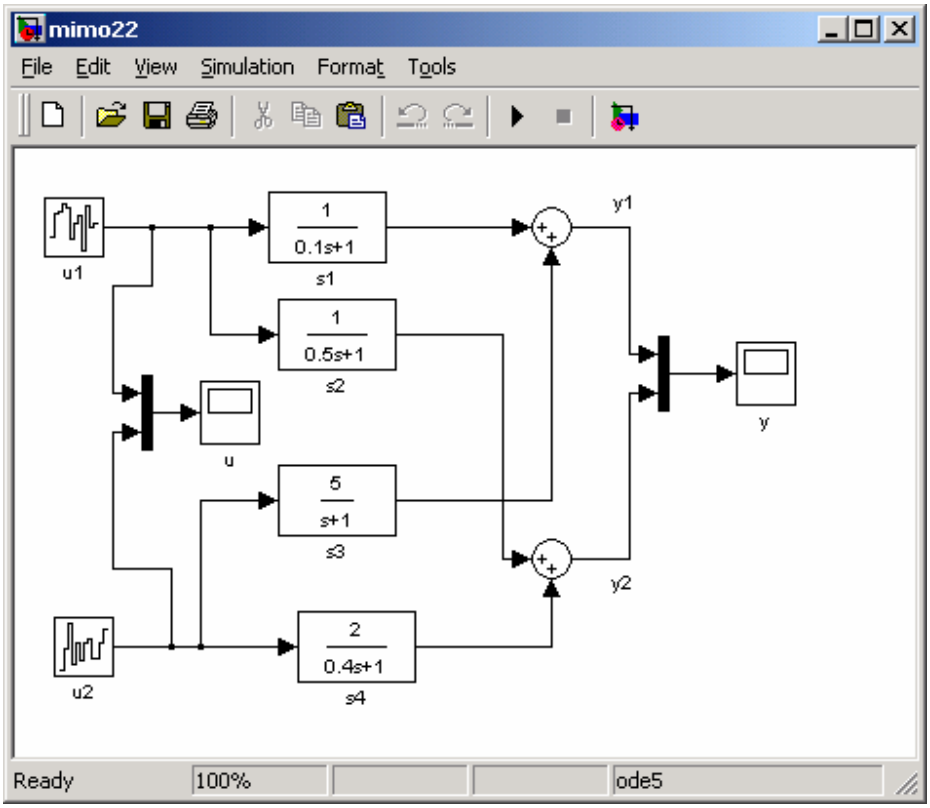


Fig. 4.14 Esquema del sistema

Los parámetros de simulación fueron los siguientes:

Tiempo total	10 u
Tiempo de muestreo	0.1 u
Método de solución	Ode5 (Dormand-Prince)

Se utilizó la misma señal de ruido blanco de banda limitada para ambas entradas simultáneamente con los siguientes parámetros:

Potencia del ruido	1
Tiempo de muestreo	0.1
Semilla (*seed*)	23341

Se realizó la identificación con el mismo programa mimopllsim, y se obtuvieron los siguientes resultados:

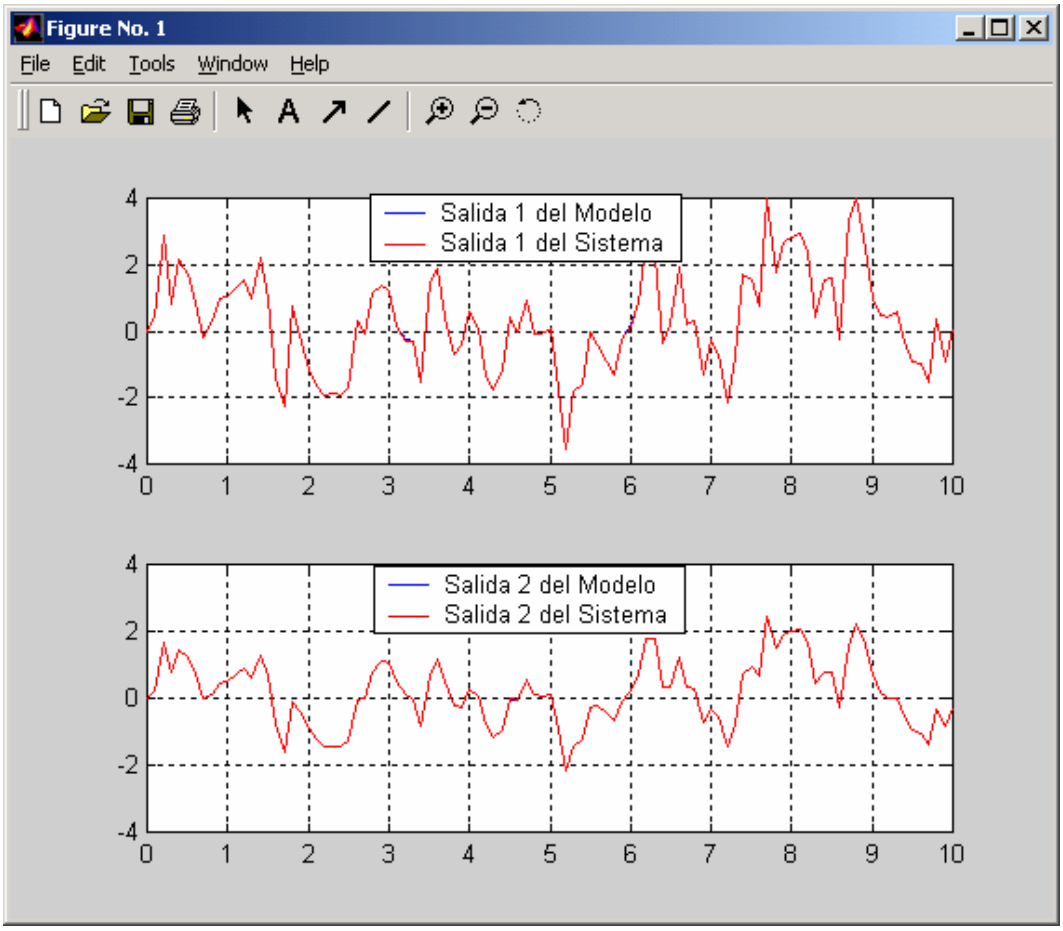


Fig. 4.15 Comparación de las salidas del modelo y el sistema

El error para la salida 1 es 0.00025017

El error para la salida 2 es 0.0002736

Las matrices del espacio de estado del modelo obtenido pueden conocerse si se teclea A,B,C,D en la ventana de comandos, luego de haberse ejecutado el programa.

Ejemplo 3. Se presenta ahora un sistema de tres entradas y dos salidas. Se utilizará el mismo programa que se usó para el ejemplo anterior, pero modificando nb y nk, pues ahora son matrices de orden 2x3.

El sistema utilizado fue el siguiente:

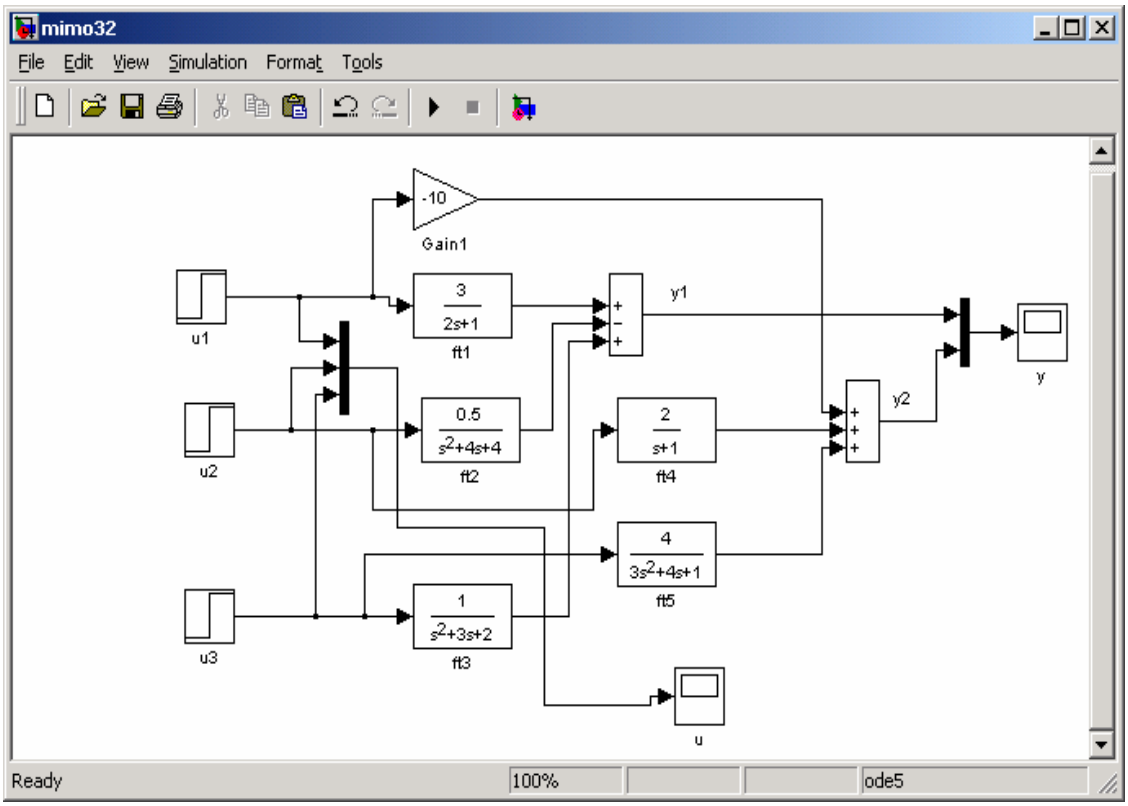


Fig. 4.16 Esquema en *Simulink* del sistema utilizado

Los parámetros de simulación fueron los siguientes:

Tiempo total	10 u
Tiempo de muestreo	0.1 u
Método de solución	Ode5 (Dormand-Prince)

Se utilizó la misma señal de ruido blanco de banda limitada para ambas entradas simultáneamente con los siguientes parámetros:

Potencia del ruido	1
Tiempo de muestreo	0.1
Semilla (*seed*)	23341

Se realizó la identificación con el mismo programa mimopllsim, y se obtuvieron los siguientes resultados:

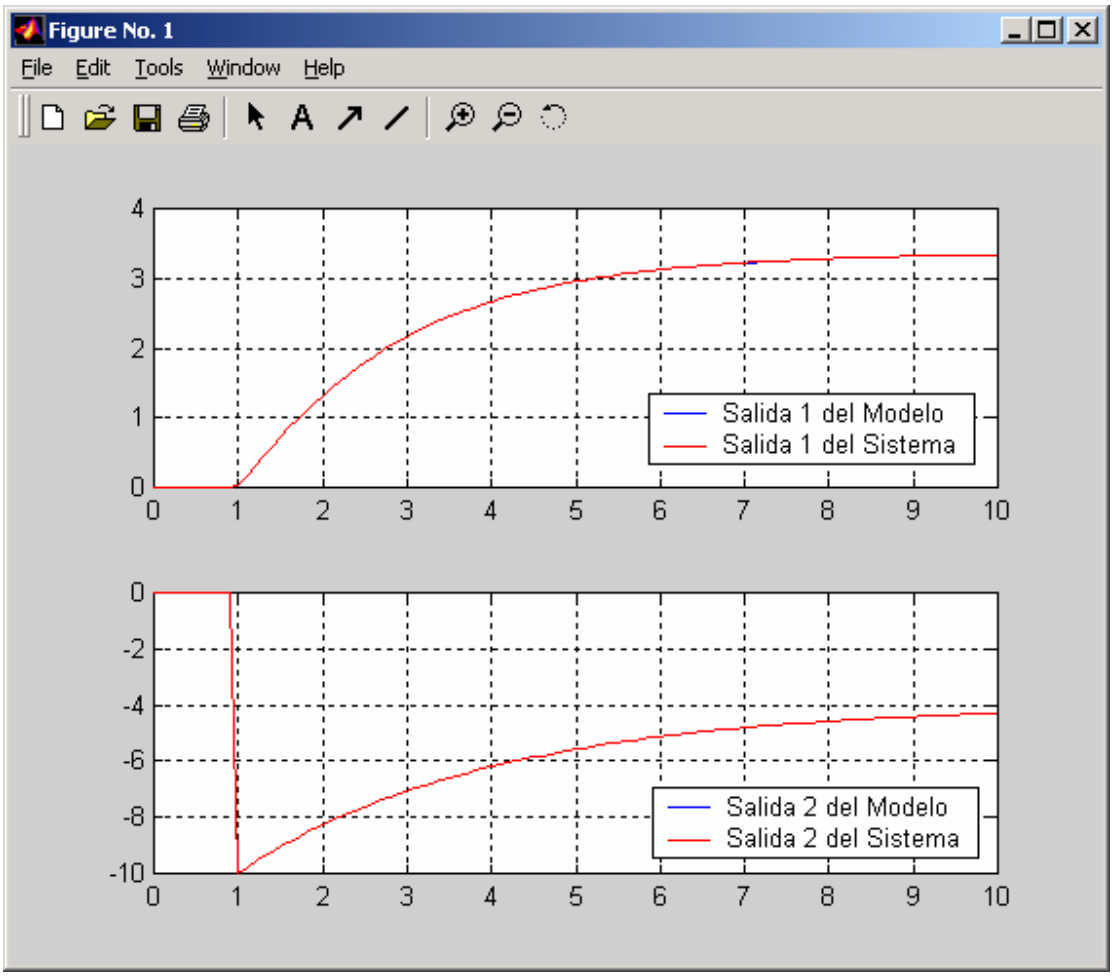


Fig. 4.17 Comparación de las salidas del modelo y el sistema

El error entre las salidas 1 del sistema y el modelo es 0.0007237

El error entre las salidas 2 del sistema y el modelo es 1.5415e-013

De igual forma las matrices del espacio de estado del modelo obtenido pueden conocerse si se teclea A,B,C,D en la ventana de comandos, luego de haberse ejecutado el programa.

Ejemplo 4. Se presenta ahora un sistema de tres entradas y tres salidas como se muestra. Se utilizará el mismo programa que se usó para el ejemplo anterior, pero modificando nb y nk, pues ahora son matrices de orden 3x3 y aumentando la gráfica de la salida tres.

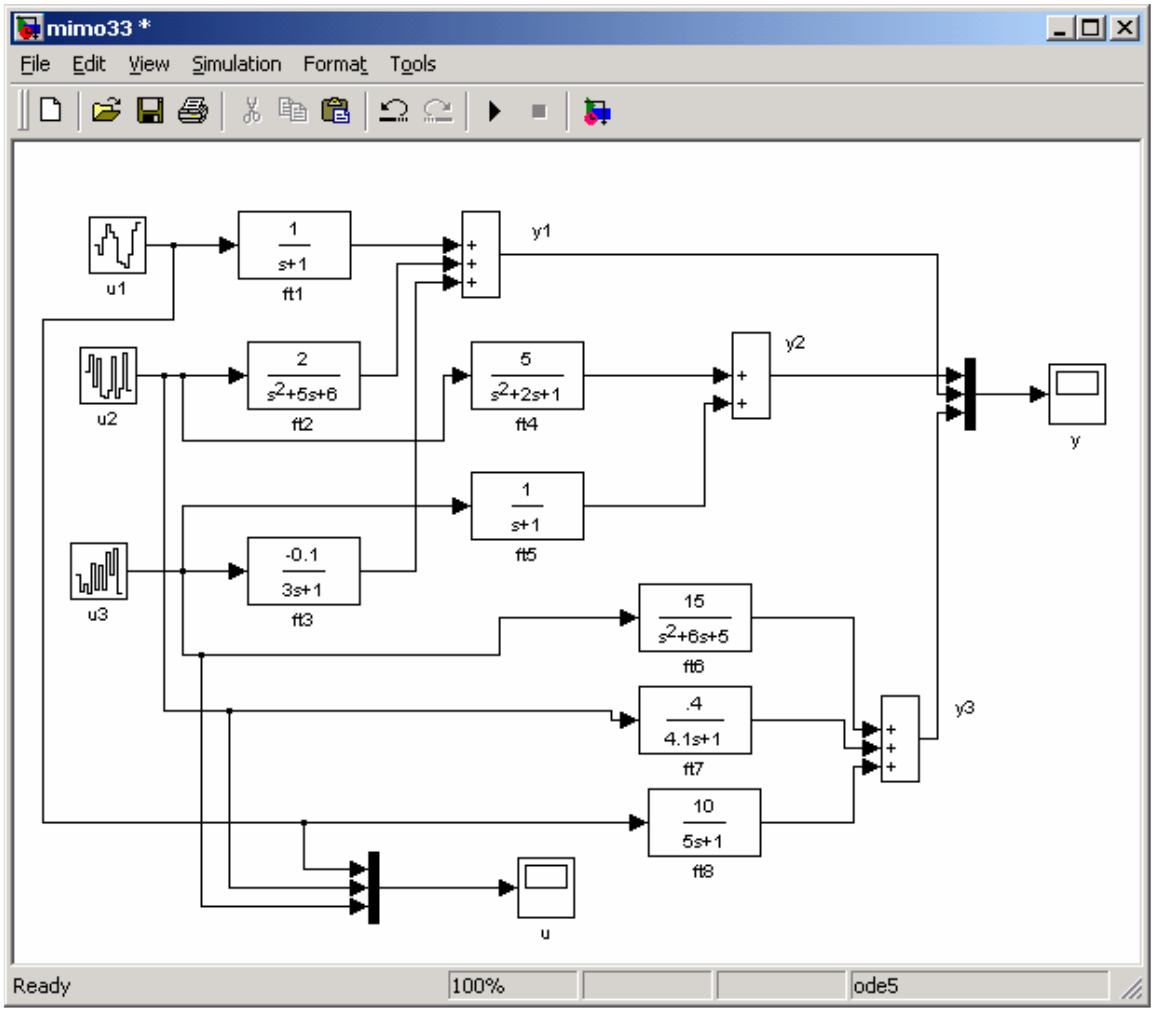


Fig. 4.18 Esquema *Simulink* del sistema de tres salidas y tres entradas

Los parámetros de simulación fueron los siguientes:

Tiempo total	35 u
Tiempo de muestreo	0.1 u
Método de solución	Ode5 (Dormand-Prince)

Se utilizó la misma señal de ruido blanco de banda limitada para ambas entradas simultáneamente con los siguientes parámetros:

Potencia del ruido	1
Tiempo de muestreo	0.1
Semilla (*seed*)	23341

Se realizó la identificación con el mismo programa mimopllsim, y se obtuvieron los siguientes resultados:

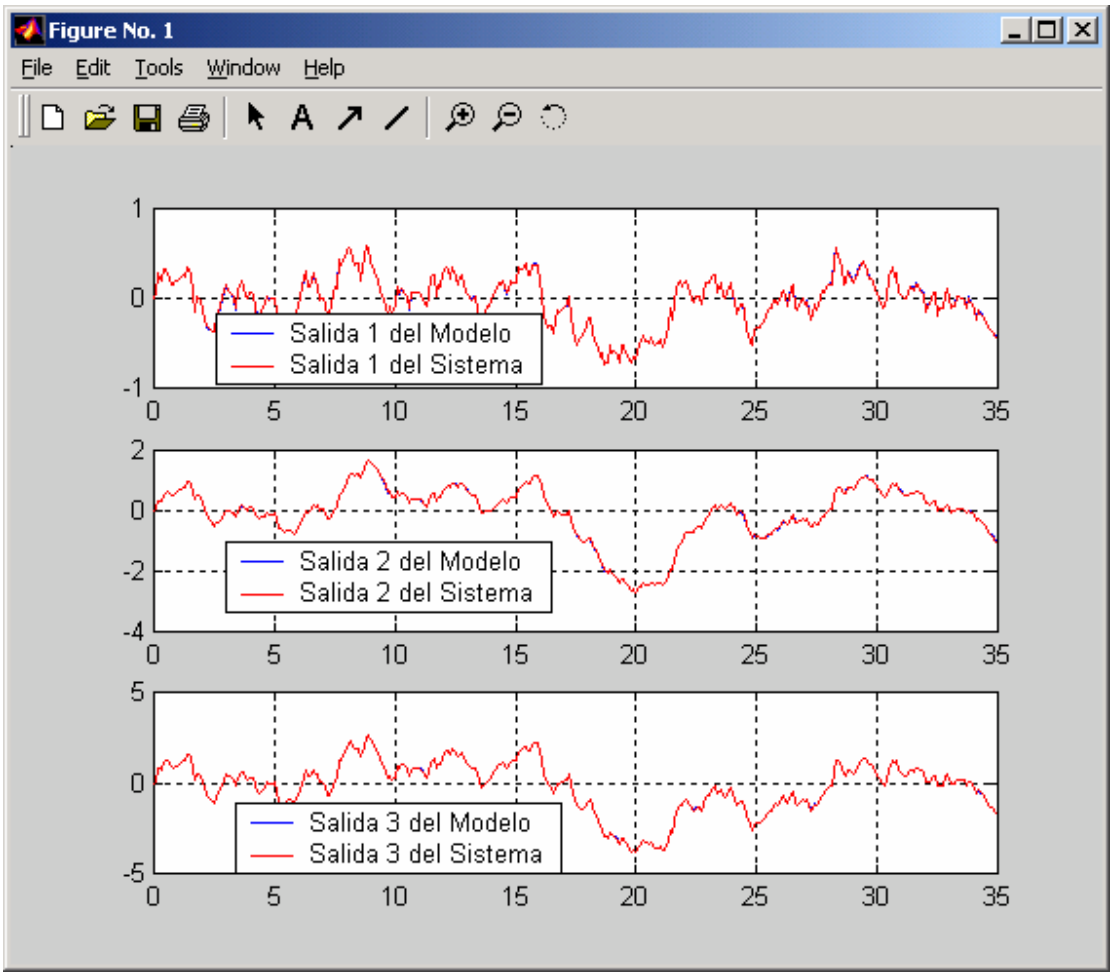


Fig. 4.19 Comparación de las salidas del modelo y el sistema

El error entre la salida 1 del sistema y el modelo es 0.0021686.

El error entre la salida 2 del sistema y el modelo es 0.0061046.

El error entre la salida 3 del sistema y el modelo es 0.0032803.

De igual forma las matrices del espacio de estado del modelo obtenido pueden conocerse si se teclea A,B,C,D en la ventana de comandos, luego de haberse ejecutado el programa.

5. Identificación de Sistemas mediante Redes Neuronales Artificiales.

La utilización de la Red Neuronal (RN) ha venido a ser, en los últimos años, un campo de gran desarrollo y rápido crecimiento en aplicaciones de la Inteligencia artificial (IA). La RN se ha empleado principalmente para detectar los atributos de objetos y sus relaciones, y es posible encontrar una amplia bibliografía al respecto.

En este capítulo vamos a centrarnos en ver una descripción del uso de la RN como un método de identificación para sistemas dinámicos lineales y no lineales.

La principal ventaja está en la explotación de su uso en sistemas no lineales que hasta ahora no han podido ser resueltos por tener una solución muy compleja, ya sea por la dificultad de implementar algoritmos complicados o por ser sistemas con parámetros desconocidos.

Los procedimientos clásicos de identificación de sistemas que utilizan los datos históricos de la planta, están basados en asumir como verdaderas varias hipótesis respecto al sistema. Dentro de estas hipótesis se pueden resaltar la linealidad, ser invariantes en el tiempo, conocer el orden del sistema (ARMAX), las constantes de tiempo del sistema en respuesta a un escalón (FIR), etc.; condiciones asumidas localmente que luego son extrapoladas globalmente a todas las condiciones de operación.

Los requerimientos para una RN son menores en comparación con los procedimientos clásicos de identificación. Comparando ambos procedimientos desde el punto de vista de entrada y estructura, observamos que el usuario debe especificar los datos para el proceso de entrada-salida idénticamente en ambos tipos de procedimientos.

En cuanto a la estructura, en los procedimientos clásicos el usuario debe especificar la característica de la estructura de la relación entre las entradas y las salidas, a veces muy complejas, mientras que para la identificación usando las RN sólo se debe especificar la topología de la red.

La RN no intenta asumir ninguna estructura de la relación entrada-salida, la red crea la

relación de las entradas a las salidas basándose en los ejemplos proporcionados durante el entrenamiento. Lo que se le pide a la RN en este caso, es que internamente consiga un modelo capaz de reproducir unos valores numéricos que se asemejen a los valores de algún sistema físico.

La RN basada en el perceptron multicapa, con al menos una capa oculta y una función de activación sigmoidal puede aproximar una gran diversidad de funciones no lineales. Los parámetros de la RN son ajustados para modelar el comportamiento de una función dada, semejante a cualquier proceso iterativo de estimación de parámetros.

La capacidad de aproximar funciones y la habilidad de aprender la relación funcional entre las variables, son dos de las más importantes propiedades de la RN que son muy útiles para la identificación de sistemas. En la práctica, con la utilización de la RN se han logrado buenos resultados y un rápido aprendizaje en aplicaciones de Reconocimiento de Patrones (clasificación), pero en aplicaciones de representación de datos la RN es más sensible y a menudo requiere mayor cantidad de datos y tiempo para el entrenamiento.

Desafortunadamente, el problema de la identificación de parámetros y su generalización es un problema sobredimensionado, ya que un conjunto finito de ejemplos de una función son consistentes o pueden coincidir con los puntos de un número infinito de otras funciones, muchas de las cuales no tengan nada que ver con el problema original.

Si la red es capaz de descubrir una función que se comporta correctamente como el conjunto de ejemplos, se debe comprobar, además, si es capaz de generalizar la solución ante entradas no conocidas. Si la red logra una buena generalización del problema, es decir, que si para esos valores de los parámetros se obtiene un error cuadrático (entre los valores de la red y los valores deseados) mínimo y aceptable, entonces será una indicación de que ha descubierto una importante estructura subyacente que contribuirá a la comprensión y resolución del problema.

5.1 Introducción a las Redes Neuronales.

El cerebro humano hace uso de dos métodos diferentes de procesamiento de información. Uno es el procesamiento inferencial secuencial y lógico en términos de símbolos. El otro es el procesamiento paralelo, el cual es realizado por las dinámicas de elementos de proceso interactuando mutuamente. Este método usa una representación de la información distribuida con habilidad de aprendizaje. La Neurocomputación es uno de los métodos más importantes de procesamiento de información que usa las interacciones dinámicas paralelas de estos elementos de proceso modificables.

Esta disciplina tecnológica trata de sistemas de procesamiento de información paralelos, distribuidos y adaptables, que desarrollan capacidad de proceso, en respuesta a la exposición a un entorno de información. Los sistemas más empleados en *Neurocomputación* son las Redes Neuronales Artificiales (*Artificial Neural Network*s), las cuales desarrollan transformaciones entre objetos.

Se puede definir una Red Neuronal Artificial (RNA) como una estructura de proceso de información paralela y distribuida, formada por Elementos de Proceso (EP) o neuronas, interconectados a través de canales unidireccionales llamados conexiones y que tiene asociadas reglas de aprendizaje. Cada EP tiene una salida única que se distribuye sobre un número de conexiones bilaterales.

El proceso de información interno a cada EP conocido también como modelo de neurona, puede definirse de forma arbitraria, pero ha de ser totalmente local, es decir, depender solamente de los valores de las señales que llegan al EP y de valores almacenados en su memoria local. Grupos de EP pueden ser interconectados en una variedad de maneras para formar RNA. A la organización que forman los EP en la red se le conoce como topología de la red. La mayoría de las redes tienen sus EP distribuidos en conjuntos disjuntos llamados capas, cuyos EP tienen la misma función de transferencia generalmente. Muchas RNA incluyen una capa de entrada en la que cada EP recibe exactamente una entrada proveniente del exterior y simplemente la distribuyen al resto de la red. Por tanto, no tienen

función de transferencia.

Una RNA está caracterizada fundamentalmente por tres aspectos.

- La topología de la red .
- Las conexiones de los EP de procesos.
- La estrategia para los patrones de aprendizaje o entrenamiento.

Algunas topologías básicas.-

a) Red simple

Se utilizan n neuronas de entrada y se conectan con m neuronas de salida mediante caminos pesados. No existe conexión entre las neuronas. Cada neurona de salida calcula una respuesta, y la salida de la red será un vector de n componentes. Si se combina con unidades de asociación resulta una arquitectura similar a la del Perceptron.

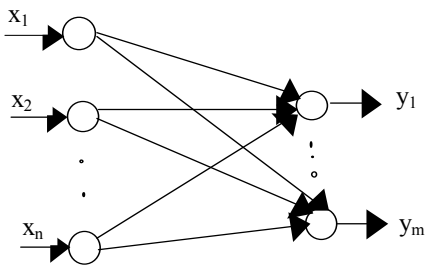


Fig 5.1 Topología de una red simple

b) Redes multicapas

Es una de las topologías más poderosas. En ella se ordena el conjunto de elementos de procesamiento (neuronas) en niveles, de modo que los enlaces se establecen desde unidades en el nivel y a unidades en el nivel j (i < j), por lo que la información fluye unidireccionalmente. Esta arquitectura se conoce como dirigida adelante (*Feedforward*). Típicamente existe una capa de unidades de entrada, una o más capas ocultas de neuronas, y una capa de salida. Esta topología es la base del Perceptron Multicapas.

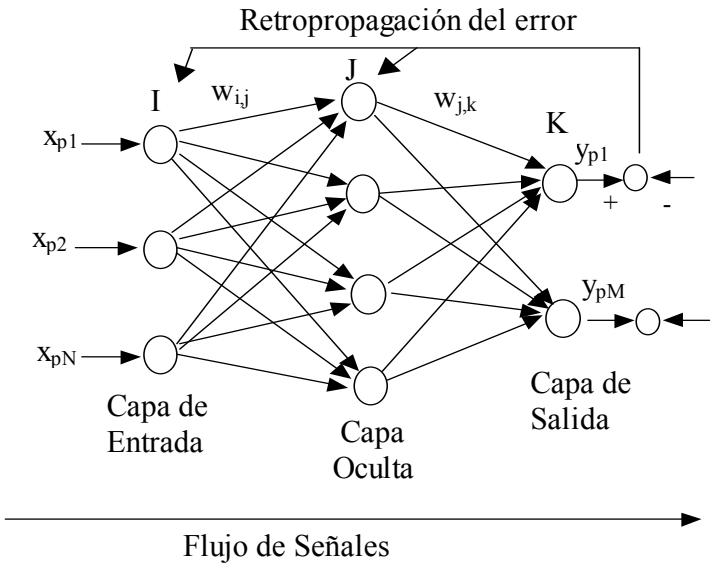


Fig. 5.2 Topología de una red multicapas

c) Topología del modelo interactivo

Se tiene un conjunto de n neuronas, las cuales se conectan completa y mutuamente, es decir, todas las unidades sirven como entrada y como salida; cada neurona se conecta a las n-1 restantes mediante caminos pesados. Esta es la base de la red de *Hopfield*.

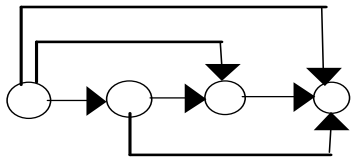


Fig 5.3 Modelo interactivo

Algunos modelos de neuronas

El modelo de la neurona define el comportamiento de la misma al recibir una entrada para producir una respuesta; a esta respuesta se le conoce como nivel de activación. Este se calcula en dos etapas: Primero se obtiene la entrada total a la neurona, la cual es la suma del producto del peso de cada enlace por el valor de la información que fluye por él, más un término que indica la predisposición de la neurona a reaccionar(sesgo).

$$h_j = \sum_i w_{ij} * x_i + \text{sesgo}_j$$

En la segunda etapa se calcula el nivel de activación utilizando una función cuyo argumento es la entrada total a la neurona (h_j). Entre las funciones más utilizadas tenemos:

a) <u>modelo lineal</u>

La activación de la neurona es igual a la entrada total.

b) <u>modelo lineal con umbral</u>

El nivel de activación toma un valor binario en dependencia del signo de la entrada total. Ejemplo de este modelo es la función signo.

c) <u>modelo continuo</u>

La salida está relacionada no linealmente con la entrada. Una de las funciones más usadas es la sigmoide.

$$s_i = \frac{1}{1 + e^{-h_i}}$$

Note que esta función transforma el valor de la entrada total a valores reales entre 0 y 1.

Otras funciones de activación son:

tangente hiperbólica: $S_i = (\exp(h_j) - \exp(h_j))/(\exp(h_j) + \exp(h_j))$, $-1 <= S_i <= 1$

gausiana: $S_i = \exp(-h_j^2/-\delta^2)$

5.2 Algoritmo de aprendizaje

El objetivo del aprendizaje es el proceso de cálculo de los pesos de las conexiones. Cinco rasgos caracterizan este paradigma:

- el algoritmo usado
- el criterio de parada
- la forma en que los datos de entrenamiento se presentan a la red
- el tipo de aprendizaje
- la regla de aprendizaje

Varios algoritmos de aprendizaje han sido usados. Entre ellos, el gradiente descendente **(**también llamado algoritmo de retropropagación de errores o *backpropagation*). Otro método usado es el del gradiente conjugado. Hay varios métodos basados en búsquedas aleatorias, y finalmente se ha propuesto un algoritmo genético.

Un criterio de mínimos cuadrados se calcula generalmente con la diferencia entre los valores de salida de la red y los valores de salida del conjunto de entrenamiento.

Min $Ep = \frac{1}{2} \sum_{i=1}^{Ns} e_i^2$ donde $e_i = d - y$, d - valor deseado

Ns - Cantidad de neuronas de salida

Min $E = \frac{1}{2} \sum_{j=1}^{Np} \sum_{i=1}^{Ns} e_{ij}^2$ Np - Cantidad de patrones

La convergencia se asume cuando el criterio de mínimos cuadrados o su razón de cambio es menor que una tolerancia prefijada o cuando el tiempo de aprendizaje ha finalizado.

Durante el aprendizaje, el conjunto de datos de entrenamiento generalmente se presenta repetidamente a la red, y los pesos son ajustados en cada iteración hasta que se obtiene una solución. Este es el procedimiento usado en las redes *feedforward*. En otros tipos de redes el conjunto de datos se puede presentar una sola vez.

Sistemas Dinámicos

Para la identificación de una planta dinámica, donde la principal característica es ser dependiente del tiempo, se asume que el sistema físico tiene una estructura matemática general expresada como una función lineal o no lineal F(y, u) de los valores actuales y de los valores pasados de sus entradas y salidas, como puede ser la descrita por la siguiente ecuación en diferencias:

$y(t) = F[y(t-1),y(t-2),...,y(t-n),u(t),u(t-1),...,u(t-m)]$

Para distinguir las arquitecturas de RN que incluyan alguna representación del tiempo las denominaremos Redes Neuronales Dinámicas. La principal diferencia es que las RN para la identificación de sistemas dinámicos utilizan un lazo de realimentación(feedback) con las salidas anteriores de la planta, considerándolas como otras entradas adicionales a la red, además de las entradas ya existentes.

La identificación de los parámetros de la planta se basará en una RN de tres capas (entrada, oculta y salida) y un bloque adicional con las señales de retardos que se añaden como entradas.

El aumento de las unidades de retardo posibilita una rápida convergencia y mejorar la precisión de los parámetros estimados por la red. Ahora bien en este punto se debe llegar a un compromiso ya que ampliar la red implica un mayor número de operaciones y por tanto se necesitará una mayor capacidad de cálculo.

Los dos esquemas tradicionales de identificación para sistemas dinámicos, llamados serie–paralelo y paralelo, son representados en las figuras 5.4 y 5.5 respectivamente.

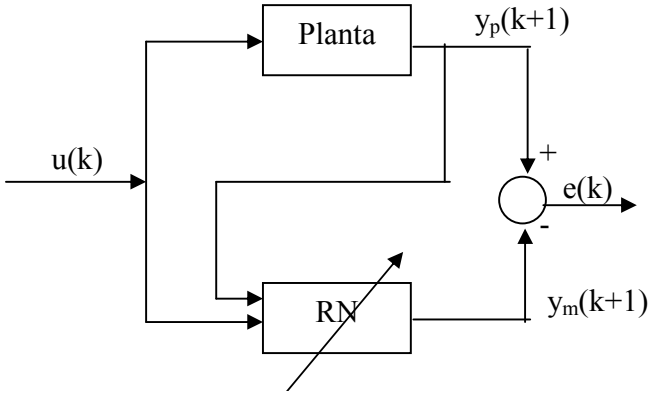


Fig. 5.4 Modelo Serie - Paralelo

Este esquema es también denominado *teacher forcing* y puede ser usado con cualquier algoritmo de aprendizaje. Es muy útil al inicio del aprendizaje para que los valores de la red no se alejen mucho y converja más rápido.

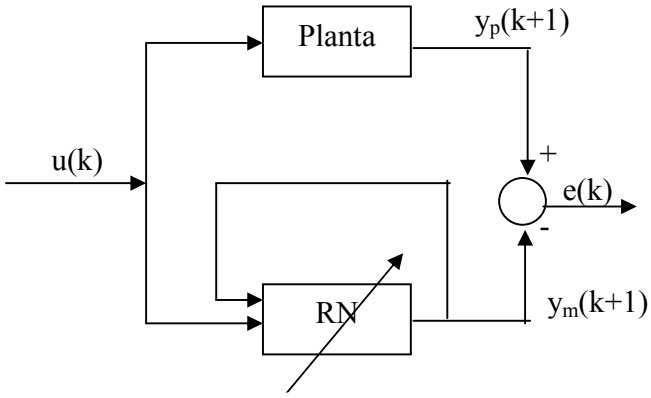


Fig. 5.5 Modelo Paralelo

5.3 Identificación con Redes Neuronales en el ambiente de Matlab.

Para la identificación de sistemas utilizando Redes Neuronales en ambiente Matlab es necesario disponer de un conjunto de datos que deben ser obtenidos por vía experimental o mediante algún modelo de *Simulink*. Aquí se utilizará esta última vía para demostrar la metodología de trabajo.

Primero se utiliza el programa que genera los datos de salida de un sistema ante una entrada de tipo seudo aleatoria. El programa se lista a continuación:

```
% Programa GeneraDatos1
% permite generar los datos que luego serán empleados
% en el entrenamiento de la red.
ts = input('Tiempo de muestreo = ');
tstop = 10 % Tiempo final de simulación
DatosEntr1
% La planta está representada por un sistema de primer orden
% y se generan la entrada, la entrada retrasada un paso,  la
% salida y la salida retrasada un paso.
sim('DatosEntr1')
% Se simula el modelo Simulink DatosEntr1
```

Se seleccionó para este ejemplo un tiempo de muestreo ts = 0.1 u

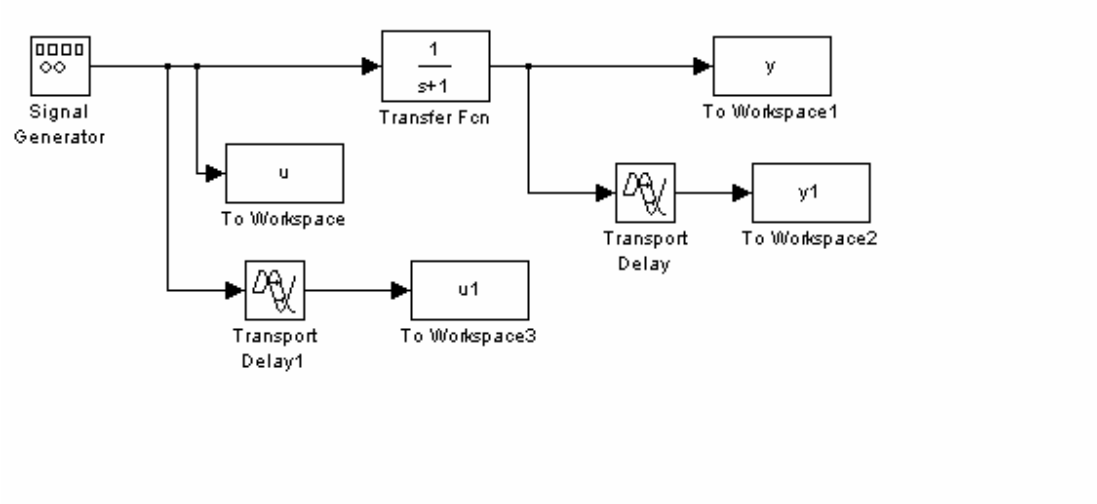


Fig. 5.6 Esquema *Simulink* para generar los datos del ejemplo 1

Se usa para generar los datos un generador de señales con una señal aleatoria de amplitud 3 y frecuencia de 1 Hz. Los retrasos de transporte utilizan el valor de ts como tiempo de retraso (*time delay*).

Una vez generados los datos se puede utilizar el siguiente programa en Matlab para entrenar la Red Neuronal.

```
% Programa EntrenaRed1
% Genera una red feedforward con 3 entradas, 20 neuronas en la capa
% oculta y una neurona de salida. Las neuronas ocultas trabajan con
% una función de activación tansig y la de salida con purelin. El
% método de entrenamiento empleado es el Levenberg -  Marquart.
% ts tiempo de muestreo.
red = newff([-2 2;-2 2;-1 1],[15,1],{'tansig','purelin'},'trainlm');
% Se usa la entrada, la entrada retrasada un paso y la salida retrasada
% un paso. Se pueden obtener con GeneraDatos1
p = [u'; u1';y1'];
% Definición de algunos parámetros de aprendizaje
red.trainParam.show = 50; % Mostrar cada 50 iteraciones
red.trainParam.goal = 0.0001; % Error de salida
red.trainParam.epochs = 100; % Total de épocas
% Entrenar la Red Neuronal.
red = train(red,p,y');
yred = sim(red,p); % Se simula la red entrenada
figure(1)
```

```
plot(yred)
hold
plot(y','red')
gensim(red,-1)
% Genera el bloque Simulink que representa la Red Neuronal
% entrenada y con el -1 se indica que se puede modificar ts.
```

Con el siguiente esquema *Simulink* (Fig. 5.7) se verifica el trabajo de la Red Neuronal entrenada ante diferentes señales de entrada y se puede observar si la misma identifica correctamente o no el sistema bajo estudio.

En este esquema es preciso indicar que en el bloque para obtener la señal escalón se toman como parámetros el valor final igual a 0.6 y el tiempo de muestreo ts. Además el bloque que corresponde a la Red Neuronal es el que se obtuvo con el programa EntrenaRed1. De igual forma es preciso señalar que los parámetros para la simulación deben ser los siguientes: tiempo de parada (tstop), tamaño del paso (ts) y se recomienda que se utilice el mismo método de solución, que en este caso fue Runge-Kutta 4to. Orden.

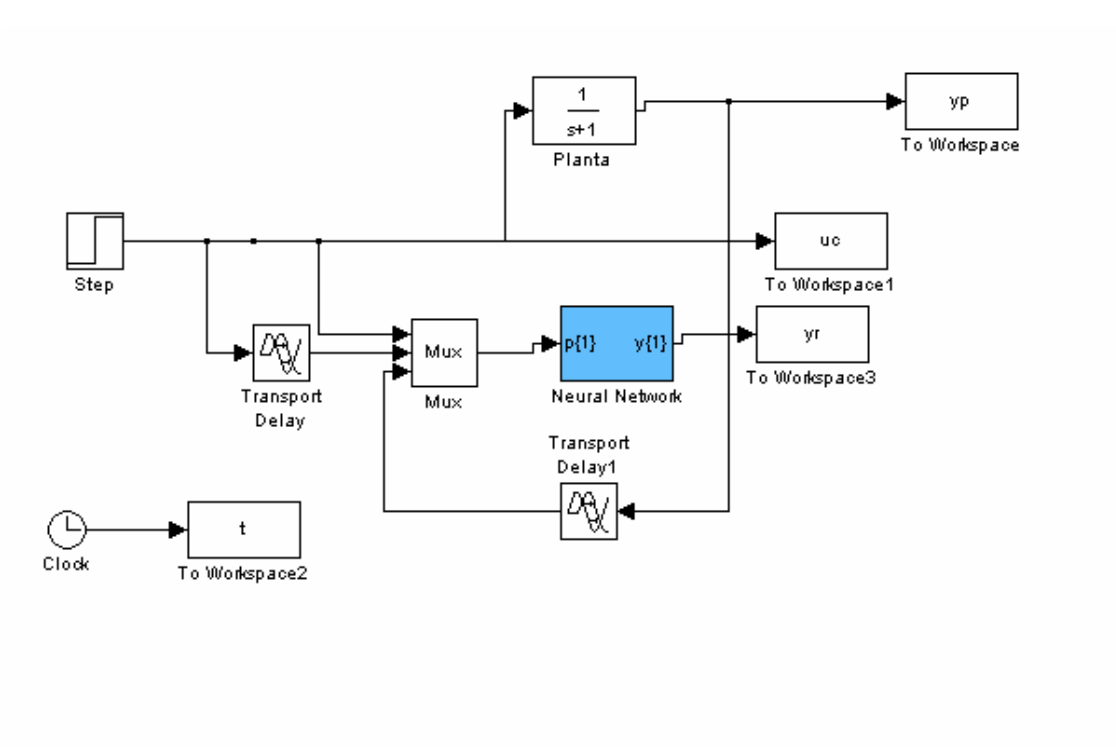


Fig. 5.7 Esquema *Simulink* para verificar el modelo de Red Neuronal

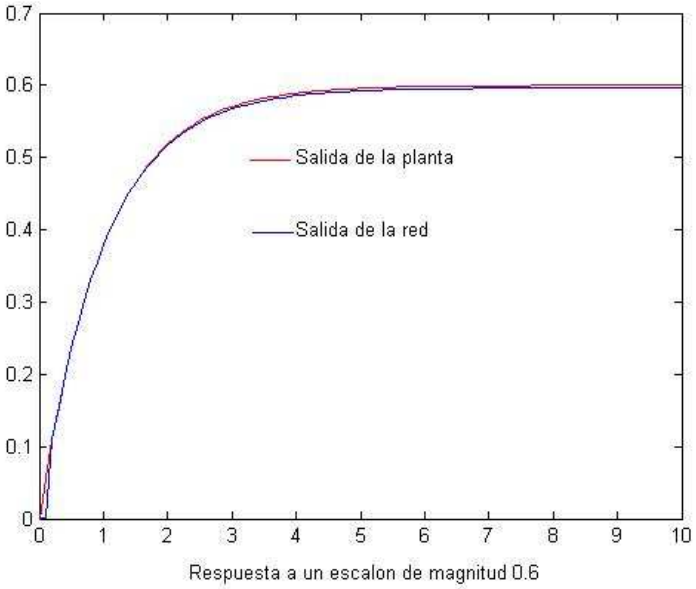


Fig. 5.8 Respuesta de la planta y la red ante un escalón

6. Algunas recomendaciones para la identificación

Algunas recomendaciones útiles para una estrategia de identificación pueden ser:

- En la etapa de análisis debe tenerse en cuenta que aunque el sistema sea no lineal, puede ser útil adoptar un modelo lineal con el objetivo de estudiar su comportamiento ante variaciones relativamente pequeñas sobre un punto de trabajo.

- Pueden utilizarse hipótesis simplificadoras para describir el comportamiento de un sistema mediante un modelo de orden reducido más fácil de identificar y luego de utilizar.

- En sistemas lineales con múltiples entradas, es posible aplicar el principio de superposición, considerando cada salida como suma de salidas elementales correspondientes a una sola entrada.

- Ante la disyuntiva de identificar un modelo de una variable o multivariable, puede influir el hecho de que el modelo sea de parámetros constantes o variables; para identificar un modelo de parámetros constantes puede ser necesario considerar distintas perturbaciones como entradas suplementarias.

- En el análisis debe tenerse en cuenta la determinación del tiempo de duración de las experiencias, porque pueden existir parámetros que varíen en función de perturbaciones lentas no medibles, o aparecer no linealidades que no están presentes en un transitorio alrededor de un punto de trabajo.

- Dependiendo de los resultados obtenidos en la etapa experimental, puede ser necesario volver a considerar la etapa del análisis para modificar la hipótesis de partida.

- Debe analizarse si el sistema cuyo modelo será identificado, trabajará a lazo cerrado o a lazo abierto, caso este que requiere mayor precisión.

- Una alternativa de la etapa de análisis es imponer el orden del sistema, otra es dejarlo libre a determinar según las medidas.

- Es importante considerar el ruido presente. Ante señales muy ruidosas es recomendable filtrar la información que se va a procesar, para que los datos analizados reflejen de forma más exacta el comportamiento real del sistema.

- Al enfrentar un problema de complejidad evidente, pudiera ser conveniente trabajar inicialmente con un modelo simplificado que aporte una idea general de la solución y finalmente con un modelo más complejo que aporte resultados más completos.

- Es importante validar la concordancia entre los resultados del análisis del modelo matemático obtenido, y los resultados del estudio experimental del sistema físico.

- Debe tenerse presente que un modelo lineal de parámetros concentrados, puede ser válido operando a baja frecuencia y no serlo a frecuencias altas, ya que las propiedades despreciadas de los parámetros distribuidos pueden volverse un factor importante en el comportamiento dinámico del sistema.

- Si se dispone de suficientes mediciones confiables del sistema a identificar, puede utilizarse una parte para la obtención del modelo y el resto para la validación del mismo.

- La caracterización de las señales (entrada, estado, salida) de un sistema dinámico, tomadas directamente de datos de registro del sistema en operación normal, permite la adición de información fenomenológica sobre el sistema para compensar las regiones de información escasa, a fin de identificar un modelo de un tipo determinado, que actúe como modelo global y robusto del sistema en todo el espacio de estado posible.

- Utilizar un escalón como señal de prueba para la identificación de sistemas tiene como ventaja la sencillez de la generación de dicha función, y como desventaja que introduce una alteración relativamente grande en el comportamiento del sistema; en procesos industriales esto no siempre es permisible.

- La utilización de un pulso como señal de prueba en la identificación de sistemas permite tiempos de experimentación cortos, en consecuencia, exige mayor exactitud de las mediciones a realizar.

- En el Método de pulso para obtener la respuesta a escalón, los errores en la determinación de la respuesta a un pulso se van acumulando, lo que exige medios de medición precisos.

- Las señales de prueba escalón y pulso sólo pueden lograrse en la práctica de forma aproximada, a los efectos de identificar un sistema, esta aproximación es adecuada, si la constante de tiempo de dichas señales es menor que la décima parte de la menor constante de tiempo que se requiere determinar en la identificación.

- Cuando se va a identificar un sistema utilizando señales sinusoidales, es necesario determinar adecuadamente la gama de frecuencias y la amplitud de las señales, la aplicación de varias señales de diferentes frecuencias hace que el tiempo de experimentación sea mucho mayor.

- Es recomendable trazar las gráficas de los datos de que se dispone para la identificación y analizarlas cuidadosamente; es posible que se trate de una respuesta a escalón y se podrá inferir el orden y la ganancia; es posible observar no linealidades, por ejemplo, diferentes respuestas a diferentes niveles; si se observa que todo el intervalo de trabajo no contiene información, se puede escoger sólo una parte de éste; permite detectar la presencia de los llamados puntos locos para su posterior eliminación.

- La aplicación del Método de integración gradual a sistemas de 1ro., 2do. y 3er. órdenes permite obtener resultados satisfactorios, obteniéndose errores promedio muy pequeños; si las señales disponibles son muy ruidosas se recomienda darle tratamiento preliminar o filtrado, de modo que reflejen más exactamente el comportamiento real del sistema.

- A partir de un modelo paramétrico es inmediato obtener las curvas de respuesta de frecuencia, y un modelo no paramétrico puede parametrizarse empleando coeficientes tales como márgenes de fase y de ganancia, ancho de banda, etc.

Anexo # 1

```
%Método de Oldenbourg Sartorius
%Calcula la pendiente y el punto de la tangente
%a la respuesta
%Deben tomarse del workspace la salida,
%la entrada y el tiempo
%La señal de salida debe almacenarse en ys
%El tiempo en t
%La señal de entrada en us
ys=y.signals.values;
t=y.time;
us=u.signals.values;
tm=t(2)-t(1);%Intervalo de muestreo
long=length(t);
tf1=t(long);%valor final de t
i=0;
tp=-tm;
p=1;
while  p>0
   tp=tp+tm;
   i=i+1;
   pendi(i)=ys(i+1)-ys(i);
   pendi(i+1)=ys(i+3)-ys(i+2);
   p=pendi(i+1)-pendi(i);
   %disp(p);
end
ti=tp;%Éste es el valor de t donde ocurre el punto de inflexión
disp(['El punto de inflexión ocurre en t= ',num2str(ti)]);
%pause
tc=0:tm:tp;
longtc=length(tc);
yp=ys(longtc);
pend=(ys(longtc)-ys(longtc-1))/tm;
%Determinaremos ahora Tc
Tc1=(ys(long)-yp+pend*ti)/pend;
Tc=Tc1-ti;
%Cálculo de Ta
Ta0=(ys(1)-yp+pend*ti)/pend;
Ta=Tc1-Ta0;
%Creación de la función T2/TA=F(T1/TA)
T1TA=0:0.1:1;
T2TA=[1 0.73 0.57 0.44 0.34 0.25 0.18 0.12 0.07 0.03 0];
f=polyfit(T1TA,T2TA,6);%Se crea un polinomio de grado 6
Tg=[];
for i=0:0.01:1,
   Tg= [Tg polyval(f,i)];
end
abc=0:0.01:1;
a=0;
j=0;
t2tag=[];
hf=figure('NumberTitle','off','ToolBar','figure',...
   'Name','Identificación por el Método de Olderbour Sartorius');
for i=0:0.01:1,
   j=j+1;
   t2tag=[t2tag Tc/Ta-i];
```

```
    if (Tc/Ta-i)>Tg(j)
       a=1;
    end
end
if a==0
    plot(abc,t2tag,abc,Tg);
    grid;
    warndlg('No hay intersección','Advertencia ')
    else
    h1 = uicontrol('Parent',hf, ...
    'Units','points', ...
    'BackgroundColor',[0.8 0.8 0.8], ...
    'ListboxTop',0, ...
    'Position',[ 20.25 63.75 140.5 19.5 ], ...
    'String',['El punto de inflexión ocurre en t= ',num2str(ti)], ...
    'Style','text', ...
    'Tag','StaticText2');
    axes('Position',[ 0.06 0.35 0.40 0.57 ])
    plot(abc,t2tag,'b',abc,Tg,'r');
    grid;
    axis([0 1 0 1]);
    title('Gráfica de OS')
    Tg1r=Tg-t2tag;
    Tg1=find(abs(Tg1r)==min(abs(Tg1r)));
    T1Ta=Tg(Tg1);
    T2Ta=abc(Tg1);
    T1=Tg(Tg1)*Ta;
    T2=abc(Tg1)*Ta;
    h1 = uicontrol('Parent',hf, ...
    'Units','points', ...
    'BackgroundColor',[0.8 0.8 0.8], ...
    'ListboxTop',0, ...
    'Position',[ 20.25 53.75 100.5 19.5 ], ...
    'String',['T1 = ',num2str(T1)], ...
    'Style','text', ...
    'Tag','StaticText2');

    h1 = uicontrol('Parent',hf, ...
    'Units','points', ...
    'BackgroundColor',[0.8 0.8 0.8], ...
    'ListboxTop',0, ...
    'Position',[ 20.25 43.75 100.5 19.5 ], ...
    'String',['T2 = ',num2str(T2)], ...
    'Style','text', ...
    'Tag','StaticText2');
    num=[(max(ys)-ys(1))/us(long)];
    den=[T1*T2 T1+T2 1];
    sal=us(1)*step(num,den,t);
    axes('Position',[ 0.53 0.35 0.4 0.57 ])
    plot(t,ys,'b',t,sal,'r');
    grid;
    axis([0 max(t) 0 max(sal)+0.1*max(sal)]);
    title('Respuestas del Sistema y el Modelo')
    legend('Resp. del sistema', 'Resp. del modelo',4);

    %****************************************
```

```
h1 = uicontrol('Parent',hf, ...
    'Units','points', ...
    'BackgroundColor',[0.8 0.8 0.8], ...
    'ListboxTop',0, ...
    'Position',[271.5 70.25 120 15], ...
    'String','Función de Transferencia', ...
    'Style','text', ...
    'Tag','StaticText7');
h1 = uicontrol('Parent',hf, ...
    'Units','points', ...
    'BackgroundColor',[0.8 0.8 0.8], ...
    'ListboxTop',0, ...
    'Position',[306 50 45 15], ...
    'String',poly2str(num,'s'), ...
    'Style','text', ...
    'Tag','StaticText6');
h1 = uicontrol('Parent',hf, ...
    'Units','points', ...
    'BackgroundColor',[0.8 0.8 0.8], ...
    'ListboxTop',0, ...
    'Position',[ 267.75 38 132 15 ], ...
    'String','_____________________________', ...
    'Style','text', ...
    'Tag','StaticText9');
h1 = uicontrol('Parent',hf, ...
    'Units','points', ...
    'BackgroundColor',[0.8 0.8 0.8], ...
    'ListboxTop',0, ...
    'Position',[275.25 15.5 114.75 15], ...
    'String',poly2str(den,'s'), ...
    'Style','text', ...
    'Tag','StaticText8');

    %****************************************
end
```

Anexo # 2

```
%Programa para la identificación de sistemas subamortiguados con entrada escalón
ys=y.signals.values;
t=y.time;
ue=u.signals.values;
int=t(2)-t(1);
long=length(ys);
k=ys(long)/ue(1); %Ganancia del sistema
%Se calcula el sobrepaso
Mp=(max(ys)-ys(long))/ys(long);
%Se calcula el amortiguamiento
delta=sqrt(log(Mp)^2/(pi^2+log(Mp)^2));
b=find(ys==max(ys));
%Búsqueda del período de osc.
ysf=ys(b:long);%Tomo los puntos de ys del sobrepaso en adelante
c=find(ysf==min(ysf));%Encontramos el punto de mínimo después del sobrepaso
longysf=length(ysf);
ysfm=ysf(c:longysf);%Es el vector desde el mínimo en adelante
d=find(ysfm==max(ysfm));%Es el otro máximo
per=(c+d)*int;
frec=(2*pi)/per;
frecnat=frec/sqrt(1-delta^2);
%La funci{on de transferencia
num=[k*frecnat^2];
den=[1 2*delta*frecnat frecnat^2];
g=tf(num,den);


%*****************************************************


h0 = figure('ToolBar','figure','Position',[ 100 51 560 477 ],...
    'NumberTitle','off','Name',....
    'Identificación de un sistema subamortiguado dada su respuesta a escalón');
posa = [0.07719298245614036 0.3418803418803419 0.8508771929824561 ....
       0.5598290598290598];
ha=axes('position',posa);
h1 = uicontrol('Parent',h0, ...
  'Units','points', ...
  'BackgroundColor',[0.8 0.8 0.8], ...
         'ListboxTop',0, ...
         'Position',[151.5 86.25 120 15], ...
         'String','Función de Transferencia', ...
         'Style','text', ...
         'Tag','StaticText7');
h1 = uicontrol('Parent',h0, ...
```

```
   'Units','points', ...
   'BackgroundColor',[0.8 0.8 0.8], ...
          'ListboxTop',0, ...
   'Position',[186 66 45 15], ...
   'String',poly2str(num,'s'), ...
          'Style','text', ...
          'Tag','StaticText6');
h1 = uicontrol('Parent',h0, ...
   'Units','points', ...
   'BackgroundColor',[0.8 0.8 0.8], ...
          'ListboxTop',0, ...
    'Position',[ 147.75 54 132 15 ], ...
          'String','_____', ...
          'Style','text', ...
          'Tag','StaticText9');
h1 = uicontrol('Parent',h0, ...
   'Units','points', ...
   'BackgroundColor',[0.8 0.8 0.8], ...
          'ListboxTop',0, ...
   'Position',[135.25 31.5 140.75 15], ...
   'String',poly2str(den,'s'), ...
          'Style','text', ...
          'Tag','StaticText8');

%*******************************************************
plot(t,ys,'r');
hold on
ysm=ue(1)*step(g,t);
plot(t,ysm,'b');
grid;
legend('Respuesta del sistema','Respuesta del modelo');
```

Anexo # 3

```
%Convierte respuesta a un pulso a respuesta a escalón
clf
veces=40;
ys=y.signals.values;
ue=u.signals.values;
t=y.time;
im=t(2)-t(1);
dondecero=find(ue==0);
Ancho=dondecero*im;
indcero=min(dondecero);
ystotal=[ys; zeros(veces*indcero,1)];
t=[t; (max(t+im):im:(length(ystotal)-1)*im)'];
p=1;
for i=1:veces
   ystotalA=[zeros(indcero*p,1);ys;zeros((veces-p)*indcero,1)];
   ystotal=ystotal+ystotalA;
   plot(t,ystotal);
   hold on
   p=p+1;
end
%La respuesta a escalón está en ystotal
plot(t,ystotal,'r');
```

Anexo # 4

```
%Identificación por el método de correlación
clf
ys=y.signals.values;
ue=u.signals.values;
t=y.time;
tm=t(2)-t(1);
ir=cra([ys ue],450,10,0);
close
Lir=length(ir);
t1=0:tm:(Lir-1)*tm;
h0 = figure('ToolBar','figure','Position',...
 [ 100 51 560 477 ],'NumberTitle','off','Name',....
 'Identificación de un sistema por el método de corelación');
axes('position',...
 [0.119642857142857 0.61625 0.775 0.310267857142857]);
plot(t1,ir)
title('Respuesta a impulso obtenida')
axis([0 t1(end)/2 0 1.1*max(ir)])
[n,d]=imp2sys(ir);
sysd=tf(n,d,tm);
sysc=d2c(sysd)
ym=step(sysc,t);
axes('position',...
 [0.123214285714286 0.205714285714286 0.775 0.321428571428571]);
plot(t,ys,'r',t,ym,'b')
axis([0 t(end)/2 0 1.1*max(ys)])
legend('Respuesta del sistema','Respuesta del modelo')
title('Respuestas a escalón del sistema y el modelo')

%*************************************************************

h1 = uicontrol('Parent',h0, ...
   'Units','points', ...
   'BackgroundColor',[0.8 0.8 0.8], ...
   'ListboxTop',0, ...
   'Position',[151.5 40.25 120 15], ...
   'String','Función de Transferencia', ...
   'Style','text', ...
   'Tag','StaticText7');
h1 = uicontrol('Parent',h0, ...
   'Units','points', ...
   'BackgroundColor',[0.8 0.8 0.8], ...
   'ListboxTop',0, ...
   'Position',[ 114.75 17.25 192.75 24.75 ], ...
   'String',poly2str(sysc.num{1},'s'), ...
   'Style','text', ...
   'Tag','StaticText6');
h1 = uicontrol('Parent',h0, ...
   'Units','points', ...
   'BackgroundColor',[0.8 0.8 0.8], ...
   'ListboxTop',0, ...
    'Position',[ 147.75 18 132 15 ], ...
   'String','_____________________________', ...
   'Style','text', ...
```

```
    'Tag','StaticText9');
h1 = uicontrol('Parent',h0, ...
    'Units','points', ...
    'BackgroundColor',[0.8 0.8 0.8], ...
    'ListboxTop',0, ...
    'Position',[135.25 5.5 140.75 15], ...
    'String',poly2str(sysc.den{1},'s'), ...
    'Style','text', ...
    'Tag','StaticText8');

%**********************************************************************
```

Anexo # 5

```
%Método de deconvolución por mínimos cuadrados
ue=u.signals.values;
ys=y.signals.values;
t=y.time;
clf
tm=t(2)-t(1);
%Hasta aquí pasamos al workspace las mediciones
%de la entrada y la salida.
int=t(2)-t(1);
n=length(ue);
a=zeros(n);
for i=1:n,
     k=i;
     for j=1:i,
        a(i,j)=ue(k);
        k=k-1;
     end
end
%Hemos formado la matriz A
h=a\ys;
h=h/int;
plot(t,h);
grid
[n,d]=imp2sys(h);
sysd=tf(n,d,int)
sysc=d2c(sysd)*tm
```

Anexo # 6

```
%Programa principal de Integración gradual
orden=input('Teclee el orden ');
us=u.signals.values;
ys=y.signals.values;
t=y.time;
Tsamp=t(2)-t(1);
z=[ys us];
[Ncap,nz]=size(z);
nu=nz-1;
if nz>Ncap,error('Los datos deben organizarse en un vector columna')
    return,end
if nu~=1,error('Esta función solo trabaja con sistemas SISO')
    return,end
num=[];den=[];
maxsize=50000;
if orden ==2
    [a1 a0 b0 b1]=anaintg2(us,ys,Tsamp);
    num=[b1 b0];
    den=[1 a1 a0];
end
if orden ==1
    [a0,b0]=anaintg1(us,ys,Ncap,Tsamp);
    num=b0;
    den=[1 a0];
end
funtrans=tf(num,den)
Ressys=lsim(funtrans,us,t);
plot(t,Ressys,'b')
hold on
plot(t,ys,'r')
grid
text(mean(t),max(ys)/2,'azul ... Resp. del sistema')
text(fix(mean(t)),max(ys)/2-max(ys)/5,'rojo ... Resp. del modelo')

function [A,B]=anaintg1(Ut,Yt,n,Delta)
       Matriz_U = Ut;
       Matriz_Y = Yt;
       n=length(Yt);
       if rem(n,2)>0,
          Matriz_U=[Matriz_U;Matriz_U(length(Matriz_U))];
          Matriz_Y=[Matriz_Y;Matriz_Y(length(Matriz_Y))];
          n=n+1;
       end,

       Matriz_J=simpsom(n);
       Matriz_J = (Delta/12)*Matriz_J;

Matriz_1C = Matriz_J*Matriz_Y;
Matriz_2C = Matriz_J*Matriz_U*(-1);
Matriz_P=[Matriz_1C Matriz_2C];
Matriz_Y= (-1)*Matriz_Y;
Matriz_Coef=Matriz_P\Matriz_Y;
A = Matriz_Coef(1);
B = Matriz_Coef(2);
```

```
function [A,B,C,D]=anaintg2(Ent,Sal,Delta)
       Matriz_U=Ent;
       Matriz_Y=Sal;
       n=size(Sal,1);
       if rem(n,2)>0,
          Matriz_U=[Matriz_U;Matriz_U(length(Matriz_U))];
          Matriz_Y=[Matriz_Y;Matriz_Y(length(Matriz_Y))];
          n=n+1;
       end,
       Matriz_J=simpsom(n);
       Matriz_J = (Delta/12)*Matriz_J;
       Matriz_J2= Matriz_J*Matriz_J;
       Matriz_1C = Matriz_J*Matriz_Y;
       Matriz_2C = Matriz_J2*Matriz_Y;
       Matriz_3C = (-1)*(Matriz_J2*Matriz_U);
       Matriz_4C = (-1)*Matriz_J*Matriz_U;
       Matriz_P = [Matriz_1C Matriz_2C Matriz_3C Matriz_4C];
       Matriz_T= (-1)*Matriz_Y;
       MatC = Matriz_P\Matriz_T;
       A=MatC(1);
       B=MatC(2);
       C=MatC(3);
       D=MatC(4);


function [MatJ]=simpsom(n)
matriz_J=zeros(n,n-1);
hw = processbar(0,'Identificando...');
tn=n+n;
in=n;
    for i=1:n,
      if rem(i,2)
         for j=1:i,
             if rem(j,2)
                matriz_J(i,j)=16;
             else
                matriz_J(i,j)=8;
             end
         end
      else
      for j=1:i,
        if rem(j,2)
           matriz_J(i,j)=16;
        else
           matriz_J(i,j)=8;
        end
      end
   end
   processbar(i/tn)
 end

for i=1:n,
  for j=1:n,
       if rem(i,2)
          matriz_J(i,i+1)=-1;
          matriz_J(i,i)=8;
```

```
            if i-1
              matriz_J(i,i-1)=9;
            end
        else
            matriz_J(i,i)=4;
    end
   end
   processbar((in+1)/tn)
   in=in+1;
end
close(hw);
matriz_J(:,n)=[];
Columna1=4*ones(n,1);
Columna1(1)=5;
MatJ=[Columna1 matriz_J];
```

Anexo # 7

```
%Se hace la identificación con el toolbox de Matlab
%Estructura ARX
ys=y.signals.values;
ue=u.signals.values;
t=y.time;
im=t(2)-t(1);
n=length(ys);
z=[ys ue];
th=arx(z,[3 3 1]);
th=sett(th,im);
tc=thd2thc(th);
[num,den]=th2tf(tc);
ft=tf(num,den);
zp=zpk(ft);
ym=lsim(ft,ue,t);
ema=mean(abs(ym-ys));

h0 = figure('ToolBar','figure','Position',[ 100 51 560 477 ],...
   'NumberTitle','off','Name',....
   'Obtención del modelo de un sistema por el método ARX');

posa = [0.0771929824561 0.341880341880 0.850877192982 0.559829059829];
ha=axes('position',posa);

h1 = uicontrol('Parent',h0, ...
   'Units','points', ...
   'BackgroundColor',[0.8 0.8 0.8], ...
   'ListboxTop',0, ...
   'Position',[ 13.5 84 135 19.5 ], ...
   'String',['El error medio es ' num2str(ema)], ...
   'Style','text', ...
   'Tag','StaticText3');
h1 = uicontrol('Parent',h0, ...
   'Units','points', ...
   'BackgroundColor',[0.8 0.8 0.8], ...
   'ListboxTop',0, ...
   'Position',[ 205.5 88.5 120 15 ], ...
   'String','Función de Transferencia', ...
   'Style','text', ...
   'Tag','StaticText7');
h1 = uicontrol('Parent',h0, ...
   'Units','points', ...
   'BackgroundColor',[0.8 0.8 0.8], ...
   'ListboxTop',0, ...
   'Position',[ 134.25 65.25 257.25 15.75 ], ...
   'String',poly2str(num,'s'), ...
   'Style','text', ...
   'Tag','StaticText6');
h1 = uicontrol('Parent',h0, ...
   'Units','points', ...
   'BackgroundColor',[0.8 0.8 0.8], ...
   'ListboxTop',0, ...
   'Position',[ 147.75 48.75 234.75 20.25 ], ...
   'String','_____', ...
   'Style','text', ...
   'Tag','StaticText9');
```

```
h1 = uicontrol('Parent',h0, ...
    'Units','points', ...
    'BackgroundColor',[0.8 0.8 0.8], ...
    'ListboxTop',0, ...
    'Position',[ 116.25 33.75 288.75 15 ], ...
    'String',poly2str(den,'s'), ...
    'Style','text', ...
    'Tag','StaticText8');

plot(t,ym,'b',t,ys,'r');
legend('Respuesta del modelo','Respuesta del sistema')
grid;
```

Anexo # 8

%Identifiación de sistemas multivariables (2 entradas y 2 salidas)
%por el método arx

```
clf
ys=y.signals.values;
ue=u.signals.values;
t=y.time;
tm=t(2)-t(1);
z=[ys ue];
na=[2 2;2 2];
nb=[2 2;2 2];
nk=[0 0;0 0];
nn=[na nb nk];
th=arx(z,nn);
th=sett(th,tm);
tc=thd2thc(th);
[A,B,C,D,K,X0]=th2ss(tc);
sysss=ss(A,B,C,D);
[Y,T] = lsim(sysss,ue,t);
figure(1)
plot(T,Y(:,1),'b',t,ys(:,1),'r')
legend('Salida 1 del Modelo','Salida 1 del Sistema')
grid on
figure(2)
plot(T,Y(:,2),'b',t,ys(:,2),'r')
grid on
legend('Salida 2 del Modelo','Salida 2 del Sistema')
er1=mean(abs(Y(:,1)-ys(:,1)));
er2=mean(abs(Y(:,2)-ys(:,2)));
disp(['El error para la salida 1 es ',num2str(er1)]);
disp(['El error para la salida 2 es ',num2str(er2)]);
disp('Comparemos los vectores propios del sistema y el modelo')
disp('Del modelo')
eig(A)
disp('Del sistema')
eig([0 1;-25 -4])
```

Anexo # 9

```
clf
ys=y.signals.values;
ue=u.signals.values;
t=y.time;
tm=t(2)-t(1);
z=[ys ue];
na=[2 2;2 2];
nb=[2 2;2 2];
nk=[0 0;0 0];
nn=[na nb nk];
th=arx(z,nn);
th=sett(th,tm);
tc=thd2thc(th);
[A,B,C,D,K,X0]=th2ss(tc);
sysss=ss(A,B,C,D);
[Y,T] = lsim(sysss,ue,t);
subplot(211)
plot(T,Y(:,1),'b',t,ys(:,1),'r')
legend('Salida 1 del Modelo','Salida 1 del Sistema')
grid on
subplot(212)
plot(T,Y(:,2),'b',t,ys(:,2),'r')
grid on
legend('Salida 2 del Modelo','Salida 2 del Sistema')
er1=mean(abs(Y(:,1)-ys(:,1)));
er2=mean(abs(Y(:,2)-ys(:,2)));
disp(['El error para la salida 1 es ',num2str(er1)]);
disp(['El error para la salida 2 es ',num2str(er2)]);
disp('Vector propio del modelo')
eig(A)
```

Bibliografía

[1] Arafet, P. y col., "Introducción al Matlab", Monografía. Universidad de Oriente, Santiago de Cuba, Enero 2001.

[2] Ogata, K., "Modern Control Engineering", Ediciones del Castillo, 1998.

[3] Ollero, A., "Control por Computador. Descripción interna y diseño óptimo", Editorial Marcombo, 1991.

[4] Trejo, V., "Identificación experimental de sistemas", Editorial ISPJAE, 1986.

[5] León, E. "Sistema para la identificación dinámica". Tesis de Maestría Universidad de Oriente. Santiago de Cuba, Abril, 2002.

[6] Sage y Melsa, "System Identification,", Mathematics in science and engineering, volume 80, AcademicPress, Inc.