ASPECTOS COMPUTACIONALES DE ALGUNOS MÉTODOS DE AJUSTE PARAMÉTRICO DE MODELOS APLICADOS A CIERTOS PROCESOS DE POLIMERIZACIÓN

OSCAR SEGUNDO ACUÑA CAMACHO

DAYRO FABIÁN GUTIÉRREZ DE LA HOZ

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR

MAESTRÍA EN CIENCIAS COMPUTACIONALES

Cartagena de Indias, D.T. y C. Julio de 2006

ASPECTOS COMPUTACIONALES DE ALGUNOS MÉTODOS DE AJUSTE PARAMÉTRICO DE MODELOS APLICADOS A CIERTOS PROCESOS DE POLIMERIZACIÓN

OSCAR SEGUNDO ACUÑA CAMACHO

DAYRO FABIÁN GUTIÉRREZ DE LA HOZ

Director

Dr. JOSE LUIS VILLA

Universidad Tecnológica de Bolívar

UNAB

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY – UNIVERSIDAD AUTONOMA DE BUCARAMANGA – UTB

MAESTRÍA EN CIENCIAS COMPUTACIONALES

Cartagena de Indias, D.T. y C. Julio de 2006

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

INTRODUCCION

En los últimos años se ha producido un cambio dramático en la industria de los procesos químicos. Los procesos industriales están ahora altamente integrados con respecto a los flujos de materia y energía, limitados aún mas fuertemente por altas calidades en las especificaciones de los productos y sujetos a estrictas medidas de seguridad y a la regulación de emisiones ambientales. Estas severas condiciones de operación a menudo colocan nuevas restricciones en la flexibilidad en la operación de los procesos. Todos estos factores producen grandes incentivos económicos para el mejoramiento y buen desempeño en los sistemas de control de las plantas industriales modernas [26].

Estas plantas requieren de sofisticados sistemas de cómputo para la implementación de estrategias de control. Es así que la mayoría de las nuevas plantas en las industrias químicas, del petróleo, papel, acero y otras están diseñadas y construidas con redes de microcomputadores para la adquisición de datos y control del proceso.

El diseño de estrategias de control clásico y avanzado requiere normalmente de desarrollos teóricos y de simulaciones dinámicas, que deben basarse en un modelo matemático del proceso a controlar. Un conocimiento del comportamiento dinámico de los procesos químicos es importante tanto para el

diseño como para las perspectivas de control del proceso. Es relativamente fácil diseñar un proceso químico basado en las condiciones en estado estable, el cual es prácticamente incontrolable cuando las dinámicas del proceso son consideradas. El estado estable es aquel estado en el que el sistema alcanza su equilibrio y los fenómenos transitorios desaparecen; ver [42], [43], [44], [45].

En el modelamiento del comportamiento de la dinámica de los procesos, el principal objetivo puede ser la obtención de una relación funcional entre algunas variables del proceso, las cuales explican el comportamiento del proceso observado. La verdadera relación fundamental está determinada por las leyes naturales, que existen entre las variables del proceso, por lo que el problema del modelado se reduce a obtener estas relaciones.

Se entiende claramente que es imposible representar perfectamente todos los detalles del comportamiento actual del proceso en forma matemática. De allí que el *modelamiento teórico* busca una forma de llegar a una relación de modelamiento fundamental a través de la aplicación sistemática de las leyes de la naturaleza a los *fenómenos más importantes* asumidos para determinar el comportamiento del proceso. El resultado final es conocido como un *modelo teórico del proceso.*

De otra parte, la *identificación empírica* busca la forma de aproximar esta relación funcional desconocida por algunas funciones matemáticas, usando

información recogida experimentalmente del sistema. A raíz de que ellas dependen enteramente de la información experimental y de la experiencia misma, las funciones matemáticas aproximadas obtenidas son conocidas como *modelos empíricos*, se puede ampliar la información en la referencia [1].

Es típico adoptar el enfoque del modelamiento teórico, cuando los mecanismos fundamentales bajo los cuales un proceso opera son razonablemente bien conocidos. Cuando el proceso es demasiado complicado para el enfoque teórico (usualmente porque muy poca información de la naturaleza fundamental del proceso está disponible, o las ecuaciones del modelo teórico son muy complejas) el enfoque empírico es la selección apropiada. El tema completo de la construcción de un modelo de procesos a partir de información puramente experimental es conocido como *Identificación de Procesos.*

En este último caso, una estructura general del modelo puede ser usada, en el cual todos los parámetros deben ser estimados a partir de las mediciones de entradas y salidas, usando un procedimiento de estimación. La identificación paramétrica se refiere a la identificación de modelos paramétricos. Estos modelos pueden ser usados para el análisis de procesos, para la simulación y para el diseño del control [18].

Para obtener los parámetros adecuados se utilizan algunas técnicas o métodos en los que se aplican algoritmos que normalmente se caracterizan por clasificarse con una determinada complejidad computacional.

En el desarrollo de los métodos existe la probabilidad de que el sistema converja a una solución del modelo o que por el contrario no converja, caso en el cual se debe hacer iteraciones de acuerdo con los diferentes posibles parámetros de ajuste del algoritmo implementado. Otro elemento importante es que los datos deben ser procesados a través de un filtrado que permita reducir el error en la construcción del modelo; esta información puede ser ampliada por el lector en las referencias [1], [2], [3].

En la construcción de los modelos de procesos de polimerización, de acuerdo con la literatura consultada, se aplican diferentes técnicas y métodos de ajuste paramétrico, pero sin embargo los aspectos computacionales de los algoritmos son mencionados de manera muy general.

En este trabajo hay especial interés por estudiar los modelos paramétricos lineales y específicamente el modelo con estructura ARX, donde “AR” hace referencia a la parte autorregresiva y “X” a la entrada externa (exogenous input) también conocida como variable exógena. En el capítulo I se ampliará esta información con más detalle.

Para este trabajo, se entiende por “aspectos computacionales”, los relacionados con el algoritmo mismo y con criterios de comparación tales como su tiempo de ejecución y la validez de los modelos obtenidos en términos de parámetros como el Error de Predicción Final (FPE).

Con el análisis de los mejores métodos y técnicas de ajuste de parámetros, se logra construir un buen modelo a través de la identificación paramétrica; se entiende por un buen modelo aquel que es capaz de predecir el comportamiento del sistema modelado con un mínimo de error; los buenos modelos permiten simulaciones adecuadas para los procesos de diseño y estrategias de control en los procesos de producción. Para el análisis de los algoritmos seleccionados se desarrolla la aplicación sobre un modelo de CSTR propuesto por [30].

En la literatura consultada hay muchos trabajos sobre el modelamiento de reactores químicos como el que nos ocupa en esta investigación. La gran mayoría usa el enfoque del modelo teórico basado en el conocimiento del proceso [4], [5], [6], [7].

En el capítulo I se explica el panorama actual de los métodos o técnicas de ajuste de parámetros de modelos dinámicos y se hace un análisis de los aspectos computacionales más importante en el desarrollo de algoritmos.

En el capítulo II se hace una revisión bibliográfica de los métodos de identificación paramétrica de reactores de polimerización y de los métodos de estimación de parámetros. A partir de la revisión se describe cada uno de los métodos seleccionados para su posterior implementación

En el capítulo III se hace una descripción de la implementación de los algoritmos seleccionados en el capítulo II para aplicar a los ejemplos utilizados en este trabajo.

En el capítulo IV se aplica cada uno de los algoritmos implementados a un modelo de reactor, bajo las mismas condiciones para poder hacer una posterior evaluación de su desempeño.

En el capítulo V se hace el análisis comparativo de las técnicas enfocado al comportamiento computacional de los algoritmos y a los criterios de comparación previamente definidos.

Posteriormente se desarrollan las conclusiones sobre la investigación y algunas recomendaciones para trabajos futuros.

CAPITULO I.- FUNDAMENTOS Y ESTADO DEL ARTE

La identificación de sistemas y la estimación de parámetros como tal ha sido un campo de investigación muy estudiado por diferentes autores. El objetivo es el de encontrar modelos de sistemas dinámicos válidos y eficientes a partir de observaciones de un sistema real. Entendiendo la validez como la capacidad de exhibir un comportamiento similar, bajo algún parámetro predefinido, con respecto al sistema real y la eficiencia en términos de la capacidad para manejar recursos computacionales adecuados.

Este capítulo presenta un resumen de los métodos de identificación usuales y de los aspectos de complejidad computacional de un algoritmo.

1.1 IDENTIFICACIÓN DE SISTEMAS Y ALGORITMOS

Los modelos de sistemas pueden ser paramétricos, que tienen la ventaja de estar dados por un conjunto pequeño de coeficientes, o bien no paramétricos como las redes neuronales, que tienen la ventaja de no estar restringidas a un cierto número, posiblemente pequeño, de descripciones posibles del modelo.

Aunque la literatura en identificación de sistemas es amplia, pocos trabajos abordan específicamente el problema de aspectos computacionales asociado a los algoritmos de identificación paramétrica, excepto en los casos donde se

prevee de antemano altos requerimientos computaciones, como es el caso de los algoritmos genéticos. Entre los trabajos que tratan este tópico, Svolos en [8] plantea un método para la reducción del tiempo computacional y el espacio de memoria requerido en su trabajo sobre análisis de imágenes en el campo de la medicina.

Chong en [9] hace un análisis sobre el comportamiento algorítmico e incluye un procedimiento para reducir el tiempo computacional en su trabajo aplicado a la robótica sobre una versión rápida del algoritmo de distancia de Gilbert-Johnson-Keerthi. En [10] Floreen describe la manera como construir redes que probablemente converjan a un resultado apropiado y un límite restringido sobre el tiempo de convergencia; de igual manera plantea que con una elección específica de los parámetros el tiempo de convergencia en el peor de los casos puede ser calculado.

Chicula en [11] plantea una metodología para el diseño secuencial de entradas de excitación para la identificación paramétrica de una clase de sistema no lineal. La optimización basada en este método permite la incorporación de consideraciones en lazo cerrado sobre características conocidas de la planta. En la construcción del modelo el esquema de estimación es el de mínimos cuadrados con observaciones independientes.

En [12] se presentan unos resultados interesantes sobre la validez de los modelos mediante la comparación de dos métodos de identificación paramétrica, el método tradicional de ajuste de curva y el método de parámetros secuenciales. El segundo de estos métodos presenta los mejores resultados fundamentado en el criterio del error de predicción del modelo obtenido.

En [13], Garrido desarrolla un método de identificación de sistemas no lineales usando una técnica sobre optimización genética restringida; en este trabajo desarrolla un algoritmo con un método de altas prestaciones para la identificación de sistemas no lineales variables con el tiempo con modelos lineales y no lineales.

Estos estudios tienen en cuenta algunos aspectos computacionales asociados al desarrollo de algoritmos tales como el tiempo computacional consumido, la capacidad de memoria de almacenamiento, el conjunto o tamaño de datos para experimentación, los cuales de alguna manera determinan la eficiencia del algoritmo.

Dado que este trabajo estudia algunos aspectos computacionales de algoritmos de identificación paramétrica de modelos lineales, en la siguiente sección se hace un compendio de las estructuras de los modelos, las técnicas de identificación.

1.2 MODELOS DE SISTEMAS DINÁMICOS Y MÉTODOS DE IDENTIFICACIÓN

En la siguiente sección se presentan diferentes estructuras de modelos como ilustración general, aunque buena parte de ellos no sean utilizados en este trabajo.

1.2.1 Modelos de Entrada-Salida. Un modelo de entrada-salida describe un sistema dinámico basándose en los datos de entrada y de salida. Dado que los datos obtenidos de un proceso real normalmente son mediciones discretas en tiempo, es natural utilizar modelos en tiempo discreto para la identificación de sistemas. Todos los modelos tratados en este trabajo son en tiempo discreto. Siendo coherente con la diversidad de la bibliografía utilizada, el índice *k* y el índice *t* denotarán tiempo discreto indistintamente.

En tiempo discreto, este tipo de modelos supone que la salida del sistema puede ser predicha a partir de las entradas y salidas pasadas del sistema. Si el sistema se supone determinista, invariante en el tiempo, de una entrada - una salida (SISO), el modelo de entrada-salida puede ser representado como:

$$y(k) = f(y(k-1), y(k-2),..., y(k-n), u(k-1), u(k-2),..., u(k-m))$$

Donde $u(k)$ y $y(k)$ representan el par de entrada-salida en el índice de tiempo k. Los enteros positivos *n* y *m* son índices de retardos medidos usualmente en

una unidad de tiempo constante, y representan el número de retardos en las señales de salida (también llamado el orden del sistema) y el número de retardos en las señales de entrada, respectivamente. En la práctica *m* es, normalmente menor o igual que *n*; *f(.)* puede ser cualquier función no-lineal que mapea el espacio de la señales de entradas y salidas pasadas en el espacio de la señal de salida en el índice de tiempo presente.

1.2.2 Modelos Paramétricos Lineales. El estado actual de la estimación de parámetros en la identificación de sistemas corresponde a la teoría clásica de la regresión. Algunos de los modelos lineales más conocidos son:

1.2.2.1 ARX (Auto-Regressive with eXogenous inputs). En este caso se considera *f(.*) como una función lineal. La relación básica entre la entrada y la salida del sistema está dada por la ecuación en diferencia lineal de la ecuación 1.1.

$$y(t) + a_1 y(t-1) + \ldots + a_n y(t-n) = b_1 u(t-1) + \ldots + b_m u(t-m) \quad (1.1)$$

En la ecuación 1.1 asumimos que el *intervalo de muestreo* será una unidad de tiempo constante.

Una forma práctica y útil de ver (1.1) es mirarlo como una forma de *determinar el próximo valor de salida* dada las observaciones previas:

$$y(t) = -a_1 y(t-1) - \ldots - a_n y(t-n) + b_1 u(t-1) + \ldots + b_m u(t-m) \quad (1.2)$$

Donde a_i y b_i corresponden a coeficientes constantes de las salidas y las entradas en tiempo pasado respectivamente. La fortaleza de esta descripción es que puede ser expresada en forma matricial, como se indica en las ecuaciones (1.3) y (1.4).

$$\theta = [a_1 \quad \ldots \quad a_n \quad b_1 \quad \ldots \quad b_m]^T \quad (1.3)$$

$$\varphi(t) = [-y(t-1) \quad \ldots \quad -y(t-n) \quad u(t-1) \quad \ldots \quad u(t-m)]^T \quad (1.4)$$

Donde θ corresponde al vector de parámetros, es decir información desconocida, y φ corresponde a la matriz de observación, es decir información conocida a partir de observaciones del sistema real. Con esto (1.2) puede ser rescrita como

$$y(t) = \varphi^T(t)\theta$$

Para enfatizar que los cálculos de y(t) de los datos pasados (1.2) dependen de los parámetros de θ, se escriben estos valores calculados como

$$\hat{y}(t \mid \theta) = \varphi^T(t)\theta \quad (1.5)$$

Donde $\hat{y}(t \mid \theta)$ corresponde a la salida de predicción del modelo. Como se explicará detalladamente en el capítulo II, esta forma resulta muy conveniente para la solución numérica del problema.

1.2.2.2 OE (Output Error). Si se supone que la relación entre la entrada y la salida w puede ser escrita como una ecuación en diferencia lineal, y que las perturbaciones consisten de ruido blanco medible, se puede obtener la siguiente descripción:

$$w(t) + f_1 w(t-1) + \ldots + f_{nf} w(t-n_f) = b_1 u(t-1) + \ldots + b_{nb} u(t-n_b) \qquad (1.6)$$

$$y(t) = w(t) + e(t) \qquad (1.7)$$

Con

$$F(q) = 1 + f_1 q^{-1} + \ldots + f_{nf} q^{-nf} \qquad (1.8)$$

Se puede escribir el modelo como

$$y(t) = \frac{B(q)}{F(q)} u(t) + e(t) \qquad (1.9)$$

Donde *q* representa el operador retardo. En este caso la representación es coherente con los modelos en función de transferencia en tiempo discreto.

El flujo de señal de este modelo se muestra en la figura 1.1:

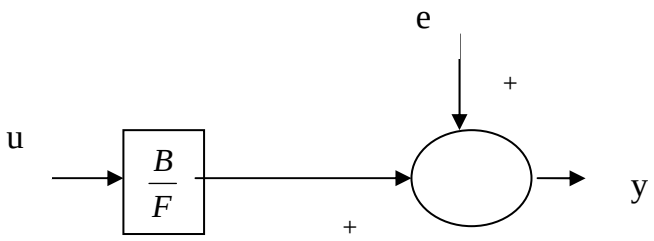

Fig. 1.1. Estructura del modelo OE.

La ecuación (1.10) representa la estructura del modelo OE. El vector de parámetros a ser determinado es:

$$\theta = \begin{bmatrix} b_1 & b_2 & \dots & b_n & f_1 & f_2 & \dots & f_n \end{bmatrix}^T \qquad (1.10)$$

Donde $w(t)$ en (1.7) nunca es observada; esto podría llevar correctamente a un índice θ, desde que sea construido de u usando (1.6). Esto es,

$$w(t,\theta) + f_1 w(t-1),\theta) + \dots + f_{nf} w(t-n_f,\theta) = b_1 u(t-1) + \dots + b_{nb} u(t-n_b) \qquad (1.11)$$

Con lo cual se puede obtener el predictor natural

$$\hat{y}(t \mid \theta) = \frac{B(q)}{F(q)} u(t) = u(t,\theta) \qquad (1.12)$$

Note que $\hat{y}(t \mid \theta)$ es construida a partir de las salidas pasadas solamente. Con la ayuda del vector

$$\varphi(t,\theta) = \left[u(t-1) \quad ... \quad u(t-n_b) \quad -w(t-1,\theta) \quad ... \quad -w(t-n_f,\theta)\right]^T \qquad (1.13)$$

Puede ser escrita como

$$\hat{y}(t \mid \theta) = \varphi^T(t,\theta)\theta \qquad (1.14)$$

Lo cual está de acuerdo con el predictor del modelo ARMAX que se describe en la siguiente sección.

1.2.2.3 **ARMAX** (Auto-Regressive Moving Average with eXogenous inputs).

Al describir la expresión del error como un promedio movil de ruido blanco se obtiene el modelo

$$y(t) + a_1 y(t-1) + ... + a_{n_a} y(t-n_a) = b_1 u(t-1) + ... + b_{n_b} u(t-n_b)$$

$$+ e(t) + c_1 e(t-1) + ... + c_n\ e(t-n_{c)} \qquad (1.15)$$

Con

$$C(q) = 1 + c_1 q^{-1} + ... + c_{n_c} q^{-n_c} \qquad (1.16)$$

Puede ser rescrita

$$A(q)y(t) = B(q)u(t) + c(q)e(t) \qquad (1.17)$$

Que corresponde con

$$G(q,\theta)=\frac{B(q)}{A(q)},\qquad H(q,\theta)=\frac{C(q)}{A(q)}$$

Donde

$$\theta=\begin{bmatrix}a_1 & \dots & a_{n_a} & b_1 & \dots & b_{n_b} & c_1 & \dots & c_{n_c}\end{bmatrix}^T \qquad (1.18)$$

En [18] se puede obtener una amplia información sobre la obtención del predictor, el cual se describe como

$$\hat{y}(t\mid\theta)=B(q)u(t)+[1-A(q)]y(t)+[C(q)-1][y(t)-\hat{y}(t\mid\theta)] \qquad (1.19)$$

Se introduce el error de predicción

$$\varepsilon(t,\theta)=y(t)-\hat{y}(t\mid\theta) \qquad (1.20)$$

Y el vector

$$j(t,q)=[-y(t-1)\ \dots\ -y(t-n_a)\ \ u(t-1)\ \dots\ u(t-n_b)\,e(t-1,q)\dots e(t-n_c,q)]^T$$

(1.21)

Y entonces puede ser reescrito como

$$\hat{y}(t\mid\theta)=\varphi^T(t,\theta)\theta \qquad (1.22)$$

1.2.3 Modelos Paramétricos No-lineales. Una no linealidad entre la relación de secuencia de entrada y la secuencia de salida da muchas posibilidades para describir un sistema. Al mismo tiempo, la situación es tan flexible que se hace necesario inferir estructuras no lineales generales de registros de datos finitos. Existen varias posibilidades de parametrizar mapeos generales de datos pasados para la obtención del predictor. En algunos casos se hacen combinaciones lineales de sistemas no lineales.

Supongamos que se ha escogido una base del espacio de funciones {g(k)}, debemos intentar aproximar la verdadera relación por una combinación lineal finita de las funciones de la base:

$$\hat{y}(t/\theta) = g(\phi(t),\theta) = \sum_{k=1}^{n} \theta(k) g_k(\phi(t)) \quad (1.23)$$

donde $\phi(t)$ está formado por las entradas u(k) y salidas y(k) pasadas:

$$\phi(t) = [y(t-1),...,y(t-n),u(t-1),...,u(y-m)]^T \quad (1.24)$$

y hemos introducido la notación $\hat{y}(t/\theta)$ para hacer hincapié en que $g(\phi(t),\theta)$ es una estimación de y(t) dada la información $\phi(t)$ y un vector de parámetros dado θ.

El mejor valor de θ está determinado entonces por el conjunto de datos

$$Z^N = \{[y(t), \phi(t)] / t = 1,..., N\} \quad (1.25)$$

y será:

$$\hat{\theta}_N = \arg\min \sum_{k=s}^{N} \left| y(t) - \hat{y}(t/\theta) \right|^2 \quad (1.26)$$

1.2.3.1 Series de Volterra. Las series de Volterra, desarrollada por el matemático del mismo nombre, en 1910, con un orden y un retardo apropiadamente escogidos pueden ser usados para modelar una cierta clase de modelos no lineales. Las series de Volterra vienen descritas por la expresión:

$$y(k) = \sum_{i=0}^{\infty} a_i u(k-i) + \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} b_{ij} u(k-i)u(k-j) + \sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\sum_{l=0}^{\infty} c_{ijl} u(k-i)u(k-j)u(-l) + ... \quad (1.27)$$

El vector *u* representa la señal de entrada, el vector *y* la salida del modelo. Las series de Volterra pueden ser representadas en notación vectorial:

$$y(k) = a_1^T x_1 + b_2^T x_2 + c_3^T x_3 = x^T a \quad (1.28)$$

Donde el vector de datos x y el vector de coeficientes viene dado por:

$$\chi^T = (x(k), x(k-1),..., x(k-N+1), x^2(k), x(k)x(k-1),..., x^2(k-N+1), x^3(k),..., x^3(k-N+1)$$

(1.29)

y

$$a^T = (a_0 .....a_{N-1}, b_{00} .....b_{N-1,N-1}, c_{000} .....c_{N-1,N-1,N-1})$$

por lo que:

$$a^T = (a_1^T, b_2^T, c_3^T) \quad (1.30)$$

Los parámetros desconocidos de la serie pueden ser determinados a partir de las medidas de entrada-salida resolviendo la ecuación

$$y = Xa \quad (1.31)$$

1.2.3.2 Modelos NARMAX. Entre los modelos no lineales el modelo NARMAX (Non- linear Auto-Regressive Moving Average with eXogenous inputs), es una generalización del modelo ARX y que puede ser descrito como:

$$y(t) = F[y(t-1),..., y(t-n_y), u(t-d),...u(t-d-n_u+1), \varepsilon(t-1),..., \varepsilon(t-n_\varepsilon)] + \varepsilon(t) \quad (1.32)$$

donde *t* representa el tiempo discreto, *y(t)*, *u(t)* y *ε(t)* son las señales de salida, la entrada, y el error de predicción, respectivamente, n_y, n_u y n_ε son los órdenes correspondientes, *F*[.] es una función no lineal y *d* es el mínimo retardo de la entrada [14]. El modelo polinomial NARMAX tiene como expresión:

$$y(k) = a_0 + \sum_{i=1}^{n_y} a_i y(k-i) + \sum_{i=1}^{n_u} b_i u(k-i) + \sum_{i=1}^{n_y}\sum_{i=1}^{n_u} c_{ij} y(k-i)u(k-j) + \sum_{i=1}^{n_y} a_i' y^2(k-i) + \sum_{i=1}^{n_u} b_i' u^2(k-i) + \sum_{i=1}^{n_\varepsilon} a_i \varepsilon(k-i) \quad (1.33)$$

Nos permite una descripción buena y sencilla de una amplia clase de sistemas no lineales. Por ejemplo, si el modelo exacto NARMAX es:

$$y(k) = y(k-1)e^{-u(k-1)} \quad (1.34)$$

Se puede desarrollar en serie la exponencial

$$e^{-x} = \sum_{i=0}^{\infty} \frac{(-x)^i}{i!} \quad (1.35)$$

y emplear un modelo polinomial aproximado NARMAX

$$y(k) = y(k-1) - y(k-1)u(k-1) + \frac{1}{2}y(k-1)u^2(k-1) - \frac{1}{6}y(k-1)u^3(k-1) \quad (1.36)$$

Se puede comprobar que éste modelo tiene ventajas sobre las series funcionales tales como las series de Volterra.

1.2.4 Modelos No-Paramétricos, Redes Neuronales. Los modelos de Redes Neuronales artificiales han sido estudiados durante muchos años con el ánimo de conseguir prestaciones parecidas a las humanas en campos como el reconocimiento del habla o de imagen.

Estos modelos están compuestos por muchos elementos computacionales trabajando en paralelo y organizados en patrones que recuerdan a las redes

neuronales biológicas. Los elementos computacionales o nodos están conectados mediante pesos que son adaptados a lo largo del proceso para mejorar sus prestaciones. Solamente se considerarán las redes feedforward y las redes recurrentes.

1.2.4.1 Redes Feedforward. Para representar la función en la base $\{g_k\}$ en forma de una red basta escoger $g_k(\phi) = \alpha_k \sigma(\beta_k \phi + \gamma_k)$ donde β_k es un vector de parámetros y γ_k y α_k son parámetros escalares para obtener

$$g(\phi) = \sum_{k=1}^{n} \alpha_k \sigma(\beta_k \phi + \gamma_k) + \alpha_0 \quad (1.37)$$

donde α_0 es el valor correspondiente al nivel medio. A este modelo se le denomina Red Feedforward con una capa oculta y una salida.

Las funciones de la base, llamadas nodos o neuronas, son funciones univaluadas que convierten la red neuronal en un desarrollo en funciones simples.

La elección específica de $\sigma(.)$ es denominada función de activación y suele ser escogida igual para todos los nodos. En la figura 1.2 se observa una de estas redes con dos capas ocultas.

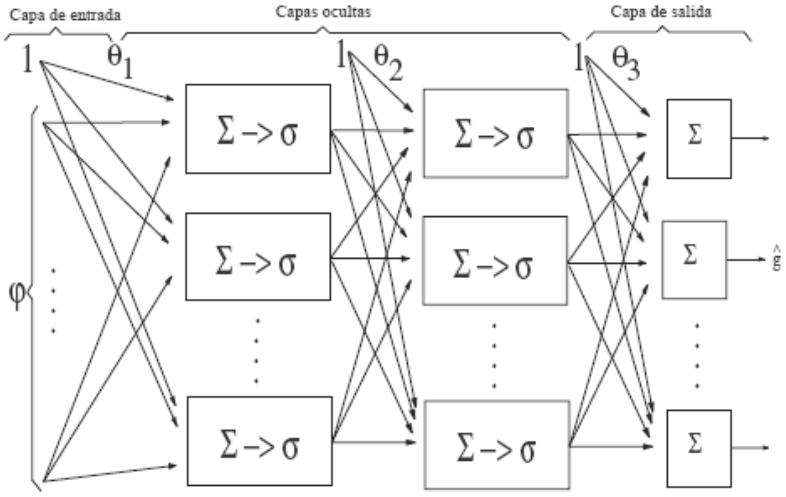

Fig. 1.2. Red Feedforward con dos capas ocultas

1.2.4.2 Redes Recurrentes. Si alguna de las entradas de la red *feedforward* consiste en salidas retardadas de la red, o bien en algún estado interno retardado, la red recibe el nombre de red recurrente o red dinámica. Éste tipo de redes es especialmente interesante para identificación; su estructura se puede apreciar en la figura 1.3.

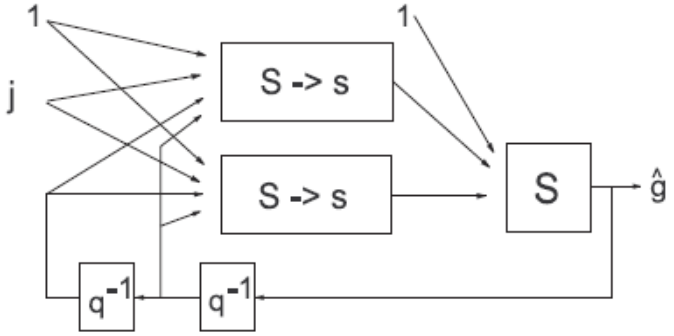

Fig. 1.3. Red recurrente, q^{-1} retarda la señal un tiempo de muestreo.

1.2.5 Identificación con Algoritmos Genéticos. La mayoría de los métodos usuales en la identificación de sistemas están basados en mínimos cuadrados o en el método de máxima verosimilitud que son métodos locales guiados por el gradiente y necesitan una función de pérdidas (loss function) diferenciable y un espacio de búsqueda suave.

A menudo fallan en la búsqueda de óptimos globales cuando éste espacio de búsqueda es no diferenciable, o no lineal en sus parámetros. Los métodos de optimización basados en algoritmos genéticos pueden mejorar los resultados porque no utilizan las propiedades de diferenciabilidad o de ser lineal en los parámetros [15], [16], [17].

1.2.6 Identificación en Lazo Cerrado. La finalidad de la identificación en lazo cerrado es obtener buenos modelos del sistema en lazo abierto, a pesar de la realimentación. Esto debido a que muchos procesos no pueden ser observados en lazo abierto, ya sea por razones de seguridad o por restricciones de operación.

Varios métodos que dan estimaciones consistentes para datos en lazo abierto pueden fallar cuando se aplican de manera directa a la identificación en lazo cerrado. Estos incluyen los análisis espectrales y de correlación, el método de la variable instrumental, los métodos de subespacios y los OE (output error) con modelo de ruido no correlacionado [18].

Consideramos el sistema

$$y(t) = G_0(q)u(t) + H_0(q)e(t) \qquad (1.38)$$

con *e(t)* ruido blanco de varianza λ_0 y el regulador es

$$u(t) = r(t) - F_y(q)y(t) \qquad (1.39)$$

El sistema en lazo cerrado puede ser escrito como:

$$y(t) = G_0(q)S_0(q)r(t) + S_0(q)H_0(q)e(t) \qquad (1.40)$$

Donde $S_0(q)$ es la función de sensibilidad

$$S_0(q) = \frac{1}{1 + F_y(q)G_0(q)} \qquad (1.41)$$

El diagrama de bloques de este sistema se observa en la figura 1.4.

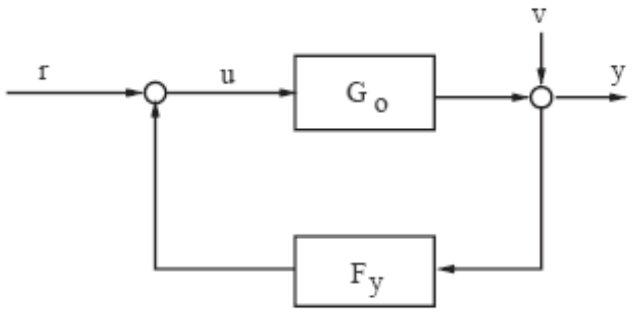

Fig. 1.4. Sistema en lazo cerrado

Es importante saber que el método de error de predicción puede ser aplicado directamente, como si la realimentación no existiera, y el resultado dará una precisión óptima si el sistema verdadero puede ser descrito dentro de la estructura de modelo escogida. Sin embargo, debido a las trampas a que nos puede llevar la identificación en lazo cerrado, han sido sugeridos diversos métodos alternativos.

1.2.7 Modelos fuzzy (fuzzy models). Los modelos fuzzy están basados en descripciones verbales e imprecisas sobre las relaciones entre las señales medidas en un sistema. Gupta y Yamakawa [19] desarrollan todo un estudio sobre modelos fuzzy dividido en tres partes:

- La primera parte presenta algunas ideas sobre incertidumbre cognitiva, percepción y perspectivas en el cálculo cognitivo. También trata sobre aspectos teóricos de los cálculos fuzzy incluyendo algunos estudios importantes sobre lógica fuzzy, visión y percepción, reconocimiento de patrones asociados y modelos de incertidumbres.
- La segunda parte está dirigida a los aspectos de hardware del cálculo fuzzy.
- En la tercera parte presenta varias aplicaciones de lógica fuzzy y cálculo fuzzy. Estos estudios van desde la lógica fuzzy en sistemas expertos hasta electrocardiografía computarizada, pronóstico de diagnósticos médicos, sistemas médicos expertos, manejo de estrés psicosocial y computacional, redes de conocimiento fuzzy, sistemas de soporte de

decisión, control experto para procesos industriales a gran escala, autosintonización de controladores PID, segmentación de imágenes, ingeniería del conocimiento y otros.

1.2.8 Identificación por subespacios. En esta sección se presentará el modelo en espacio de estado y el método de subespacios.

1.2.8.1. Modelos en espacio de estado. Matemáticamente estos modelos son descritos por el siguiente conjunto de ecuaciones en diferencia:

$$x_{k+1} = Ax_k + Bu_k + w_k \qquad (1.42)$$

$$y_k = Cx_k + Du_k + v_k \quad (1.43)$$

Con

$$E[\begin{pmatrix} w_p \\ v_p \end{pmatrix}(w_q^T \qquad v_q^T)] = \begin{pmatrix} Q & S \\ S^T & R \end{pmatrix}\delta_{pq} \geq 0 \qquad (1.44)$$

Donde $E[.]$ es un operador que significa el valor esperado; Q,R,S son matrices definidas positivas y δ representa el operador impulso. En este modelo, tenemos:

Vectores: Los vectores $u_k \in \Re^m$ y $y_k \in \Re^l$ son las mediciones en el instante de tiempo k de las entradas *m* y las salidas l respectivamente del proceso. El

vector $x_k \in \Re^n$ es el vector de estados del proceso en el instante de tiempo discreto k y contiene los valores numéricos de los estados *n*. Estos estados no tienen necesariamente una interpretación física directa pero tienen una relevancia conceptual.

Si los estados del sistema tuvieran un significado físico, siempre se podría encontrar una transformación similar de los modelos en espacio de estado para convertir los estados en significados físicos. $v_k \in \Re^l$ y $w_k \in \Re^n$ son vectores de señal no medibles, se asume que tienen valor medio cero, son estacionarios y tienen un vector de secuencias de ruido blanco.

Matrices: $A \in \Re^{nxn}$ es llamada la matriz del sistema. Describe la dinámica del sistema (completamente caracterizada por sus valores propios). $B \in \Re^{nxm}$ es la matriz de entrada, la cual representa la transformación lineal por la cual las entradas determinísticas influyen en el próximo estado. $C \in \Re^{lxn}$ es la matriz de salida, la cual describe cómo es transferido el estado interno a la salida en las mediciones y_k.

El término con la matriz $D \in \Re^{lxm}$ es llamado el término de realimentación directa. En sistemas en tiempo continuo este término es regularmente cero, lo cual no es el caso en sistemas de tiempo discreto debido al muestreo. Las matrices $Q \in \Re^{nxn}$, $S \in \Re^{nxl}$ y $R \in \Re^{lxl}$ son las matrices de covariancia de las

secuencias de ruido w_k *y* v_k . Se asume que la matriz cuadrada [A, C] es observable, lo cual implica que todos los *modos* en el sistema pueden ser observados en la salida y_k y pueden entonces ser identificados.

Se asume que la matriz cuadrad [A, [B $Q^{1/2}$]] es observable, lo cual a su turno implica que todos los *modos* del sistema son excitados por cada una de las entradas determinísticas u_k y/o entradas estocásticas w_k. Una representación gráfica del sistema se ilustra en la figura 1.5.

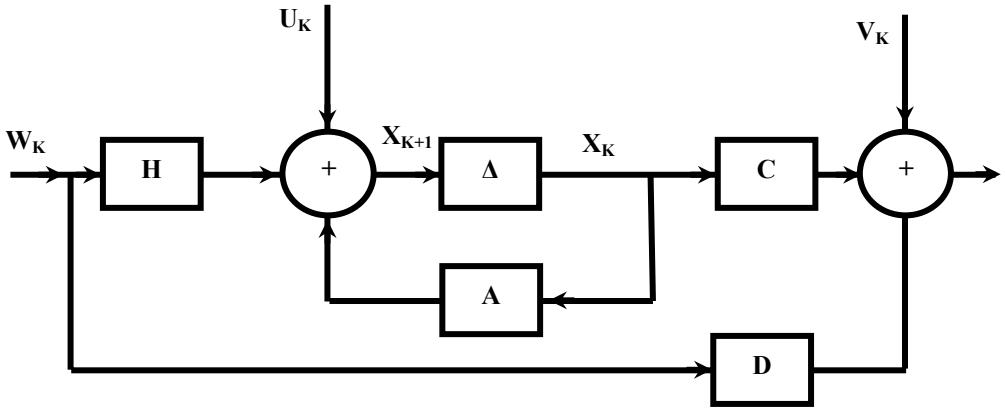

Fig. 1.5. Representación gráfica del modelo en espacio de estado

1.2.8.2 Método de subespacios. Los algoritmos de identificación por subespacios están basados en los conceptos de teoría de sistemas, algebra lineal (numérica) y estadística, el cual es reflejada en la tabla 1.1 que reúne los principales elementos de acuerdo con [20].

Tabla 1.1. Conceptos teóricos para algoritmos de identificación por subespacios.

Sistema	Geometría	Algoritmo
Secuencia de estado de orden alto	Proyección (ortogonal u oblicua)	Descomposición QR
Secuencia de estado de bajo orden	Determina el subespacio dimensional finito	Descomposición del valor singular(Generalizado)
Sistema de matrices	Relaciones lineales	Mínimos cuadrados

La identificación de sistemas por subespacios hace un completo uso de los muy bien desarrollados conceptos y algoritmos del algebra numérica lineal. Mientras que los métodos clásicos están básicamente inspirados en mínimos cuadrados, los métodos por subespacios usan “modernos” algoritmos como el QR – descomposición, descomposición de valores, entre otros.

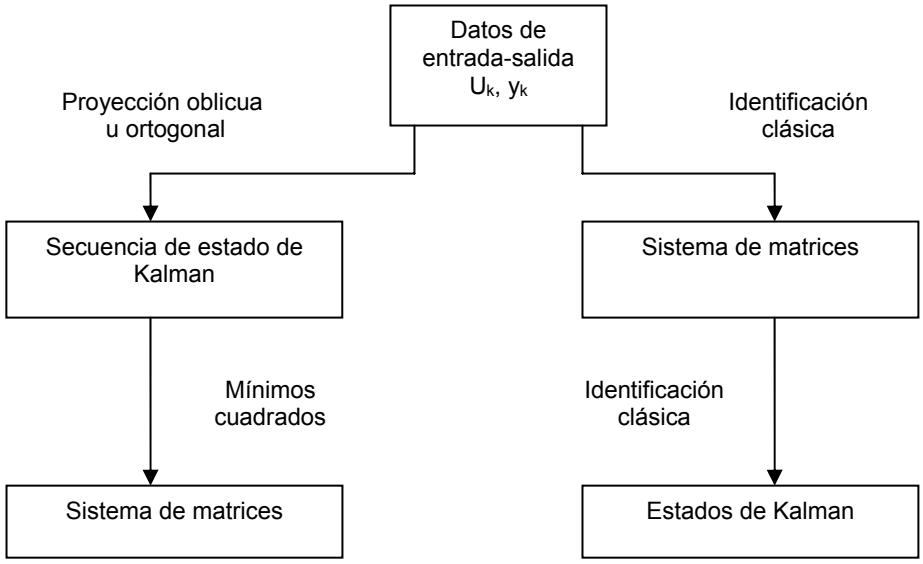

Fig. 1.6. Identificación de sistemas dirigida a la construcción de modelos en espacio de estado a partir de los datos de entrada-salida.

La simplicidad conceptual y algorítmica de los algoritmos de identificación por subespacios representa una ventaja sobre el enfoque clásico del error de predicción. La precisión conceptual de los algoritmos de identificación por subespacios se traslada a la implementación de un software amigable para el usuario.

Algunos autores han desarrollado trabajos sobre las clases de algoritmos de identificación por subespacios [23], algoritmos que parten de las series de tiempo a los sistemas lineales y algoritmos de reducción de modelos [21] y [22], los cuales son un referente para establecer elementos de comparación y diferencias entre sus aplicaciones.

1.2.9 Resumen y conclusión. Hemos descrito en esta sección un resumen de los diferentes modelos y métodos de identificación consultados en la literatura, con sus aspectos principales, lo que permite una ilustración general sobre ellos. Bajo estos conceptos en el capítulo II se determina la estructura seleccionada del modelo y el método de identificación.

1.3 ASPECTOS COMPUTACIONALES

Esta sección hace una descripción de conceptos asociados a los aspectos computacionales de los algoritmos tales como la complejidad computacional. Se define conceptualmente la complejidad computacional, eficiencia

computacional y las clases de complejidad, las cuales servirán como elementos para el análisis de los resultados obtenidos con la implementación de los algoritmos seleccionados.

1.3.1 Complejidad Computacional. La complejidad computacional hace referencia a que tan costoso es encontrar la solución de un problema, desde el punto de vista de recursos computacionales. Fundamentalmente se entiende que estos recursos consisten en tiempo o en memoria de ejecución. En términos computacionales un problema puede ser visto como una pregunta general que debe ser respondida, la cual usualmente contiene varios parámetros, o variables libres, cuyos valores se dejan sin especificar.

1.3.2 Eficiencia y complejidad computacional. En el sentido más amplio, la noción de eficiencia involucra todos los recursos computacionales necesarios para ejecutar un algoritmo. Normalmente se entiende por más eficiente el más rápido; esto señala que los requerimientos de tiempo son un factor dominante que determina si un algoritmo es o no suficientemente eficiente como para ser útil en la práctica.

Un problema puede ser planteado dando: (1) una descripción general de todos sus parámetros, y (2) una sentencia de qué propiedades debe satisfacer la solución o respuesta. Una instancia del problema se obtiene especificando valores particulares para todos los parámetros del problema.

La función de complejidad en tiempo para un algoritmo expresa el requerimiento de tiempo, proporcionando, para cada posible longitud de entrada, la mayor cantidad de tiempo necesitada por el algoritmo para resolver una instancia del problema de ese tamaño.

1.3.3 Clases de complejidad

1.3.3.1 La clase P. Los problemas que pertenecen a la clase **P** son aquellos para los que existe algún algoritmo cuyo tiempo de ejecución crece en forma polinomial respecto al número de variables del problema. Se considera en general que un problema es eficientemente resuelto cuando pertenece a esta clase.

1.3.3.2 La clase NP. Existen algunos problemas para los cuales no se conoce un algoritmo cuya complejidad sea de la clase P. NP significa Nondeterministic polinomial y es común la creencia de que los problemas no son polinomiales, es decir , no existe un algoritmo de la clase P que resuelva el problema.

1.3.4 Complejidad constante, lineal y cuadrática. Los dos criterios para medir la economía en un algoritmo son el tiempo y el espacio. Para medir el tiempo de corrida de un programa o algoritmo es conveniente utilizar una función T(*n*) que represente su complejidad temporal [46], [47]. Sea T(*n*) el tiempo de corrida de algún programa, debemos asumir:

1. El argumento *n* es un entero no negativo, y
2. T(*n*) es no negativo para todos los argumentos de *n*

Los casos más sencillos de complejidad de un algoritmo son constante, lineal y cuadrática.

Complejidad constante. Cuando la complejidad del algoritmo puede expresarse como una función constante T(*n*)=*a*, se dice que es constante, no depende de *n*.

Complejidad lineal: Cuando la complejidad de un algoritmo puede expresarse como una función lineal T(*n*)=*an* + *b*, donde *a* y *b* son constantes se dice que es *lineal* en *n*.

Complejidad Cuadrática: Cuando la complejidad de un algoritmo puede expresarse como una función cuadrática del tipo $T(n)=an^2+bn+c$, donde *a, b* y *c* son constantes, se dice que el algoritmo es cuadrático en *n* [47].

1.3.4.1 Notación O. Para hacer referencia a velocidad de crecimiento de los valores de una función se usa la notación conocida como notación asintótica “o mayúscula”. Por ejemplo, decir que el tiempo de ejecución de un programa es $O(n^3)$, que se lee “o mayúscula de n al cubo” o tan solo “o de n al cubo”,

significa que existen constantes enteras positivas c y n_0 tales que para n mayor o igual a n_0, se tiene que $T(n) \leq cn^3$.

Ejemplo1: Sea T(*n*)=3*n*+2 la complejidad de un algoritmo. Si escogemos *f*(*n*)=*n*, n_0=1, y *c*=5, la condición 0≤T(*n*)≤5*n* se cumple para todos los *n*≥1, por lo que podemos decir que T(*n*) es O(*n*).

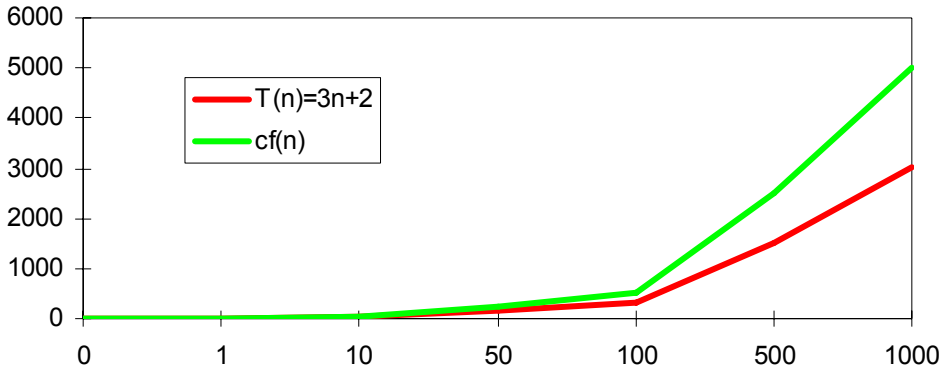

Fig.1.7. Gráfica de las funciones del ejemplo 1.

Cuando se dice que T(*n*) es O(*f*(*n*)), se refiere a que *f*(*n*) es la cota superior de la razón de crecimiento de T(*n*)

1.3.4.2 Notación Ω. La notación Ω(*f*(*n*)) se utiliza para especificar la cota inferior de la razón de crecimiento de T(*n*). Formalmente, decimos que T(*n*) es Ω(*f*(*n*)) si existe un entero n_0 y una constante *c*>0 tal que para todos los enteros *n*≥n_0, tenemos que 0≤*cf*(*n*)≤T(*n*).

Ejemplo 2: Sea T(n) = 3(n log n), entonces T(n) es Ω(log n) ya que podemos escoger n_0=1, f(n)=log n, y c=1. De esta forma, para todos los $n \geq 1$, log $n \leq$ 3(n log n).

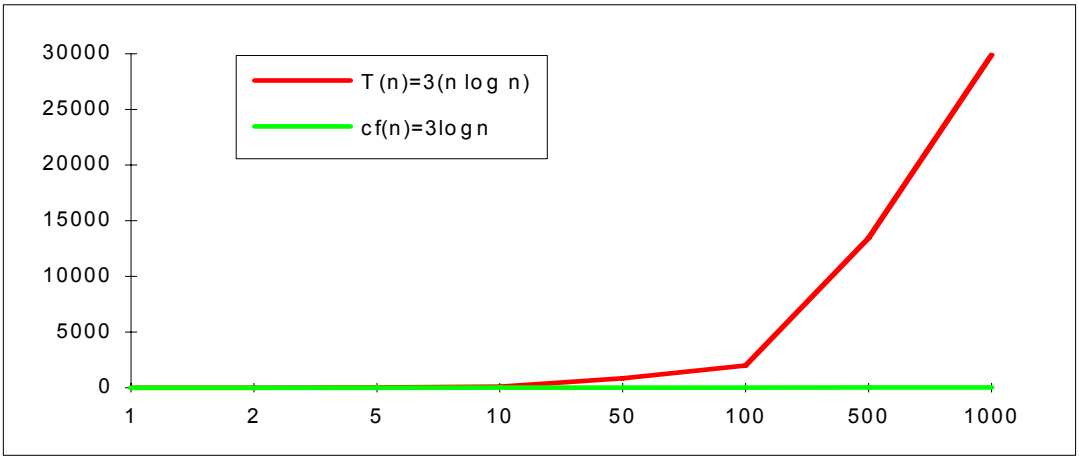

Fig. 1.8. Gráfica de las funciones del ejemplo 2.

Para especificar la cota inferior de la razón de crecimiento de T(n) se usa la notación $\Omega(f(n))$

1.3.4.3 Notación Θ. Formalmente, decimos que T(n) es $\Theta(f(n))$ si existe un entero n_0 y las constantes $c1$ y $c2>0$ tal que para todos los enteros $n \geq n_0$, tenemos que $0 \leq c1f(n) \leq T(n) \leq c2f(n)$.

Ejemplo 3: Sea $T(n)=2n^2$, entonces T(n) es $\Theta(n^2)$ ya que existen n_0=1, c1=1 y c2=3 de manera que para todos los $n \geq 1$, $0 \leq n^2 \leq 2n^2 \leq 3n^2$.

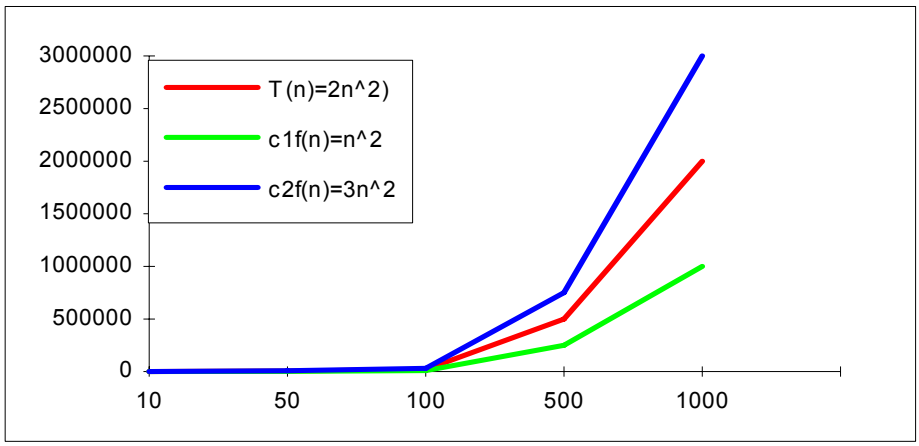

Fig. 1.9. Gráfica de las funciones ejemplo 3.

Los conceptos presentados en esta sección serán utilizados para analizar la complejidad computacional de los algoritmos implementados en este estudio, atendiendo principalmente al orden de la función de trabajo.

CAPITULO II.- IDENTIFICACION PARAMETRICA DE REACTORES DE POLIMERIZACIÓN Y METODOS DE ESTIMACION

Las investigaciones sobre el modelamiento y control para varios reactores de polimerización han sido revisadas en detalle por varios autores. Dentro de la literatura sobre el control de reactores de polimerización se han desarrollado trabajos en modelamiento, estimación de parámetros, control de procesos y monitoreo del desempeño.

En la primera sección de este capítulo se trata el estado del arte en la identificación de reactores para procesos de polimerización. En la segunda sección se describe el modelo de predicción para la estructura ARX y en la tercera sección se presentan los métodos de estimación de parámetros utilizados en este trabajo.

2.1 ESTADO DEL ARTE EN IDENTIFICACION DE REACTORES DE POLIMERIZACION

La mayoría de los reactores de polimerización continuos que se han estudiado en la literatura han sido modelados como un CSTR. Los modelos que han sido desarrollados necesitan ser validados antes de que sean usados para el monitoreo, control y optimización de procesos de polimerización. En [4] Choi identifica las complejas estructuras en estado estable dentro de un CSTR en un

proceso de polimerización del metilmetacrilato; en su trabajo concluye que las estructuras en estado estable predichas por el modelo simplificado y el modelo original son muy similares tanto cualitativa como cuantitativamente.

En [7] Wissner propone un modelo matemático de un CSTR de un tren de reactores que le permiten diseñar un esquema para el control de conversión del estireno butadieno, mediante la manipulación de la temperatura del reactor; para evaluar el esquema de control se apoya en la simulación por computador. En [24], [27] y [34], los autores desarrollan trabajos de investigación sobre los reactores de polimerización, donde se proponen diferentes modelos matemáticos para el control, control avanzado y control predictivo.

Embiricu et al en [26] presentan un amplio desarrollo sobre el control avanzado de los reactores de polimerización, partiendo de modelos paramétricos en su simulación.

Kiparissides en [25] hace un estudio muy detallado sobre el modelamiento de los reactores de polimerización; señala lo complejo de las reacciones químicas dentro de los procesos de polimerización y la dificultad que representa poder medir todas las variables químicas que intervienen en el proceso. A partir de ese análisis resalta la importante necesidad por desarrollar modelos matemáticos comprensivos con una alta capacidad de predecir las propiedades moleculares y morfológicas en términos de la configuración del reactor y las

condiciones de operación, que puedan ser simulados eficientemente con herramientas computacionales adecuadas.

Dentro de esta gama los modelos paramétricos pueden ser generados por la estimación de los parámetros usando información y datos disponibles del proceso. Esta estimación de parámetros puede ser obtenida por el uso de mínimos cuadrados, mínimos cuadrados ponderados, máxima probabilidad o estimación Bayesiana. En [28], Bard presenta un tratado bastante amplio sobre la estimación de parámetros de sistemas no lineales. De igual forma Stewart et al en [29] hacen una descripción de la estimación de parámetros para sistemas de los que puede lograrse una gran cantidad de datos independientemente de las salidas del sistema.

La mayoría de los investigadores en la literatura sobre control de procesos de polimerización han hecho uso de los modelos de simulación de tal forma que no es necesario estimar los parámetros cinéticos después que el modelo es desarrollado y simulado. Adicionalmente, estos investigadores han desarrollado algunas clasificaciones de modelos de plantas para la demostración de sus estrategias de control.

De otra parte también se han planteado algunos trabajos mediante el uso de conjuntos de datos de experimentos de laboratorio monitoreados cuidadosamente que no pueden mostrar el comportamiento típico de un

proceso industrial a gran escala. Los parámetros desconocidos también han sido obtenidos al proveer las condiciones experimentales apropiadas para facilitar su estimación usando regresión lineal [49].

En [31] Sawattanakit y Jaovisidha, presentan un trabajo sobre el diagnóstico y detección de fallas en un CSTR usando un extimador en línea. El modelo descrito en este trabajo plantea un grupo de ecuaciones adimensionales del modelo para las simulaciones del proceso. Este modelo permite detectar y diagnosticar fallas del proceso durante el periodo transitorio.

En la literatura consultada se presentan varios estudios sobre el modelamiento y control de reactores de polimerización, construidos a partir del análisis de datos en línea o fuera de línea. Entre los reactores estudiados se destacan los CSTR y los reactores tipo batch; para ampliar la información el lector puede consultar las referencias [35], [36], [37] y [38].

La estimación de parámetros en línea usando el método de mínimos cuadrados ponderados ha sido comparada a un filtro extendido de Kalman usando diferentes modelos de simulación para la planta y el modelo del proceso. Los parámetros estimados han sido reportados de diferentes condiciones experimentales para un proceso de polimerización. Los parámetros estimados obtenidos por los investigadores en procesos de polimerización han sido

mayormente obtenidos por la minimización de los mínimos cuadrado ponderados (weighted least-squares) de mediciones de error en las salidas.

Anderson et al. en [32] presentan un trabajo donde se requiere de valores cercanos al mínimo global de la suma de las desviaciones cuadradas para la estimación de parámetros de manera sucesiva. En vista a que la búsqueda de valores iniciales requiere de un tiempo considerable y disminuye la efectividad del procedimiento total, los autores proponen un algoritmo de búsqueda automática de los valores para la optimización de la energía de activación y el factor de frecuencia de una reacción en un proceso de polimerización.

Sirohi y Choi en [33] proponen un enfoque simultáneo para el problema de estimación de parámetros. Este enfoque es particularmente eficiente en el manejo de problemas de regresión avanzada aplicada a procesos de polimerización como la formulación de los errores en las variables medidas. La metodología es robusta, rápida y confiable por lo que puede ser aplicada para procesos en línea o fuera de línea.

En [39], Arora y Biegler, presentan un estudio sobre la estimación de parámetros de un reactor de polimerización. El trabajo está enfocado en la robustez del algoritmo para solucionar los problemas de optimización condicionada de modelos con ecuaciones diferenciales algebraicas. Para problemas no lineales un eficiente y confiable algoritmo de estimación de

parámetros es esencial para la selección del procedimiento de medición. El estudio considera además el problema del error en las variables del modelo y compara los resultados al problema de la estimación de parámetros estándar.

Los reactores químicos se clasifican en tres tipos básicos que son: reactor discontinuo (Batch), reactor tubular o de flujo pistón (PFR) y reactor de tanque agitado (CSTR) [52]. El primer tipo de reactor es utilizado para procesos no continuos o en donde se produzcan pequeñas cantidades de sustancias ó para la producción de muchas sustancias diferentes en el mismo aparato. Su operación es ineficiente debido a la pérdida de tiempo en la carga y descarga de este.

Los tres tipos de reactores pueden ser usados en el proceso de polimerización del estireno. La mayoría de los procesos productivos existentes utilizan uno o una combinación de reactores tipo CSTR ó tipo tubular (PFR) [53].

La diferencia fundamental entre los reactores tipo CSTR y los reactores tubulares es que en el primer tipo al ser un tanque completamente agitado, la concentración de los reactivos es constante en toda la geometría del reactor, mientras que en el reactor tubular la concentración varía a medida que se avanza en el reactor en el sentido longitudinal del tubo.

Matemáticamente un reactor tubular puede ser representado por N- reactores CSTR en serie. A medida que se aumenta el número de reactores CSTR en serie más se asemeja a un comportamiento de reactor tubular, ver [52] página 149.

En [54] se describe un estudio sobre las diferentes formas de modelar un reactor tubular ó de flujo pistón para una reacción de polimerización en emulsión y se muestra que una de esas formas es el uso de un modelo dinámico de reactores CSTR en serie.

Para nuestro estudio se escogió un reactor tipo CSTR debido a que es el reactor en el que se lleva a cabo la mayor parte de la conversión del estireno y porque además, los resultados encontrados en este estudio pueden ser aplicados para un estudio posterior en reactores tubulares tratando estos reactores como varios CSTR en serie.

De acuerdo con las referencias consultadas para la estimación paramétrica de procesos de polimerización con CSTR, se escogió el modelo ARX (explicado en el capítulo I) como un modelo lineal de referencia y los métodos de mínimos cuadrados y mínimos cuadrados ponderados, para desarrollar la posterior implementación y análisis de los algoritmos aplicados a un tipo de reactor CSTR.

2.2 MODELO DE PREDICCIÓN PARA LA ESTRUCTURA ARX

El problema de identificar un sistema dinámico se reduce, una vez obtenidos el conjunto de datos de entrada $u(t)$ y salida $y(t)$, a obtener los coeficientes $a_1,...,a_n,b_1,...,b_m$ de la función de (1.2). Para calcular el valor numérico de dichos coeficientes y dado que el conjunto de datos experimentales $[u(t), y(t)]$ se encuentran en el dominio del tiempo, para realizar el procedimiento de identificación se utilizará (1.3).

Si suponemos que empleamos tan solo el valor de las dos últimas mediciones de las entradas y salidas para predecir la conducta del sistema en el momento actual y(t)

$$y(t) = -a_1 y(t-1) - a_2 y(t-2) + b_1 u(t-1) + b_2 u(t-2) \quad (2.1)$$

Supongamos además que disponemos de N mediciones experimentales. Por lo tanto, escribiendo (2.1) para cada una de las 1,2, … , *N* mediciones

$$y(1) = -a_1 y(0) - a_2 y(-1) + b_1 u(0) + b_2 u(-1)$$
$$y(2) = -a_1 y(1) - a_2 y(0) + b_1 u(1) + b_2 u(0)$$
$$\vdots$$
$$y(N) = -a_1 y(N-1) - a_2 y(N-2) + b_1 u(N-1) + b_2 u(N-2)$$

Este sistema de ecuaciones se puede escribir en forma matricial

$$Y = \phi\theta \quad (2.2)$$

Donde

$$Y = \begin{bmatrix} y(1) \\ y(2) \\ . \\ . \\ . \\ y(2 \\ y(N) \end{bmatrix}, \qquad \phi = \begin{bmatrix} y(0) & y(-1) & u(0) & u(-1) \\ y(1) & y(0) & u(1) & u(0) \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ y(N-1) & y(N-2) & u(N-1) & u(N-2) \end{bmatrix}, \qquad \theta = \begin{bmatrix} -a_1 \\ -a_2 \\ b_1 \\ b_2 \end{bmatrix}$$

2.2.1 Selección de la estructura y criterios de comparación. Un paso previo a la estimación de los parámetros de un modelo, es seleccionar cual va a ser la estructura de dicho modelo. Para este caso particular como se definió se escogió la estructura ARX, descrita a través de la ecuación diferencial:

$$y(t) + a_1 y(t-1) + ... + a_{na} y(t - na) = b_1 u(t - nk) + ... + b_{nb} u(t - nk - nb + 1) \quad (2.3)$$

La cual relaciona la salida actual *y(t)* a un número finito de salidas *y(t-k)* y entradas *u(t-k)* pasadas.

2.2.1.1 Primer criterio de comparación. Como se deben seguir algunos criterios para discernir cual es la mejor estructura para el modelo de un sistema, se escogió como posible primer criterio seleccionar aquella estructura que presente el FPE (Akaike's Final Prediction Error) más pequeño [41]. El FPE está dado por la expresión:

$$FPE = \min_{d,\theta} \left(\frac{1+\frac{d}{N}}{1-\frac{d}{N}} * \frac{1}{N} * \sum_{t=1}^{N} \in^2 (t,\theta) \right) \quad (2.4)$$

Donde

d: es el número total de parámetros estimados

N: Es la longitud del número de datos muestreados

ε : es la función del error entre el valor real y el proyectado

$$\varepsilon = \left| y(t) - \hat{y}(t \mid \theta) \right|^2 \quad (2.5)$$

Este criterio permite buscar la mejor estructura, penalizando la utilización de un número *d* excesivo de parámetros. Se ha de pensar que la utilización de muchos parámetros no es sinónima de tener un mejor modelo, muchas veces se está complicando excesivamente el modelo sin producir una mejora apreciable en su comportamiento.

2.2.1.2 Segundo criterio de comparación. El segundo criterio de comparación está relacionado con el tiempo de ejecución de cada algoritmo, para las mismas condiciones de datos de entrada en cuanto al número de datos y al orden del sistema seleccionado. En estos términos, se hizo un análisis de la eficiencia de los algoritmos y a la vez lo hemos correlacionado con el mejor modelo posible dentro de los resultados simulados, para conocer su desempeño.

2.3 ESTIMACIÓN POR MÍNIMOS CUADRADOS

Lo que se tiene en lugar de datos históricos es una colección de observaciones previas de valores relacionados de y y φ. Es conveniente enumerar estos valores usando un argumento t:

$$y(t), \varphi(t), \qquad t = 1.....N \quad (2.6)$$

Siendo N el número de muestras, con los datos históricos se podría remplazar la variancia por la variancia de muestras

$$\frac{1}{N}\sum_{t=1}^{N}[y(t) - g(\varphi(t))]^2$$

En el caso lineal, se tiene entonces que

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N}[y(t) - \varphi^T(t)\theta]^2 \quad (2.7)$$

Donde $V_N(\theta)$ es la función criterio a ser minimizada. Un conveniente valor de θ (parámetros) a seleccionar es la minimización del argumento de (2.7):

$$\hat{\theta}_N = \arg\min V_N(\theta) \quad (2.8)$$

La única característica de (2.7) es que es una función cuadrática de θ. De allí que pueda ser minimizada analíticamente. Encontramos que todos los $\hat{\theta}_N$ que satisfacen

$$\left[\frac{1}{N}\sum_{t=1}^{N}\varphi(t)\varphi^T(t)\right]\theta_N = \frac{1}{N}\sum_{t=1}^{\hat{N}}\varphi(t)y(t) \quad (2.9)$$

Produciendo el mínimo global de $V_N(\theta)$. Este conjunto de ecuaciones lineales es conocido como las *ecuaciones normales* [18]. Si la matriz del lado izquierdo de (2.9) es invertible, entonces tenemos el Estimador de Mínimos Cuadrados (LSE)

$$\theta_N = \left[\frac{1}{N}\sum_{t=1}^{N}\varphi(t)\varphi^T(t)\right]^{-1}\frac{1}{N}\sum_{t=1}^{N}\varphi(t)y(t) \quad (2.10)$$

Se puede plantear la formulación matricial de la estimación de mínimos cuadrados. Las ecuaciones (2.6) a (2.10) pueden ser escritas en forma matricial. Definimos el vector columna Nx1

$$Y_N = \begin{bmatrix} y(1) \\ . \\ . \\ . \\ y(N) \end{bmatrix} \quad (2.11)$$

Donde los y(k) representan los datos, y la matriz Nxd

$$\phi_N = \begin{bmatrix} \varphi^T(1) \\ . \\ . \\ . \\ \varphi^T(N) \end{bmatrix} \quad (2.12)$$

Entonces la ecuación (2.4) puede ser escrita como

$$V_N(\theta) = \frac{1}{N} | Y_N - \phi_N \theta |^2 = \frac{1}{N} (Y_N - \phi_N \theta)^T (Y_N - \phi_N \theta) \quad (2.13)$$

La ecuación normal toma la forma

$$[\phi_N^T \phi_N] \hat{\theta}_N = \phi_N^T Y_N \quad (2.14)$$

Y el estimador

$$\hat{\theta}_N = [\phi_N^T \phi_N]^{-1} \phi_N^T Y_N \quad (2.15)$$

En la ecuación (2.15) se puede reconocer la pseudoinversa de ϕ_N :

$$\phi_N^{\div} = [\phi_N^T \phi_N]^{-1} \phi_N^T \qquad (2.16)$$

La ecuación (2.15) da la solución a la pseudoinversa a las ecuaciones lineales del sistema sobredeterminado (N>d)

$$Y_N = \phi_N \theta \quad (2.17)$$

2.4 MÉTODO DE MÍNIMOS CUADRADOS PONDERADOS

En la estimación de parámetros por el método de mínimos cuadrados ponderados, como regularmente se da en el método de mínimos cuadrados, los valores de los parámetros desconocidos, θ_0, θ_1,…, en la función de regresión son estimados al encontrar los valores numéricos de los parámetros que minimizan la suma de las desviaciones cuadradas entre las respuestas observadas y la porción funcional del modelo.

A diferencia de los mínimos cuadrados, sin embargo, cada término del criterio de mínimos cuadrados ponderados incluye un peso adicional, w_i, que

determina que tanto influye cada observación en el conjunto de datos de los parámetros estimados. El criterio de mínimos cuadrados ponderados que es minimizado para obtener los parámetros estimados es:

$$Q = \sum_{i=1}^{n} w_i [y_i - f(x_i; \theta)]^2 \quad (2.18)$$

La minimización de los mínimos cuadrados ponderados es usualmente hecha de manera analítica para modelos lineales y por métodos numéricos para modelos no lineales.

En un modelo de regresión lineal se supone que la matriz de varianzas-covarianzas de los errores es de la forma

$$E[\vec{\varepsilon}\,\vec{\varepsilon}^{\,t}] = \sigma^2 I_n \quad (2.19)$$

Siendo $\boldsymbol{I_n}$ la matriz identidad de orden *n* y $\boldsymbol{\sigma}$ la desviación estándar. Si no se verifica la hipótesis de homocedasticidad (esto es, la varianza de los residuos es constante y no varía en los diferentes niveles del factor), o la de independencia (la hipótesis de que las observaciones muestrales son independientes es una hipótesis bajo la cual los errores $\{\varepsilon_i\}_{j=1}$ son variables aleatorias independientes), o ambas, entonces la matriz de varianzas-covarianzas tiene la forma general

$$E[\vec{\varepsilon}\,\vec{\varepsilon}^{\,t}] = \sigma^2\psi \quad (2.20)$$

Siendo ψ una matriz simétrica, definida positiva de orden *n* × *n*. En este caso, se puede calcular el estimador de $\vec{\alpha}$ por el método de mínimos cuadrados generalizados**.** Este método se desarrolla en dos etapas: en una primera etapa se transforma el modelo de regresión original

$$\vec{Y} = X\vec{\alpha} + \vec{\varepsilon} \quad (2.21)$$

Para ello y por ser ψ una matriz simétrica, definida positiva, existe una matriz cuadrada P tal que

$$P\psi P^t = I_n \Rightarrow \psi = P^{-1}(P^t)^{-1} \Rightarrow \psi^{-1} = P^t P \quad (2.22)$$

esta matriz ψ no tiene porque ser única, pero si existe. Multiplicando por P la ecuación de regresión se obtiene

$$P\vec{Y} = PX\vec{\alpha} + P\vec{\varepsilon} \quad (2.23)$$

Denominando $\vec{Y}^{\,*} = P\vec{Y}, X^* = PX$ y $\vec{\varepsilon}^{\,*} = P\vec{\varepsilon}$, se obtiene la ecuación de regresión

$$\vec{Y}^{\,*} = X^*\vec{\alpha} + \vec{\varepsilon}^{\,*} \quad (2.24)$$

y los errores del modelo verifican

$$E[\vec{\varepsilon}^{*}\vec{\varepsilon}^{*t}] = PE[\vec{\varepsilon}\vec{\varepsilon}^{t}]P^{t} = \sigma^{2}P\psi P^{t} = \sigma^{2}I_{n} \quad (2.25)$$

por tanto los errores son incorrelados y homocedásticos. Ahora se puede aplicar el método de mínimos cuadrados ordinarios a estos datos transformados $(PX, P\vec{Y})$ para obtener el estimador

$$\hat{\alpha}_{G} = \left((PX)^{t}PX\right)^{-1}(PX)^{t}PY = (X^{t}P^{t}PX)^{-1}X^{t}P^{t}PY = (X^{t}\psi^{-1}X)^{-1}X^{t}\psi^{-1}Y \quad (2.26)$$

Por el Teorema de Gauss-Markov, ver [18], este estimador $\hat{\alpha}_{G}$ es el mejor estimador lineal no sesgado. En la práctica, la matriz P, aunque existe, es desconocida y es necesario estimarla $\hat{P}$ a partir de las observaciones, obteniendo el estimador

$$\hat{\alpha}_{F} = (X^{t}\hat{P}^{t}\hat{P}X)^{-1}X^{t}\hat{P}^{t}\hat{P}Y = (X^{t}\hat{\psi}^{-1}X)^{-1}X^{t}\hat{\psi}^{-1}Y \quad (2.27)$$

A continuación se exponen dos situaciones comunes en las que se puede aplicar este método de estimación.

2.4.1 Heterocedasticidad. La heterocedasticidad es la existencia de una varianza no constante en las perturbaciones aleatorias de un modelo [51]. Si

las observaciones son independientes pero heterocedásticas entonces la matriz de varianzas-covarianzas viene dada por

$$E[\vec{\varepsilon}\,\vec{\varepsilon}^{\,t}] = \begin{pmatrix} \sigma_1^2 & 0 & . & . & . & 0 \\ 0 & \sigma_2^2 & . & . & . & . \\ . & 0 & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ 0 & 0 & . & . & . & \sigma_n^2 \end{pmatrix}$$

Y la matriz P

$$P = \begin{pmatrix} \frac{1}{\sigma_1} & 0 & . & . & . & 0 \\ 0 & \frac{1}{\sigma_2} & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ 0 & 0 & . & . & . & \frac{1}{\sigma_n} \end{pmatrix}$$

En este caso los datos transformados son

$$\vec{Y}^{*} = P\vec{Y} = \begin{pmatrix} \frac{1}{\sigma_1} & 0 & . & . & . & 0 \\ 0 & \frac{1}{\sigma_2} & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ 0 & 0 & . & . & . & \frac{1}{\sigma_n} \end{pmatrix} \begin{pmatrix} Y_1 \\ . \\ . \\ . \\ . \\ Y_n \end{pmatrix} = \begin{pmatrix} \frac{Y_1}{\sigma_1} \\ . \\ . \\ . \\ . \\ \frac{Y_n}{\sigma_n} \end{pmatrix}$$

$$X^{*} = PX = \begin{pmatrix} \frac{1}{\sigma_1} & 0 & . & . & . & 0 \\ 0 & \frac{1}{\sigma_2} & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ 0 & 0 & . & . & . & \frac{1}{\sigma_n} \end{pmatrix} \begin{pmatrix} \vec{X}_1 \\ . \\ . \\ . \\ . \\ \vec{X}_n \end{pmatrix} = \begin{pmatrix} \frac{\vec{X}_1}{\sigma_1} \\ . \\ . \\ . \\ . \\ \frac{\vec{X}_n}{\sigma_n} \end{pmatrix}$$

Esto equivale a trabajar con el modelo transformado

$$\frac{Y_i}{\sigma_i} = \frac{\vec{x}_i}{\sigma_i}\vec{\alpha} + \frac{\varepsilon_i}{\sigma_i}, \qquad i = 1,...,n. \quad (2.28)$$

Sobre este modelo se aplica ahora el método de mínimos cuadrados ordinarios. En particular, si se trabaja con el modelo de regresión lineal se obtiene el siguiente estimador del coeficiente de regresión (α_1) [49].

$$\hat{\alpha}_{1,G} = \frac{\sum_{i=1}^{n} \frac{1}{\sigma_i^2}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} \frac{1}{\sigma_i^2}(x_i - \bar{x})} \quad (2.29)$$

Este estimador se denomina estimador por mínimos cuadrados ponderados y es un caso particular del estimador por mínimos cuadrados generalizados. En

la práctica, para utilizar este estimador hay que calcular estimadores de los parámetros $\sigma_1^2,...,\sigma_n^2,$ lo que puede hacerse por uno de los siguientes métodos:

- Suponer que la varianza se ajusta a una función

$$\sigma_i^2 = g(x_i), \qquad i = 1,..., n \qquad (2.30)$$

y estimar la función g .

- Hacer grupos en las observaciones (en el orden en que se han recogido) normalmente del mismo tamaño *k* y suponer que en cada grupo la varianza es constante. Entonces se estima la varianza en cada grupo a partir de las observaciones del grupo. una forma de conseguir esto es ajustar el modelo de regresión por mínimos cuadrados ordinarios a las observaciones originales y a partir de los residuos de este modelo obtener los estimadores de la varianza en cada grupo.

2.4.2 Observaciones dependientes. Si las observaciones son homocedásticas pero dependientes entonces la matriz de varianzas-covarianzas es de la forma general

$$E[\vec{\varepsilon}\,\vec{\varepsilon}^{\,t}] = \sigma^2 \begin{pmatrix} 1 & \rho_1 & \cdot & \cdot & \cdot & \rho_{n-1} \\ \rho_1 & 1 & \cdot & \cdot & \cdot & \rho_{n-2} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \rho_{n-1} & \rho_{n-2} & \cdot & \cdot & \cdot & 1 \end{pmatrix}$$

En la mayoría de las situaciones la estructura de dependencia de los errores puede ajustarse a un modelo paramétrico. Un modelo sencillo y muy utilizado es el modelo $AR^{(1)}$, (modelo autorregresivo de orden uno). En este caso se verifica que los errores siguen la ecuación

$$\varepsilon_i = \rho\varepsilon_{i-1} + a_i, \qquad i = 1,...,n. \quad (2.31)$$

siendo ρ la autocorrelación de orden 1 del proceso ε_i por tanto, $|\rho| < 1$, y a_i es una sucesión de variables aleatorias independientes e igualmente distribuídas.

En este caso, la matriz de varianzas-covarianzas es

$$E[\vec{\varepsilon}\,\vec{\varepsilon}^{\,t}] = \sigma^2\psi = \sigma^2 \frac{1}{1-\rho^2} \begin{pmatrix} 1 & \rho & \cdot & \cdot & \cdot & \rho^{n-1} \\ \rho & 1 & \cdot & \cdot & \cdot & \rho^{n-2} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \rho^{n-1} & \rho^{n-2} & \cdot & \cdot & \cdot & 1 \end{pmatrix}$$

la matriz P de transformación es

$$p = \begin{pmatrix} \sqrt{1-\rho^2} & 0 & 0 & . & 0 & 0 \\ -\rho & 1 & . & . & 0 & 0 \\ 0 & -\rho & 1 & . & 0 & 0 \\ . & . & . & . & . & . \\ 0 & 0 & 0 & . & 1 & 0 \\ 0 & 0 & 0 & . & -\rho & 1 \end{pmatrix}$$

y la matriz ψ^{-1} es

$$\psi^{-1} = \begin{pmatrix} 1 & -\rho & 0 & ... & 0 & 0 \\ -\rho & 1+\rho^2 & -\rho & ... & 0 & 0 \\ 0 & -\rho & 1+\rho^2 & ... & 0 & 0 \\ . & . & . & . & . & . \\ 0 & 0 & 0 & . & 1+\rho^2 & -\rho \\ 0 & 0 & 0 & . & -\rho & 1 \end{pmatrix}$$

Utilizando esta matriz se obtiene el estimador por mínimos cuadrados generalizados

$$\hat{\alpha}_G = (X^t \psi^{-1} X)^{-1} X^t \psi^{-1} Y \quad (2.32)$$

Donde X corresponde a la matriz de observación, ψ^{-1} es la matriz de ponderación y Y es el vector de datos del experimento.

Nuevamente, en la práctica, ψ_1 es desconocido y se tiene que estimar. Por la forma de la matriz ψ^{-1}, es suficiente con estimar el parámetro ρ y sustituir en la matriz. Para estimar ρ, puede utilizarse el siguiente procedimiento: ajustar a los datos el modelo de regresión lineal por mínimos cuadrados ordinarios y calcular los residuos mínimo cuadráticos

$$e_i = Y_i - \vec{x}_i \hat{\alpha}_{MCO}, \qquad i = 1,...,n. \quad (2.33)$$

A partir de estos residuos se obtiene el siguiente estimador de ρ,

$$\hat{\rho} = \frac{\sum_{i=1}^{n-1} e_i e_{i+1}}{\sum_{i=1}^{n} e_i^2} \quad (2.34)$$

sustituyendo ρ por $\hat{\rho}$ en la matriz ψ^{-1} se obtiene la matriz estimada $\hat{\psi}^{-1}$, a partir de la cual se obtiene el estimador

$$\hat{\alpha}_F = (X^t \hat{\psi}^{-1} X) X^t \hat{\psi}^{-1} Y \quad (2.35)$$

Siguiendo este procedimiento se puede obtener el siguiente estimador iterativo:

Paso 1. Se utiliza el estimador $\hat{\alpha}_F$ para obtener nuevos residuos e_i'.

Paso 2. De estos residuos se obtiene un nuevo estimador $\hat{\rho}'$

Paso 3. Utilizando $\hat{\rho}$ se calcula un nuevo estimador α_F'

Se continúa el proceso de forma iterativa (volver al Paso 1) hasta obtener la convergencia del estimador $\hat{\alpha}_F$.

En este problema también se pueden considerar otros estimadores del parámetro ρo modelos de dependencia más complejos que dependen de un número mayor de parámetros [49].

En este capítulo se presentaron los diferentes métodos de identificación de sistemas y los métodos de estimación de parámetros de interés para este trabajo.

CAPITULO III.- IMPLEMENTACIÓN DE LOS ALGORITMOS

Los algoritmos a implementar que fueron seleccionados son el de mínimos cuadrados y el de mínimos cuadrados ponderados. Estos métodos fueron ampliamente desarrollados en el capítulo II de esta tesis, y son los métodos mas utilizados para el caso particular de la literatura consultada sobre reactores químicos, tal como se indicó en el anterior capítulo.

En la primera sección de este capítulo se presentan elementos de la herramienta computacional utilizada. En la segunda y tercera sección se describen cada uno de los algoritmos seleccionados para su implementación

3.1 HERRAMIENTA COMPUTACIONAL UTILIZADA

Dada la gran versatilidad y manejo de funciones, se escogió Matlab (versión 6.5) como herramienta para el desarrollo e implementación de los algoritmos. Matlab es un programa orientado a la implementación de algoritmos numéricos, con rutinas eficientes de manejo de operaciones matemáticas utilizando matrices; se considera un lenguaje de alto nivel en un ambiente interactivo que permite la implementación de desarrollos con alto requerimiento computacional. Adicionalmente, Matlab dispone de un gran número de funciones repartidas en diferentes “toolboxes” para la identificación y optimización de funciones. En la Tabla 3.1 se hace un breve resumen de estas funciones.

Tabla 3.1 Resumen de funciones de Matlab usadas en la implementación de los algoritmos

Nombre de la función	Descripción
detrend	Remueve la tendencias de los datos y hace el valor medio igual cero
clock	Señala la fecha y hora del procesador en un vector fila
simout	Almacena los datos de entrada y salida de un experimento
zeros	Construye una matriz de ceros
poly2th	Crea una estructura del modelo para modelos de entrada salida definido como un polinomio con numerador y denominador
idsim	Simula un modelo para su validación
d2c	Convierte un modelo discreto en un modelo continuo
idplot	Muestra los datos de entrada y salida
inv	Extrae la inversa de una matriz
sim	Simula un modelo construido en simulink usando los parámetros de simulación
size	Retorna la longitud de un arreglo (matriz)
zeros	Genera una matriz de ceros
inv	Retorna la inversa de una matriz cuadrada
sum	Suma los elementos de una matriz o de un vector
ones	Retorna una matriz de unos de orden especificado por el usuario
tril	Genera una matriz triangular inferior
triu	Genera una matriz triangular superior
eyes	Genera una matriz identidad de orden especificado
plot	Genera una gráfica de un conjunto de datos

Como ejemplo, la función ***detrend*,** es una función que permite remover tendencias dentro de un conjunto de datos muestreados, de tal manera que hace el valor medio de estos datos igual a cero, como se indica en la tabla 3.1. Esta es una función muy utilizada para el tratamiento de datos muestreados de un proceso. La estructura de la instrucción viene dada por:

y=detrend(x)

donde x representa el vector de datos muestreados y el vector que retorna los datos previo tratamiento señalado en el párrafo anterior.

Además de las funciones descritas, Matlab ofrece un entorno gráfico para realizar desarrollos de estimación de parámetros e identificación de sistemas. Permite trabajar con funciones de transferencia o con representación en variables de estado en el dominio temporal o en frecuencia. Tiene la posibilidad de trabajar con programas en lenguaje C; permite construir modelos ARX multivariables y modelos de variables de estado con parámetros de acoplo; el lector puede ampliar la información sobre esta herramienta en la referencia [40].

El toolbox utilizado en la implementación de estos algoritmos es el simulink. El simulink es una herramienta gráfica de matlab que contiene todos los elementos básicos que se necesitan para la construcción en diagrama de

bloques de un modelo. Posee una librería con bloques de operaciones matemáticas básicas, swiches (conmutadores), conectores, elementos de simulación en control, entre otros.

El Simulink de matlab utiliza ODE45 para resolver las ecuaciones diferenciales no lineales. ODE45 está basado en una fórmula explícita de Runge-Kutta (4,5) y Dormand-Prince. Es un solucionador de un solo paso; en el cálculo de $y(t_n)$, solamente necesita la solución en el punto de tiempo precedente inmediatamente anterior, $y(t_n - 1)$ ver [40], [50].

3.2 ALGORITMO DE MÍNIMOS CUADRADOS

En esta sección se detalla la implementación del algoritmo de mínimos cuadrados. El algoritmo es el siguiente:

- Selección de datos para construcción del modelo.

  Del total de datos de entrada se escoge la primera mitad de ellos para la construcción del modelo como los datos experimentales.

- Tratamiento de señal de los datos seleccionados.

  Mediante el uso de funciones de Matlab, se hace un tratamiento de señal de los datos seleccionados, removiendo los niveles constantes de la entrada y la salida, haciendo el valor medio igual a cero.

- Construcción de la matriz de parámetros.

Se determinan las dimensiones de la matriz de entrada y de la matriz de salida de acuerdo con el número de datos. Se escoge el rango de variación del orden de la matriz y se construye la matriz de observación de acuerdo con:

$$\phi_N = \begin{bmatrix} \varphi^T(1) \\ . \\ . \\ . \\ \varphi^T(N) \end{bmatrix}$$

Donde *u* y *y* corresponden a los datos de entrada y salida respectivamente.

- Obtención de la solución matricial.

  De acuerdo con la solución matricial para el algoritmo de mínimos cuadrados descrita por la ecuación:

$$\hat{\theta}_N = [\phi_N^T \phi_N]^{-1} \phi_N^T Y_N$$

  Donde Y_n representa el vector de datos del experimento, se obtiene el estimador de los parámetros del modelo $\hat{\theta}$.

- Cálculo de la función de transferencia del modelo.

  Se calcula la función de transferencia del modelo generado para poder evaluar posteriormente el comportamiento de la salida comparada con los datos de los experimentos.

- Validación del modelo.

Para la validación del modelo se toman la segunda mitad de los datos de entrada del experimento debidamente procesados y se aplican al modelo obtenido. Con los datos de salida de este modelo y los datos del experimento se hace una comparación para analizar su validez.

- Cálculo el tiempo de ejecución del algoritmo.

  Mediante la función de reloj del matlab se calcula el tiempo de ejecución del algoritmo como elemento de comparación para los criterios propuestos.

- Calculo del FPE (Final Prediction Error).

  Después de comparar los resultados, se calcula el FPE mediante la expresión:

$$FPE = \min_{d,\theta} \left(\frac{1+\frac{d}{N}}{1-\frac{d}{N}} * \frac{1}{N} * \sum_{t=1}^{N} \in^2 (t,\theta) \right)$$

- Validación con una entrada escalón.

  Posteriormente se valida la respuesta del sistema ante una entrada escalón unitario. Esta validación permite comparar que tan cerca está el modelo calculado de los datos del proceso.

En la figura 3.1 se esquematiza de manera breve el algoritmo de mínimos cuadrados propuesto.

Algoritmo de Mínimos Cuadrados

1. Medición de datos de entrada y datos para validación.
2. Selección y tratamiento de datos para construcción del modelo.
3. Construcción de la matriz de parámetros.

$$\phi_N = \begin{bmatrix} \varphi^T(1) \\ . \\ . \\ . \\ \varphi^T(N) \end{bmatrix}$$

4. Obtención de la solución matricial.

$$\hat{\alpha}_G = (X^t X)^{-1} X^t Y$$

5. Cálculo de la función de transferencia del modelo.
6. Validación del modelo.
7. Cálculo del tiempo de ejecución del algoritmo y del FPE

$$FPE = \min_{d,\theta} \left(\frac{1+\frac{d}{N}}{1-\frac{d}{N}} * \frac{1}{N} * \sum_{t=1}^{N} \in^2 (t,\theta) \right)$$

Fig. 3.1. Esquema del algoritmo de Mínimos Cuadrados.

El código en matlab del algoritmo de mínimos cuadrados para la obtención de un modelo paramétrico ARX se encuentra en el anexo A.

Los archivos para la ejecución del algoritmo son:

1. **MinCuadrado.m**. Este es el archivo de matlab que contiene la implementación del algoritmo de mínimos cuadrados. Es el archivo ejecutable.
2. **cstr_parameter.m.** Este es el archivo de matlab que contiene los parámetros del reactor.
3. **ReactorBenchmarkDataId**. Este es el archivo de simulink que se ejecuta con MinCuadrado.m y genera los datos para la construcción del modelo.
4. **valida1.** Este es el archivo de simulink que se ejecuta con MinCuadrado.m y genera los datos para la validación del modelo.

3.3 ALGORITMO DE MÍNIMOS CUADRADOS PONDERADOS

Al igual que en el algoritmo de mínimos cuadrados, para la implementación de este algoritmo se aplicó un método para la identificación de un sistema SISO (single input single output) con modelos paramétricos ARX. Se utilizó la herramienta de simulink para la generación de los datos de entrada al algoritmo y datos de validación. El algoritmo de mínimos cuadrados ponderados es el siguiente:

- Selección de datos para construcción del modelo.

Del total de datos de entrada se escoge la primera mitad de ellos para la construcción del modelo como los datos experimentales.

- Tratamiento de señal de los datos seleccionados.

  Mediante el uso de funciones de Matlab, se hace un tratamiento de los datos seleccionados, removiendo los niveles constantes de la entrada y la salida, haciendo el valor medio igual a cero.

- Construcción de la matriz de parámetros.

  Se determinan las dimensiones de la matriz de entrada y de la matriz de salida de acuerdo con el número de datos. Se escoge el rango de variación del orden de la matriz y se construye la matriz de observación de acuerdo con:

$$\phi_N = \begin{bmatrix} \varphi^T(1) \\ . \\ . \\ . \\ \varphi^T(N) \end{bmatrix}$$

Donde *u* y *y* corresponden a los datos de entrada y salida respectivamente.

- Determinación de la matriz de ponderación.

  De acuerdo con lo descrito en el capítulo III, se calcula la matriz de ponderación de acuerdo con:

$$\psi^{-1} = \begin{pmatrix} 1 & -\rho & 0 & ... & 0 & 0 \\ -\rho & 1+\rho^2 & -\rho & ... & 0 & 0 \\ 0 & -\rho & 1+\rho^2 & ... & 0 & 0 \\ . & . & . & . & . & . \\ 0 & 0 & 0 & . & 1+\rho^2 & -\rho \\ 0 & 0 & 0 & . & -\rho & 1 \end{pmatrix}$$

- Obtención de la solución matricial.

  De acuerdo con la solución matricial para el algoritmo de mínimos cuadrados ponderados descrita por la ecuación:

$$\hat{\alpha}_G = (X^t \psi^{-1} X)^{-1} X^t \psi^{-1} Y$$

  Donde X corresponde a la matriz de observación, ψ^{-1} es la matriz de ponderación, Y es el vector de datos del experimento y $\hat{\alpha}_G$ corresponde a los parámetros estimados.

- Cálculo de la función de transferencia del modelo.

  Se calcula la función de transferencia del modelo generado para poder evaluar posteriormente el comportamiento de la salida comparada con los datos de los experimentos.

- Validación del modelo.

  Para la validación del modelo se toman la segunda mitad de los datos de entrada del experimento debidamente procesados y se aplican al modelo obtenido. Con los datos de salida de este modelo y

los datos del experimento se hace una comparación gráfica para analizar su validez.

- Cálculo el tiempo de ejecución del algoritmo.

  Mediante la función de reloj del matlab se calcula el tiempo de ejecución del algoritmo como elemento de comparación para los criterios propuestos.

- Cálculo el FPE (Final Prediction Error).

  Después de comparar los resultados, se calcula el FPE mediante la expresión:

$$FPE = \min_{d,\theta} \left(\frac{1+\frac{d}{N}}{1-\frac{d}{N}} * \frac{1}{N} * \sum_{t=1}^{N} \in^2 (t,\theta) \right)$$

- Validación con una entrada escalón.

  Posteriormente se valida la respuesta del sistema ante una entrada escalón unitario. Esta validación permite comparar que tan cerca está el modelo de predicción de la salida del reactor.

En la figura 3.2 se esquematiza de manera breve el algoritmo de mínimos cuadrados ponderados propuesto.

El código en matlab del algoritmo de mínimos cuadrados ponderados para la obtención de un modelo paramétrico ARX se encuentra en el anexo B.

Algoritmo de Mínimos Cuadrados Ponderados

1. Medición de datos de entrada y datos para validación.
2. Selección y tratamiento de señales de los datos para construcción del modelo.
3. Construcción de la matriz de parámetros.

$$\phi_N = \begin{bmatrix} \varphi^T(1) \\ . \\ . \\ . \\ \varphi^T(N) \end{bmatrix}$$

4. Determinación de la matriz de ponderación.

$$\psi^{-1} = \begin{pmatrix} 1 & -\rho & 0 & ... & 0 & 0 \\ -\rho & 1+\rho^2 & -\rho & ... & 0 & 0 \\ 0 & -\rho & 1+\rho^2 & ... & 0 & 0 \\ . & . & . & . & . & . \\ 0 & 0 & 0 & . & 1+\rho^2 & -\rho \\ 0 & 0 & 0 & . & -\rho & 1 \end{pmatrix}$$

5. Obtención de la solución matricial.

$$\hat{\alpha}_G = (X^t \psi^{-1} X)^{-1} X^t \psi^{-1} Y$$

6. Cálculo de la función de transferencia del modelo.
7. Validación del modelo.
8. Cálculo del tiempo de ejecución del algoritmo y del FPE

$$FPE = \min_{d,\theta} \left(\frac{1+\frac{d}{N}}{1-\frac{d}{N}} * \frac{1}{N} * \sum_{t=1}^{N} \in^2 (t,\theta) \right)$$

Fig. 3.2. Esquema del algoritmo de mínimos cuadrados ponderados

Los archivos para la ejecución del algoritmo de mínimos cuadrados ponderados son:

1. **MinCuadPond.m**. Este es el archivo de matlab que contiene la implementación del algoritmo de mínimos cuadrados ponderados. Es el archivo ejecutable.
2. **cstr_parameter.m.** Este es el archivo de matlab que contiene los parámetros del reactor.
3. **ReactorBenchmarkDataId**. Este es el archivo de simulink que se ejecuta con MinCuadrado.m y genera los datos para la construcción del modelo.
4. **valida1.** Este es el archivo de simulink que se ejecuta con MinCuadrado.m y genera los datos para la validación del modelo.

En este capitulo se presentaron los conceptos y funciones del matlab como herramienta computacional utilizada y se describieron los algoritmos de mínimos cuadrados y mínimos cuadrados implementados.

CAPITULO IV.- APLICACIÓN DE LAS TECNICAS IMPLEMENTADAS

En la primera sección de este capítulo se hace una descripción detallada del modelo utilizado y su implementación en simulink. Posteriormente se explica el diseño de las pruebas para los datos de entrada y finalmente se hace la aplicación de los algoritmos al modelo descrito para diferentes tiempos de muestreo. Como un caso especial se aplican los algoritmos para una señal de entrada con perturbaciones a los modelos con menor FPE y se hace una validación de un modelo de referencia con un impulso como una señal de entrada.

4.1 PROCESO BENCHMARK UTILIZADO

Para la aplicación de los algoritmos con las técnicas de mínimos cuadrados y mínimos cuadrados ponderados se escogió el modelo de un proceso de un reactor químico tipo tanque agitado continuamente (CSTR) con sus respectivas variables de entradas y salidas, constantes propias del sistemas y algunas consideraciones para la prueba en puntos de equilibrio de CSTR, presentado por [38].

El reactor CSTR consiste en un tanque al que continuamente fluye el alimento y descarga productos a flujos volumétricos tales que el volumen de reacción

permanece constante. Esta provisto de facilidades para transferencia de calor y para agitación. En condiciones ideales la mezcla es homogénea.

El modelo presentado está dado en variables adimensionales, tal como expresan los autores en sus respectivos artículos, de tal forma que los resultados puedan ser fácilmente escalados a cualquier reactor de este tipo. En la bibliografía se puede observar que esta es una manera muy común para reportar los estudios investigativos sobre reactores químicos [55], [56].

El modelo matemático, en variables adimensionales, está dado por:

$$\frac{dx_1}{d\tau} = -Dax_1 \exp\left[\frac{x_2}{1+x_2/\lambda}\right] + q(x_{1f} - x_1) \quad (4.1)$$

$$\frac{dx_2}{d\tau} = \beta Dax_1 \exp\left[\frac{x_2}{1+x_2/\lambda}\right] - (q+\delta)x_2 + \delta x_3 + x_{2f} \quad (4.2)$$

$$\frac{dx_3}{d\tau} = \frac{q_c(x_{3f} - x_3)}{\delta_1} + \frac{\delta(x_2 - x_3)}{\delta_1\delta_2} \quad (4.3)$$

Donde:

x_1, x_{1f}: composición del reagente de entrada del reactor y composición del reagente

x_2, x_{2f}: temperatura de entrada del reactor y temperatura del reactor

x_3, x_{3f} : temperatura de entrada de la chaqueta y temperatura de la chaqueta

q, q_c : flujos de entrada del reactor y de la chaqueta

Da : número de Damköler

λ : energía de activación

β : calor de reacción

δ : coeficiente de transferencia de calor

δ_1 : relación de volumen

δ_2 : densidad de la relación de capacidad de calor

τ : tiempo

La ecuación 4.1 describe el balance de masas del componente del reactivo dentro del reactor, la ecuación 4.2 corresponde al balance de energía dentro del reactor y la ecuación 4.3 al balance de energía dentro de la chaqueta de enfriamiento. El CSTR descrito aquí se ilustra en la figura 4.1.

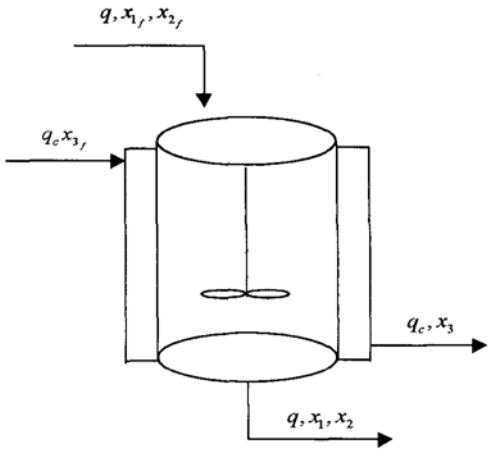

Fig. 4.1 Reactor tipo tanque agitado (CSTR)

Los parámetros de este sistema están resumidos en la tabla 4.1.

Tabla 4.1. Parámetros del CSTR usado

x_{1f}	x_{2f}	x_{3f}	β	Da	δ	δ_1	δ_2	γ	q_c
1.0	0.0	-1.0	8.0	0.072	0.3	0.1	0.5	20.0	1.65102

Para los parámetros dados en la tabla 4.1 y una entrada igual a 1.0, los puntos múltiples de equilibrios se muestran en la tabla 4.2.

Tabla 4.2. Puntos de equilibrio del CSTR

	Estable (1)	Inestable (2)	Estable (3)
x_1^E	0.8893	0.5528	0.1890
x_2^E	0.5193	2.7517	5.1373
x_3^E	-0.5950	0.0000	0.6359

El superíndice E *E*, se refiere a equilibrio; por ejemplo, para el caso estable (1), el valor de la salida de la temperatura de la chaqueta representada por x_3^E, significa que la temperatura de la chaqueta en equilibrio estable se alcanza para un valor de -0.5950.

Ya se explicó que la estabilidad se alcanza cuando en el sistema desaparecen todos los estados transitorios y la señal de salida alcanza una condición en estado estable.

4.2 GENERACIÓN DE LOS DATOS EXPERIMENTALES

Para la generación de los datos experimentales con los que se construyó el modelo, se implementó el modelo del reactor descrito en el inciso anterior. Para la implementación se utilizó el toolbox de simulink de matlab. Simulink es un software comúnmente utilizado en área de control para el modelamiento, simulación y análisis de sistemas dinámicos. El modelo y sus componentes se organizan a través de bloques que contienen la información de la dinámica del proceso. La implementación de este modelo con el uso del simulink se ilustra en la figura 4.2.

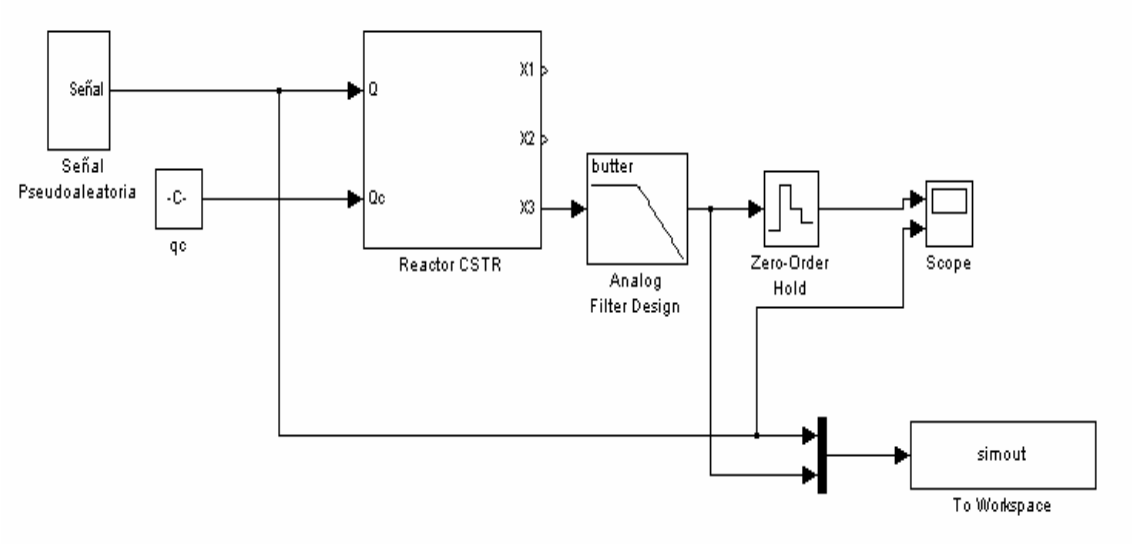

Fig. 4.2. Esquema del modelo del CSTR

Esta figura tiene un diagrama de bloques interconectados entre si que describen al modelo a simular. El bloque principal es el reactor CSTR, el cual tiene dos entradas; una entrada corresponde a la señal pseudoaleatoria

seleccionada, la cual simula el flujo de entrada al reactor (Q) y una segunda entrada que es el flujo de entrada a la chaqueta (Q_c).

En la figura 4.2 del bloque del reactor sale la señal de la temperatura de la chaqueta (x_3); esta señal se conecta a un filtro análogo y un retenedor de orden cero para hacer el tratamiento de la señal. Posteriormente se almacenan los datos de entrada al reactor y de salida en el bloque simout, para que puedan ser utilizados en el algoritmo implementado. El esquema de la señal de prueba se ilustra en la figura 4.3.

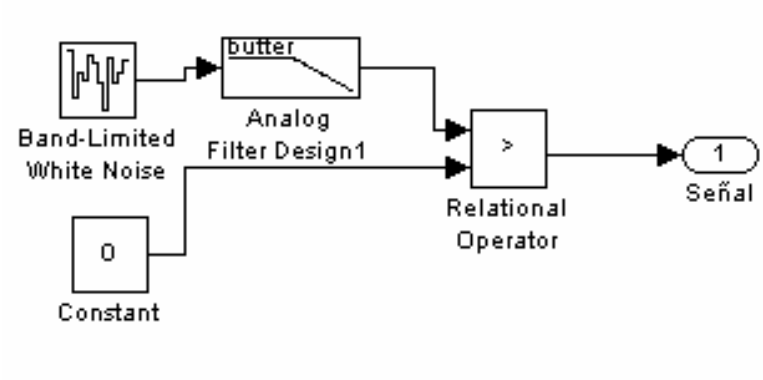

Fig. 4.3. Diagrama de bloques de la señal de prueba

En la figura 4.3 un operador relacional recibe como entradas la señal de ruido blanco con ancho de banda limitada, la cual es previamente filtrada y un bloque con valor constante e igual a cero. El operador relacional convierte las entradas a valores que oscilan entre 0 y 1. Estos valores son los que ingresan al subsistema del CSTR.

El reactor CSTR de la figura 4.2 es un subsistema. Este subsistema se representa en la figura 4.4.

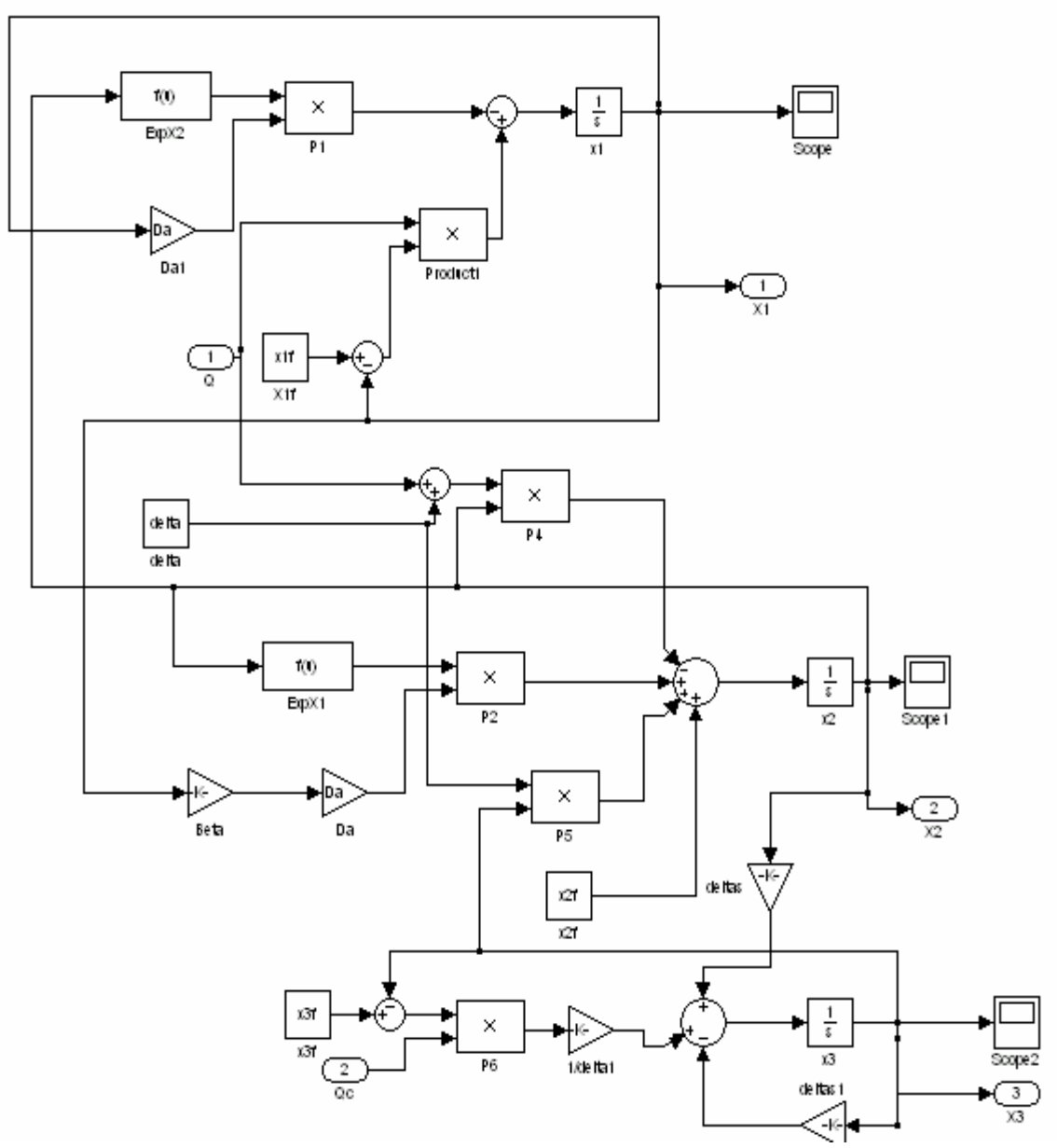

Fig. 4.4. Subsistema del CSTR

En la figura 4.4 se describe el diagrama de bloques de simulink, en el cual se relacionan el modelo del reactor seleccionado, el cual está descrito por las ecuaciones 4.1 a 4.3. Los bloques identificados con scope hacen las veces de un osciloscopio que permite apreciar gráficamente las señales.

4.3 DISEÑO DE LAS SEÑALES DE PRUEBA

De acuerdo con [18], para la identificación de sistemas, hay tres factores básicos que gobiernan la selección de señales de entrada en un experimento:

1. Las propiedades asintóticas del estimado dependen solamente del espectro de entrada – no de la forma de la entrada.
2. La entrada debe tener amplitud limitada.
3. Las entradas periódicas pueden tener ciertas ventajas.

Basado en estos factores, para el caso de estudio se desarrollaron simulaciones del modelo del reactor para verificar las condiciones de estabilidad propuestas por [30] y observar su comportamiento ante diferentes entradas.

En la figura 4.5 se observa el resultado de la simulación de la temperatura de la chaqueta ante una entrada igual a cero y valores iniciales de parámetros (x_1,

x_2, x_3) igual a cero. Como era de esperarse la salida tiende al valor de -1 (tabla 4.1).

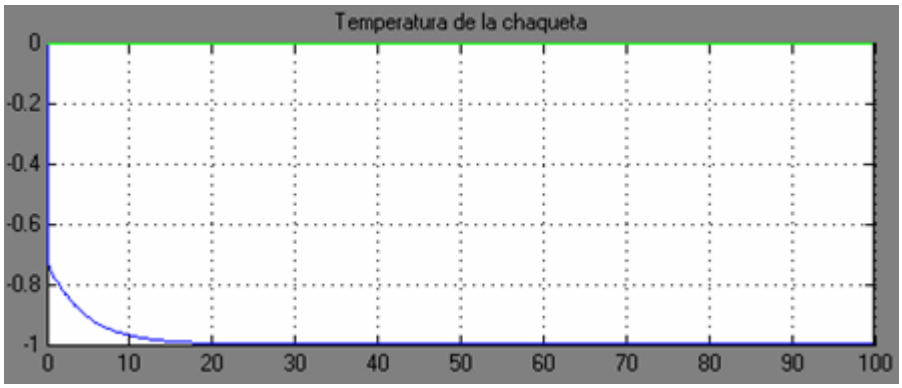

Fig. 4.5. Temperatura de la chaqueta ante entrada cero.

En la figura 4.6 se puede ver como la temperatura de la chaqueta tiende a un valor de -0.5950 (tabla 4.2) ante una entrada escalón unitario (q=1) y valores iniciales de parámetros (x_1, x_2, x_3) igual a cero.

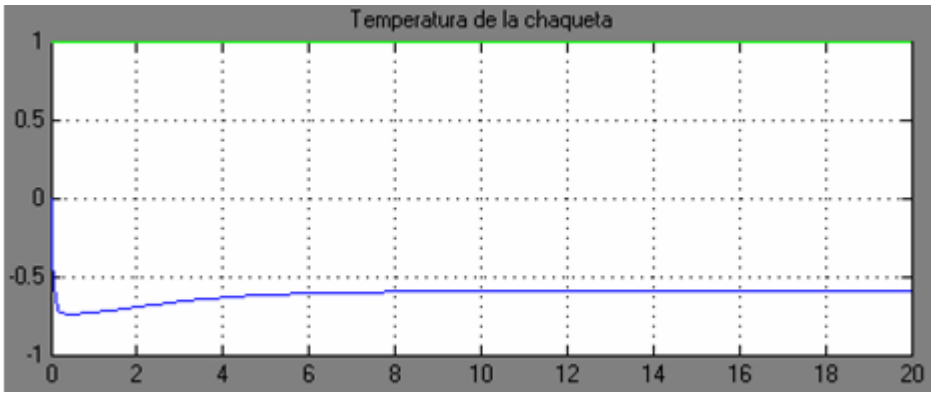

Fig. 4.6. Temperatura de la chaqueta ante entrada unitaria.

Como era de esperarse ante una entrada escalón unitario (q=1) para punto1 de equilibrio estable (x_1=0.8933, x_2= 0.5193, x_3=-0.5950), la respuesta se

mantiene en -0.595 para la temperatura de la chaqueta tal como se ilustra en la figura 4.7.

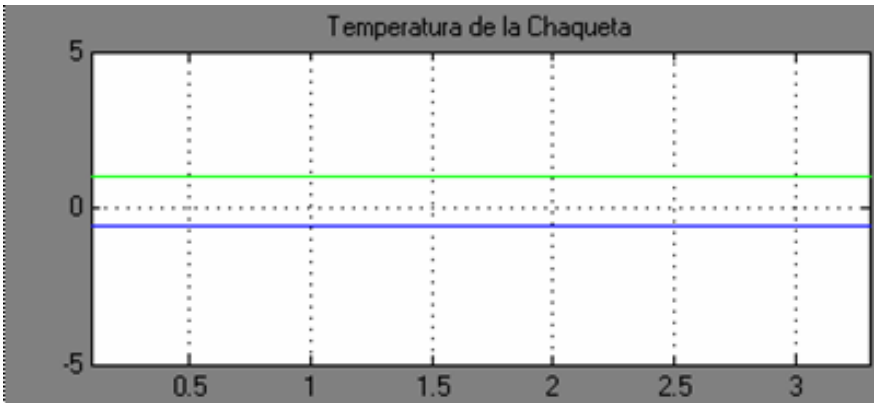

Fig. 4.7. Temperatura de la chaqueta ante entrada unitaria y punto de equilibrio estable.

Estos resultados de la simulación permiten verificar la correspondencia del modelo con los parámetros sugeridos por los autores [30].

Con base en el comportamiento de la salida ante diferentes amplitudes de la señal de entrada y a partir de las pruebas se escogió una señal pseudoaleatoria. Esta señal tiene las características de tener un espectro en frecuencia apropiado y mantener una amplitud constante, factores claves para la selección de la señal de entrada [18]. El bloque de la señal pseudoaleatoria descrito en la figura 4.3 está formado por una señal de ruido que proporciona valores aleatorios, un operador relacional con el cual se compara y entrega solo dos valores a la salida (0 y 0.5), como se muestra en la figura 4.8.

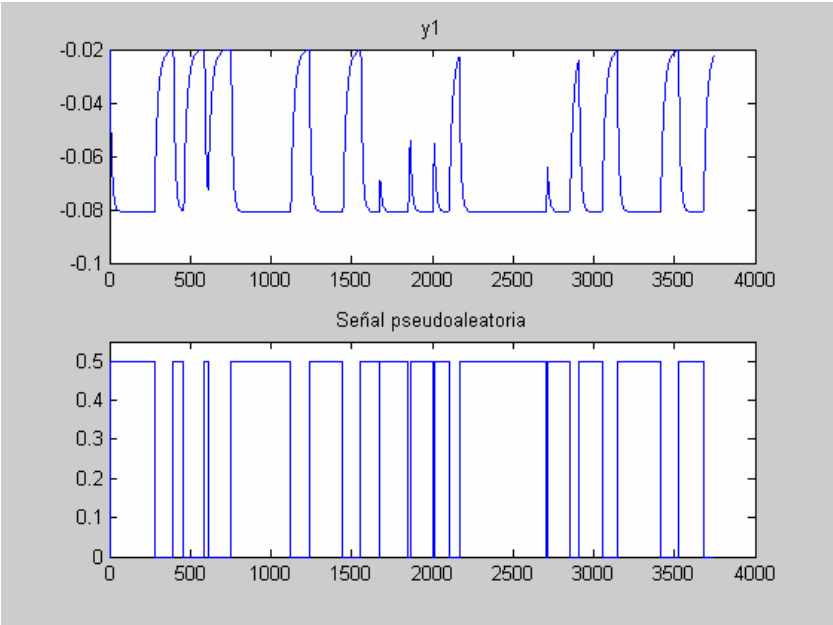

Fig. 4.8. Señal pseudoaleatoria y temperatura de la chaqueta del reactor

Como resultado de la experimentación con esta señal pseudoaleatoria aplicada al modelo verificado del reactor, se observó que la salida del sistema se estabilizaba, en el peor de los casos, alrededor de los 150 segundos por lo que se tomó una ventana de observación de datos para la construcción del modelo de 300 segundos. La razón de duplicar el tiempo de estabilización de la salida del modelo es por la necesidad de tomar la mitad de los datos para la construcción del modelo de predicción y la otra mitad para hacer la validación.

Los datos para la experimentación fueron procesados adecuadamente como entradas a los algoritmos. Es importante señalar que la simulación se hizo para el primero de los puntos de equilibrio estable descrito en la tabla 4.2.

4.4 APLICACIÓN DE LOS ALGORITMOS

Para el análisis comparativo de los algoritmos, se tomaron como referencia los tiempos de ejecución de cada algoritmo y el valor del FPE. Los algoritmos generaron modelos que van desde el orden dos (2) hasta el orden diez (10) con un periodo de muestreo variable entre 0.08 y 1.0 segundos.

4.4.1 Resultados para el algoritmo de mínimos cuadrados. Como se explicó anteriormente, se escogieron los algoritmos relacionados con el modelo ARX. Además se escogió un rango de análisis de trescientos segundos (300s) para diferentes tiempos de muestreo. Este rango es suficientemente amplio para que el modelo logre su estabilización ante las entradas pseudoaleatorias, de acuerdo con los resultados previos de simulación del modelo del CSTR. Además, el tiempo de muestreo es el que determina la cantidad de datos tomados para la obtención del modelo, que es una variable de referencia en la comparación de los algoritmos.

4.4.1.1 Caso 1. Este caso se analiza para un tiempo de muestreo de 0.08 segundos. Durante un rango de observación de 300 segundos, se generaron 3750 datos (que equivale a dividir 300 segundos entre 0.08 segundos), de los cuales se tomó la primera mitad de ellos (1875 datos) para la construcción del modelo y la segunda mitad para la validación. Los resultados se muestran en la tabla 4.3.

Tabla 4.3. Modelos ARX con Ts=0.08

Numero De datos para construir el modelo	Orden del sistema	MINIMOS CUADRADOS	
		Tiempo ejecución (segundos)	FPE
1875	2	0,7190	0,0126750
	3	1,5630	0,0166670
	4	2,6100	0,0194490
	5	3,7040	0,0213080
	6	4,6720	0,0224870
	7	5,5790	0,0023180
	8	6,4850	0,0023540
	9	7,3600	0,0236890
	10	8,2500	0,0237230

Del resultado de la simulación para una muestra de 1875 datos (tabla 4.3) se pueden extraer las siguientes observaciones:

1. El modelo de orden 2 es el que presenta el menor tiempo de ejecución del algoritmo con 0,7190 segundos.
2. En la medida en que el orden del sistema aumenta, no representa una mejora del FPE.
3. Como es de esperarse, a medida aumenta el orden del sistema aumenta el tiempo de ejecución del algoritmo.

De acuerdo a estas observaciones se nota una tendencia a que para valores grandes del orden del modelo del sistema, el tiempo de ejecución del algoritmo también aumenta.

En la figura 4.9 se puede observar la comparación de las gráficas de temperatura de la chaqueta (x_3); lo que se identifica como salida real en la gráfica, corresponde a la salida del modelo no lineal descrito por la simulación de las ecuaciones 4.1 a 4.3. La salida del modelo (verde) que aparece en la gráfica corresponde a la salida del modelo de predicción generado por el algoritmo.

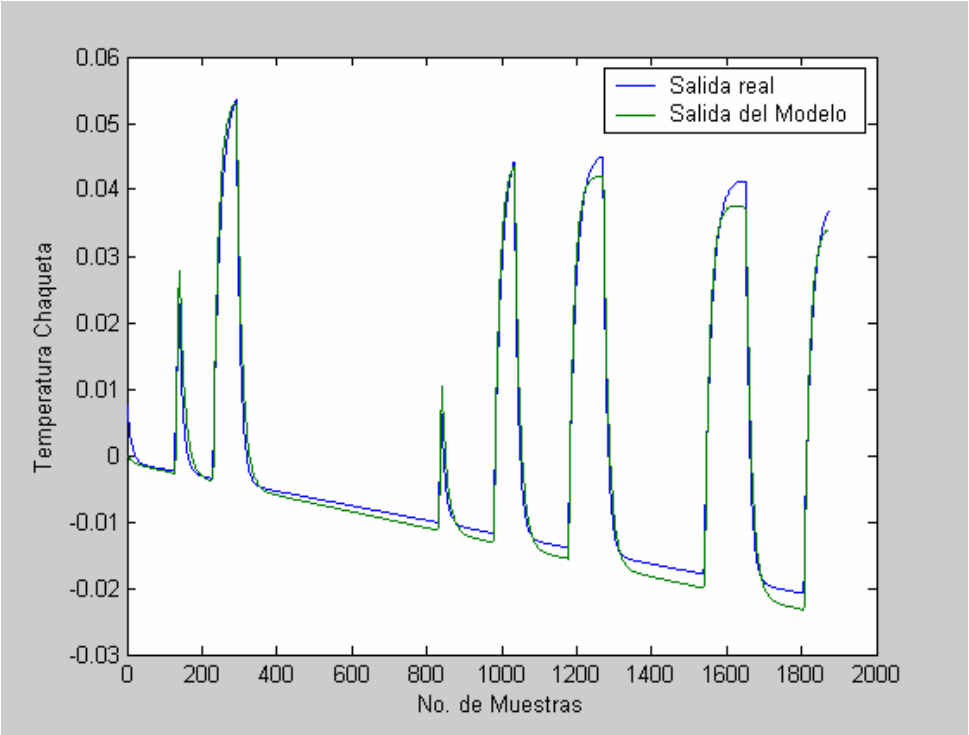

Fig. 4.9 Temperatura de la chaqueta del reactor

De esta gráfica se pueden extraer las siguientes observaciones:

1. Para las primeras 400 muestras hay una alta correlación entre las dos señales de salida. Esta correlación se refiere a que tan aproximadas son las dos señales bajo comparación.

2. Para un número de muestras superiores a 400 se presentan variaciones entre las señales de salida, principalmente en los puntos donde la pendiente es pequeña.

En la figura 4.10 se ilustra la validación del modelo generado por el algoritmo, ante una entrada escalón unitario al sistema. Lo que se busca con una entrada escalón unitario es poder hacer un análisis cualitativo de la respuesta, porque al aplicar una entrada de este tipo se puede hacer una comparación de dos señales de salida (real y simulada por ejemplo) evaluando variables como el tiempo de establecimiento y la forma de la señal.

El tiempo de establecimiento es el tiempo que se demora la señal para alcanzar su estado de equilibrio ante una entrada escalón. Con esta comparación se puede emitir un juicio acerca de que tanto el modelo construido tiene la capacidad de comportarse lo más cercanamente posible a la planta o al proceso original.

En la figura 4.10 la curva de color amarillo representa la temperatura de la chaqueta a partir de los datos obtenidos por la simulación de las ecuaciones 4.1 a 4.3 (Temperatura real) y las curvas de color diferente representan la temperatura de la chaqueta a partir de los modelos de predicción para diferente orden generados por el algoritmo (Temperatura del modelo). Esta referencia de colores se mantiene a lo largo de esta sección en los casos de análisis.

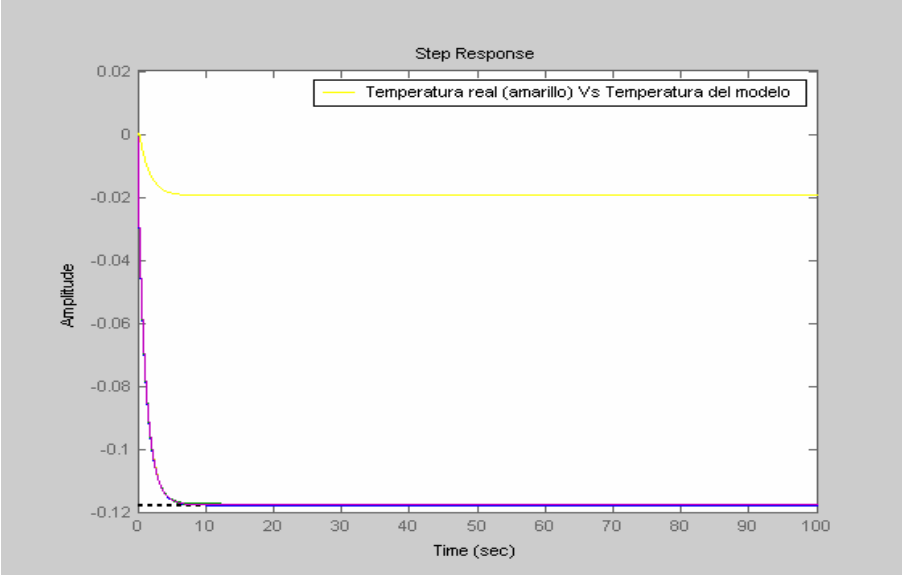

Fig. 4.10. Temperatura de la chaqueta para diferente orden del modelo

De esta gráfica se pueden extraer las siguientes observaciones desde el punto de vista cualitativo:

1. Hay una diferencia entre las amplitudes de las señales del modelo generado por el algoritmo y la señal de salida del reactor; este es un problema que puede ajustarse con la ganancia del modelo.
2. El comportamiento de las señales comparadas es similar dado que parten del mismo punto (cero) y con pendiente negativa se estabilizan alrededor de un punto. Estos puntos son -0.02 para la salida del reactor y de -0.12 para la salida del modelo.

4.4.1.2 Caso 2. Este caso se analiza para un tiempo de muestreo de 0.1 segundos. Durante un rango de observación de 300 segundos, se generaron

3000 datos, de los cuales se tomó la primera mitad de ellos (1500) para la construcción del modelo y la segunda mitad para la validación. Los resultados se muestran en la tabla 4.4.

Del resultado de la simulación para una muestra de 1500 datos (tabla 4.4) se pueden extraer las siguientes observaciones:

Tabla 4.4. Modelos ARX con Ts=0.1

Numero De datos para construir el modelo	Orden del sistema	MINIMOS CUADRADOS	
		Tiempo ejecución (segundos)	FPE
1500	2	0,5940	0,0205860
	3	1,2970	0,0266480
	4	1,9690	0,0307410
	5	2,8290	0,0333970
	6	3,5320	0,0350330
	7	4,3130	0,0035970
	8	5,1410	0,0364460
	9	5,9380	0,0366420
	10	6,7500	0,0366990

1. El modelo de orden 2 es el que presenta el menor tiempo de ejecución del algoritmo con 0,5940 segundos.
2. El menor FPE corresponde al modelo de orden 7.

En la figura 4.11 se puede observar la relación que hay entre la temperatura de la chaqueta a partir de los datos de salida del reactor (salida real) y la

temperatura del proceso a partir del modelo generado por el algoritmo (salida del modelo).

De esta gráfica se pueden extraer las siguientes observaciones:

1. Para las primeras 500 muestras hay una alta correlación entre las dos señales de salida. Esta correlación se refiere a que tan aproximadas son las dos señales bajo comparación.
2. Para un número de muestras superiores a 500, se presentan variaciones entre las señales de salida, principalmente en los puntos donde la pendiente es pequeña o la respuesta tiende a un valor constante (-0.01).

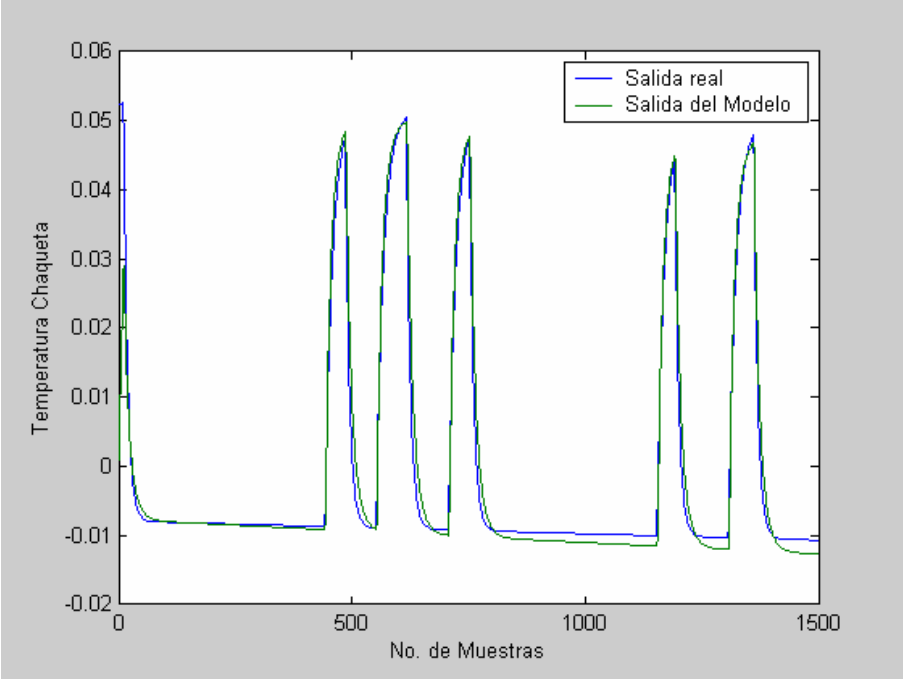

Fig. 4.11. Temperatura de la chaqueta del reactor (azul) y la temperatura del modelo (verde).

En la figura 4.12 se ilustra la validación del modelo generado por el algoritmo, ante una entrada escalón unitario al sistema.

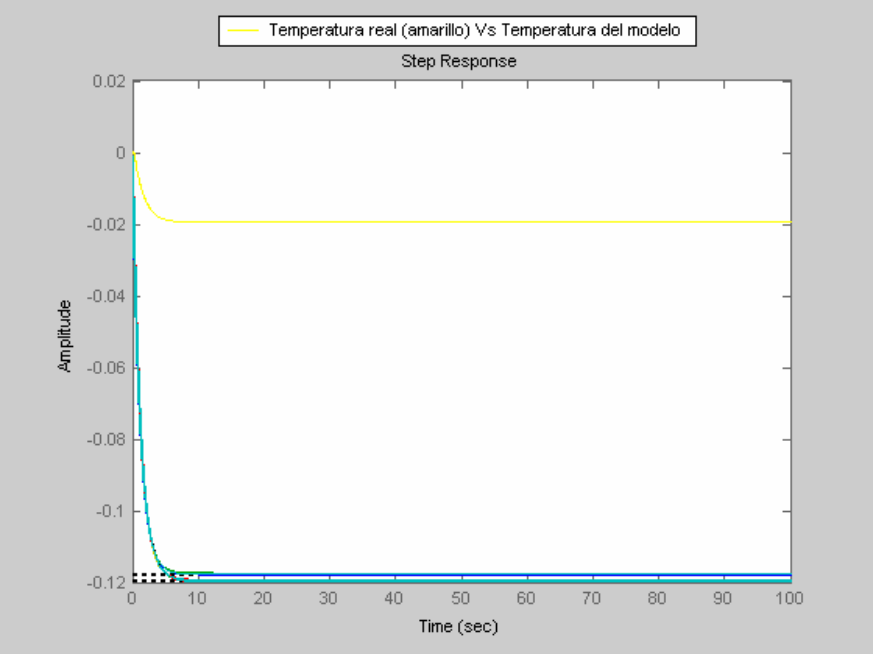

Fig. 4.12. Temperatura de la chaqueta para diferente orden del modelo.

De la figura 4.12 se pueden extraer las siguientes observaciones:

1. Hay una diferencia entre las amplitudes de las señales del modelo generado por el algoritmo y la señal de salida del reactor; este es un problema que puede ajustarse con la ganancia del modelo.
2. El comportamiento de las señales comparadas es similar dado que parten del mismo punto (cero) y con pendiente negativa se estabilizan alrededor de un punto a los 10 segundos aproximadamente.

4.4.1.3 Caso 3. Este caso se analiza para un tiempo de muestreo de 0.2 segundos. Durante un rango de observación de 300 segundos, se generaron 1500 datos, de los cuales se tomó la primera mitad de ellos (750) para la construcción del modelo y la segunda mitad para la validación. Los resultados se muestran en la tabla 4.5.

Tabla 4.5. Modelos ARX con Ts=0.2 s

Numero De datos para construir el modelo	Orden del sistema	MINIMOS CUADRADOS	
		Tiempo ejecución (segundos)	FPE
750	2	0,6410	0,0443570
	3	1,4530	0,0509820
	4	2,2810	0,0535320
	5	3,1720	0,0542130
	6	4,0630	0,0542830
	7	5,0160	0,0544860
	8	5,9530	0,0055150
	9	6,9850	0,0563520
	10	8,0310	0,0580940

Del resultado de la simulación para una muestra de 750 datos (tabla 4.5) se pueden extraer las siguientes observaciones:

1. El modelo de orden 2 es el que presenta el menor tiempo de ejecución del algoritmo con 0,6410 segundos.
2. El menor FPE corresponde al modelo de orden 8.

En la figura 4.13 se puede observar la relación que hay entre la temperatura de la chaqueta a partir de los datos de salida del reactor (salida real) y la temperatura del proceso a partir del modelo generado por el algoritmo (salida del modelo).

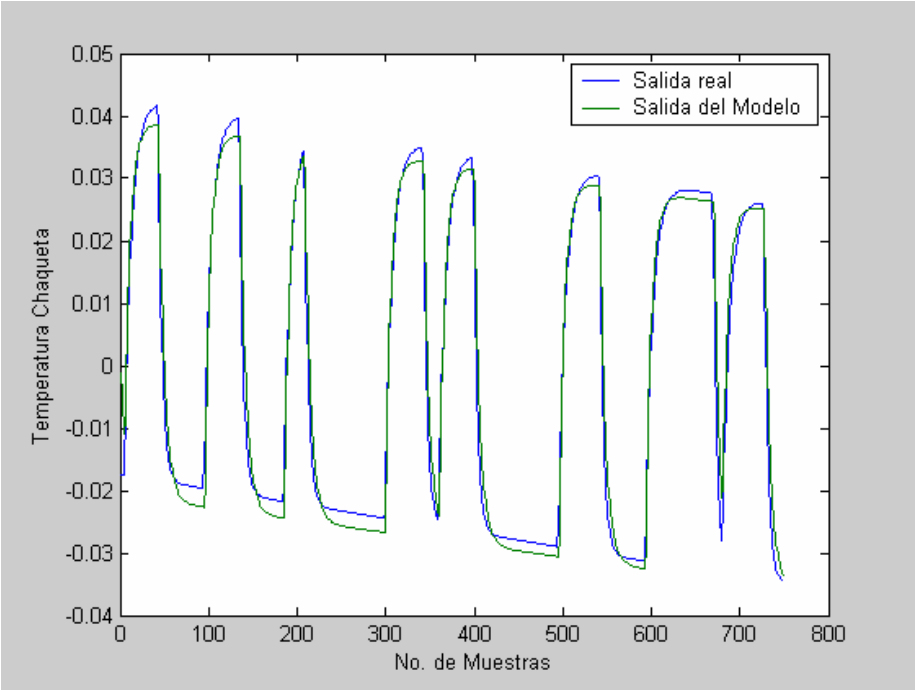

Fig. 4.13. Temperatura de la chaqueta del reactor (azul) y la temperatura del modelo (verde).

De la figura 4.13 se pueden extraer las siguientes observaciones:

1. Para las primeras 500 muestras hay una baja correlación entre las dos señales de salida, especialmente en aquellos puntos donde el valor de la

señal tiende a ser constante. Esta correlación se refiere a que tan aproximadas son las dos señales bajo comparación.

2. Para un número de muestras superiores a 500, la correlación mejora principalmente alrededor de las 600 muestras.

En la figura 4.14 se ilustra la validación del modelo generado por el algoritmo, ante una entrada escalón unitario al sistema.

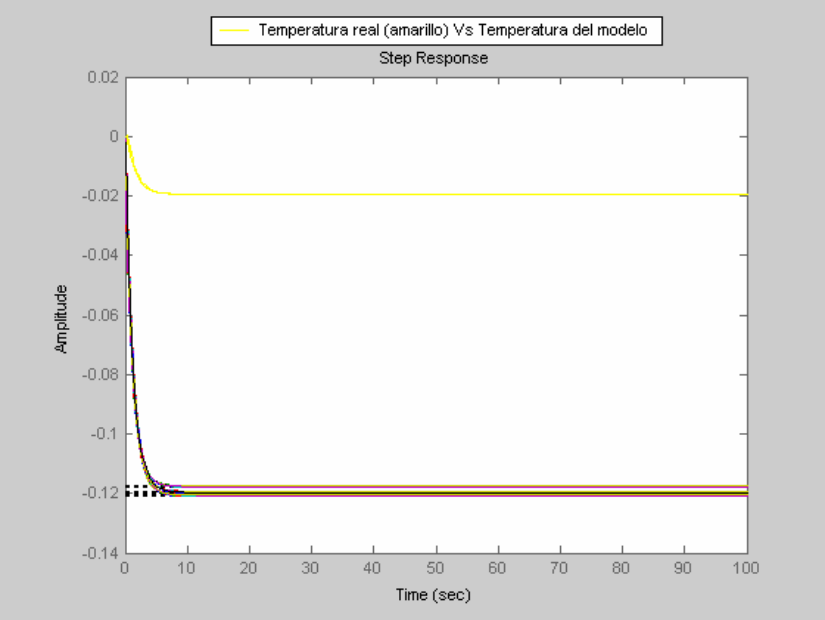

Fig. 4.14. Temperatura de la chaqueta para diferente orden del modelo.

De la figura 4.14 se pueden extraer las siguientes observaciones:

1. Hay una diferencia entre las amplitudes de las señales del modelo generado por el algoritmo y la señal de salida del reactor; este es un problema que puede ajustarse con la ganancia del modelo.

2. El comportamiento de las señales comparadas es similar dado que parten del mismo punto (cero) y con pendiente negativa se estabilizan alrededor de un punto a los 10 segundos aproximadamente.

4.4.1.4 Caso 4. Este caso se analiza para un tiempo de muestreo de 0.4 segundos. Durante un rango de observación de 300 segundos, se generaron 750 datos, de los cuales se tomó la primera mitad de ellos (375) para la construcción del modelo y la segunda mitad para la validación. Los resultados se muestran en la tabla 4.6.

Tabla 4.6. Modelos ARX con Ts=0.4 s

Numero De datos para construir el modelo	Orden del sistema	MINIMOS CUADRADOS	
		Tiempo ejecución (segundos)	FPE
375	2	0,4690	0,0867650
	3	1,8590	0,0088030
	4	3,3590	0,0880050
	5	4,8280	0,0900210
	6	6,3900	0,0941520
	7	7,9840	0,1000700
	8	9,5620	0,1072900
	9	11,2810	0,1154900
	10	12,9690	0,1244400

Del resultado de la simulación para una muestra de 375 datos (tabla 4.6) se pueden extraer las siguientes observaciones:

1. El modelo de orden 2 es el que presenta el menor tiempo de ejecución del algoritmo con 0,4690 segundos.
2. El menor FPE corresponde al modelo de orden 3.

En la figura 4.15 se puede observar la relación que hay entre la temperatura de la chaqueta a partir de los datos de salida del reactor (salida real) y la temperatura del proceso a partir del modelo generado por el algoritmo (salida del modelo).

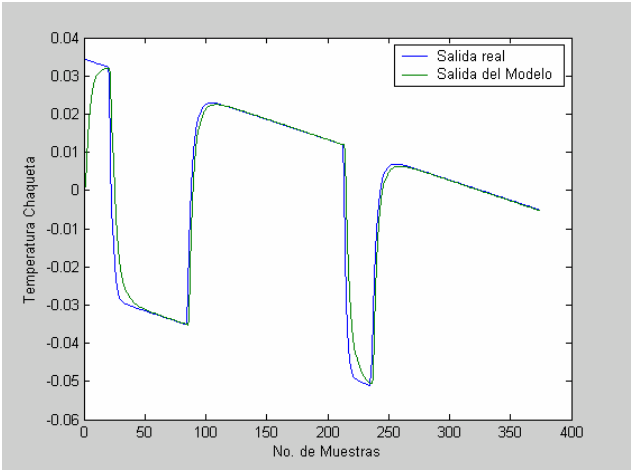

Fig. 4.15. Temperatura de la chaqueta del reactor (azul) y la temperatura del modelo (verde).

De la figura 4.15 se pueden extraer las siguientes observaciones:

1. Para las muestras tomadas, hay una alta correlación entre las dos señales. Esta correlación se refiere a que tan aproximadas son las dos señales bajo comparación.

2. A diferencia de los anteriores experimentos, las señales presentan menos correlación en los puntos de mayor pendiente.

En la figura 4.16 se ilustra la validación del modelo generado por el algoritmo, ante una entrada escalón unitario al sistema.

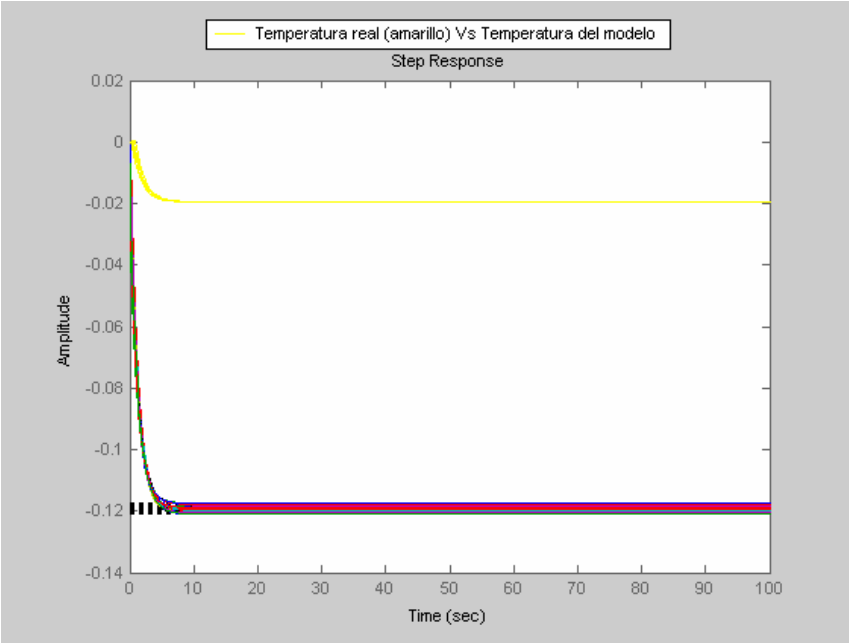

Fig. 4.16. Temperatura de la chaqueta para diferente orden del modelo.

De la figura 4.16 se pueden extraer las siguientes observaciones:

1. Hay una diferencia entre las amplitudes de las señales del modelo generado por el algoritmo y la señal de salida del reactor; este es un problema que puede ajustarse con la ganancia del modelo.
2. El comportamiento de las señales comparadas es similar dado que parten del mismo punto (cero) y con pendiente negativa se estabilizan alrededor de un punto a los 7 segundos aproximadamente.

4.4.1.5 Caso 5. Este caso se analiza para un tiempo de muestreo de 0.8 segundos. Durante un rango de observación de 300 segundos, se generaron 375 datos, de los cuales se tomó la primera mitad de ellos (188) para la construcción del modelo y la segunda mitad para la validación. Los resultados se muestran en la tabla 4.7.

Tabla 4.7. Modelos ARX con Ts=0.8 s

Numero De datos para construir el modelo	Orden del sistema	MINIMOS CUADRADOS	
		Tiempo ejecución (segundos)	FPE
188	2	0,4840	0,1539100
	3	4,7030	0,1628600
	4	8,7340	0,1851400
	5	12,7500	0,2138900
	6	16,8280	0,2461400
	7	20,9840	0,2794400
	8	25,1250	0,3155400
	9	29,4060	0,3533800
	10	33,7970	0,3930100

Del resultado de la simulación para una muestra de 188 datos (tabla 4.7)

1. El modelo de orden 2 es el que presenta el menor tiempo de ejecución del algoritmo con 0,4840 segundos.
2. El menor FPE corresponde al modelo de orden 2.

De acuerdo a estas observaciones se nota una tendencia a que para valores grandes del orden del modelo del sistema, el tiempo de ejecución del algoritmo también aumenta.

En la figura 4.17 se puede observar la relación que hay entre la temperatura de la chaqueta a partir de los datos de salida del reactor (salida real) y la temperatura del proceso a partir del modelo generado por el algoritmo (salida del modelo).

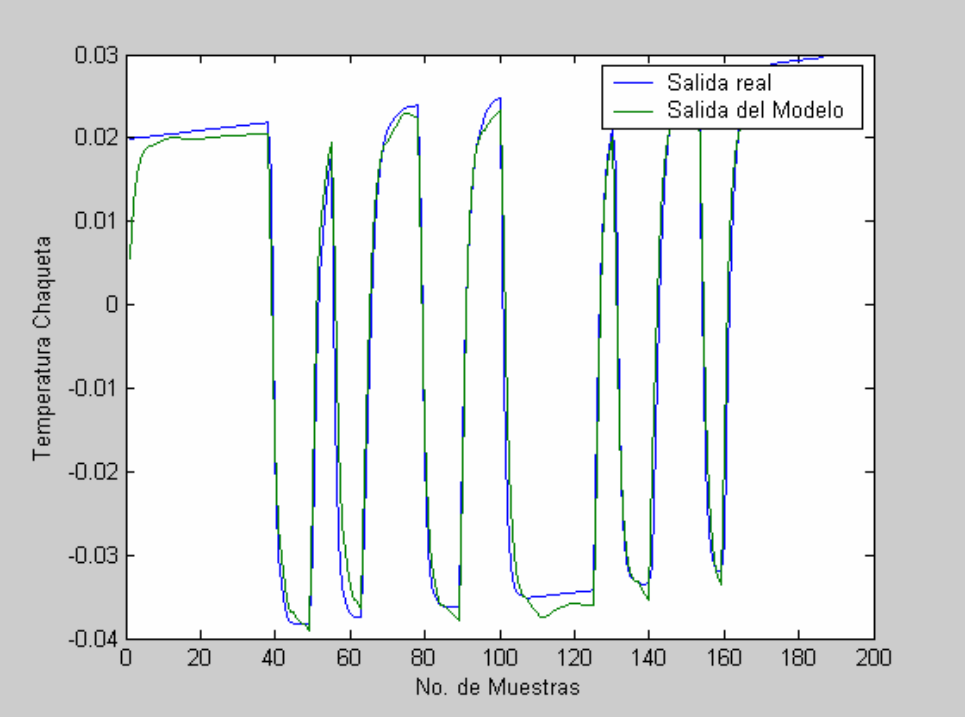

Fig. 4.17. Temperatura de la chaqueta del reactor (azul) y la temperatura del modelo (verde).

De la figura 4.17 se pueden extraer las siguientes observaciones:

1. Para las primeras 100 muestras hay una relativa alta correlación entre las dos señales de salida, especialmente en aquellos puntos donde el valor de la señal tiende a presentar variaciones en su pendiente. Esta correlación se refiere a que tan aproximadas son las dos señales bajo comparación.
2. Para un número de muestras superiores a 100, la correlación baja y las dos señales tienen diferencias, especialmente en amplitud cuando tienden a un valor constante.

En la figura 4.18 se ilustra la validación del modelo generado por el algoritmo, ante una entrada escalón unitario al sistema.

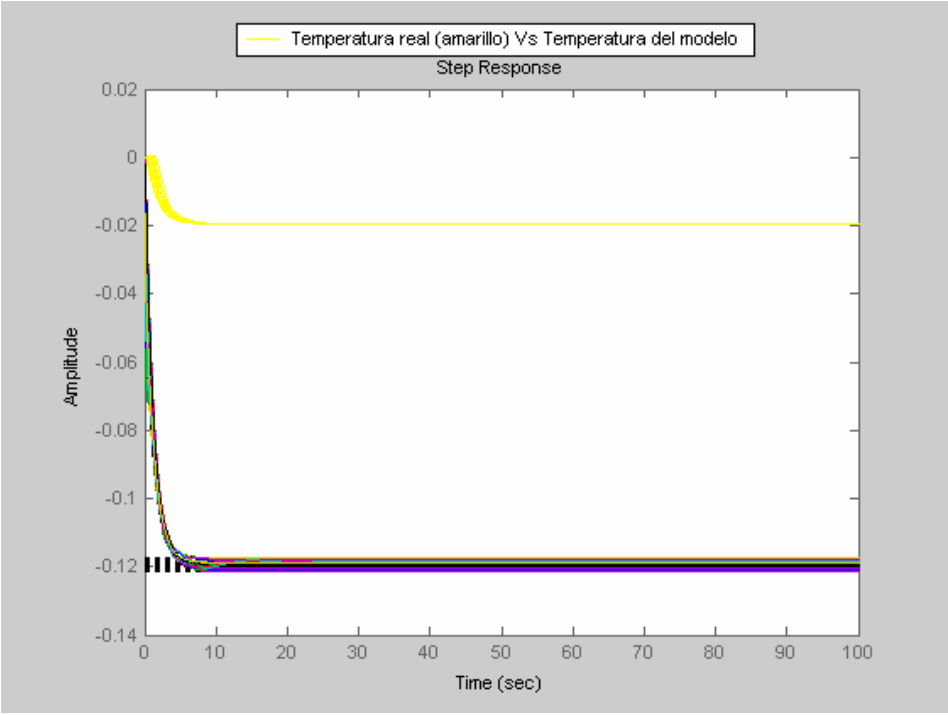

Fig. 4.18. Temperatura de la chaqueta para diferente orden del modelo.

De la figura 4.18 se pueden extraer las siguientes observaciones:

1. Hay una diferencia entre las amplitudes de las señales del modelo generado por el algoritmo y la señal de salida del reactor; este es un problema que puede ajustarse con la ganancia del modelo.
2. El comportamiento de las señales comparadas es similar dado que parten del mismo punto (cero) y con pendiente negativa se estabilizan alrededor de un punto (-0.12) a los 5 segundos aproximadamente.

4.4.2 Resultados del algoritmo de mínimos cuadrados ponderados. Para poder establecer una comparación sobre las mismas bases y variables de referencia, al igual que en el caso de mínimos cuadrados, se escogió el modelo ARX para hacer la implementación y aplicación del algoritmo de mínimos cuadrados ponderados. Se seleccionó el mismo rango de análisis de trescientos segundos (300s) para los tiempos de muestreo referenciados en el algoritmo de mínimos cuadrados.

4.4.2.1 Caso 1. Este caso se analiza para un tiempo de muestreo de 0.08 segundos. Durante un rango de observación de 300 segundos, se generaron 3750 datos, de los cuales se tomó la primera mitad de ellos (1875) para la construcción del modelo y la segunda mitad para la validación. Los resultados se muestran en la tabla 4.8.

Tabla 4.8. Modelos ARX con Ts=0.08

Numero de datos para construir el modelo	Orden del sistema	MINIMOS CUADRADOS PONDERADOS	
		Tiempo ejecución (segundos)	FPE
1875	2	1,8440	0,0137610
	3	3,8750	0,0189380
	4	6,2500	0,0224270
	5	8,4370	0,0244960
	6	10,7340	0,0254000
	7	13,0940	0,0254170
	8	15,5470	0,0249670
	9	18,0780	0,0244750
	10	20,7190	0,0241060

1. Del resultado de la simulación para una muestra de 1875 datos (tabla 4.8) en la figura 4.19 se puede observar la relación que hay entre la temperatura de la chaqueta a partir de los datos de salida del reactor (salida real) y la temperatura del proceso a partir del modelo generado por el algoritmo (salida del modelo).

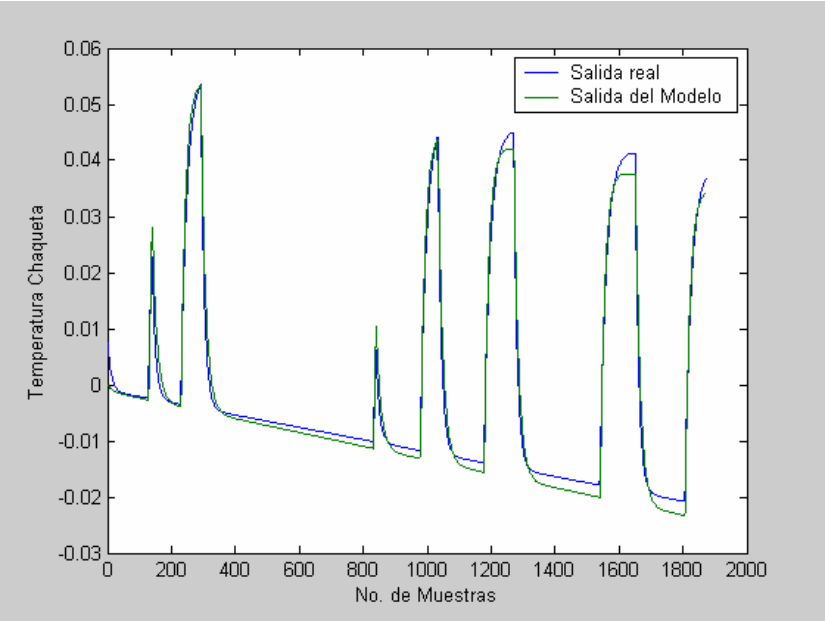

Fig. 4.19. Temperatura de la chaqueta del reactor (azul) y la temperatura del modelo (verde).

De la figura 4.19 se pueden extraer las siguientes observaciones:

1. Para las primeras 400 muestras hay una relativa alta correlación entre las dos señales de salida, especialmente en aquellos puntos donde el valor de la señal tiende a un valor constante.
2. Para un número de muestras superiores a 400, la correlación entre las dos señales tiene a disminuir, especialmente en los puntos de menor pendiente.

En la figura 4.20 se ilustra la validación del modelo generado por el algoritmo, ante una entrada escalón unitario al sistema.

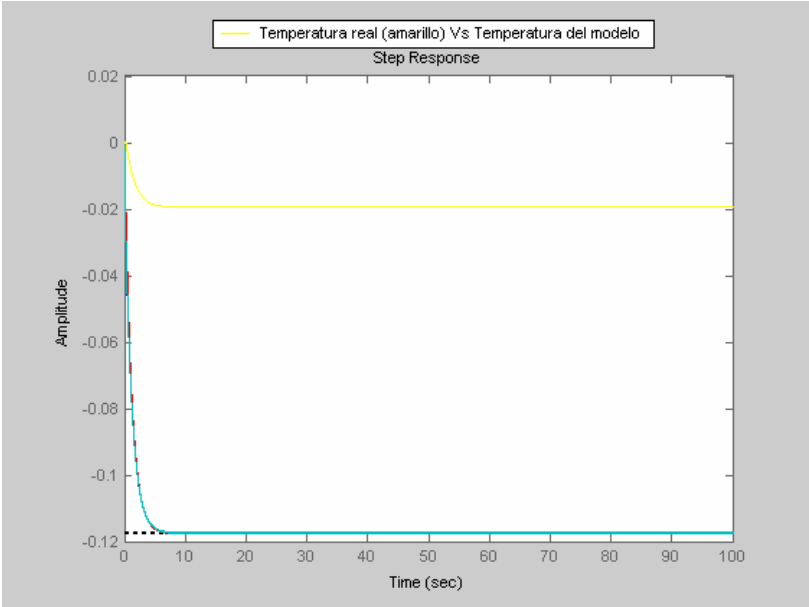

Fig. 4.20. Temperatura de la chaqueta para diferente orden del modelo.

De la figura 4.20 se pueden extraer las siguientes observaciones:

1. Hay una diferencia entre las amplitudes de las señales del modelo generado por el algoritmo y la señal de salida del reactor; este es un problema que puede ajustarse con la ganancia del modelo.
2. El comportamiento de las señales comparadas es similar dado que parten del mismo punto (cero) y con pendiente negativa se estabilizan alrededor de un punto (aproximadamente -0,12) a los 5 segundos.

4.4.2.2 Caso 2. Este caso se analiza para un tiempo de muestreo de 0.1 segundos. Durante un rango de observación de 300 segundos, se generaron 3000 datos, de los cuales se tomó la primera mitad de ellos (1500) para la construcción del modelo y la segunda mitad para la validación. Los resultados se muestran en la tabla 4.9.

Tabla 4.9. Modelos ARX con Ts=0.1 s

Numero de datos para construir el modelo	Orden del sistema	MINIMOS CUADRADOS PONDERADOS	
		Tiempo ejecución (segundos)	FPE
1500	2	1,3910	0,0223260
	3	3,2340	0,0302160
	4	5,0470	0,0352960
	5	6,9530	0,0381080
	6	8,8910	0,0391170
	7	11,0940	0,0389140
	8	13,3750	0,0382630
	9	15,5620	0,0376710
	10	17,7810	0,0037250

Del resultado de la simulación para una muestra de 1500 datos (tabla 4.9) en la figura 4.21 se puede observar la relación que hay entre la temperatura de la

chaqueta a partir de los datos de salida del reactor (salida real) y la temperatura del proceso a partir del modelo generado por el algoritmo (salida del modelo).

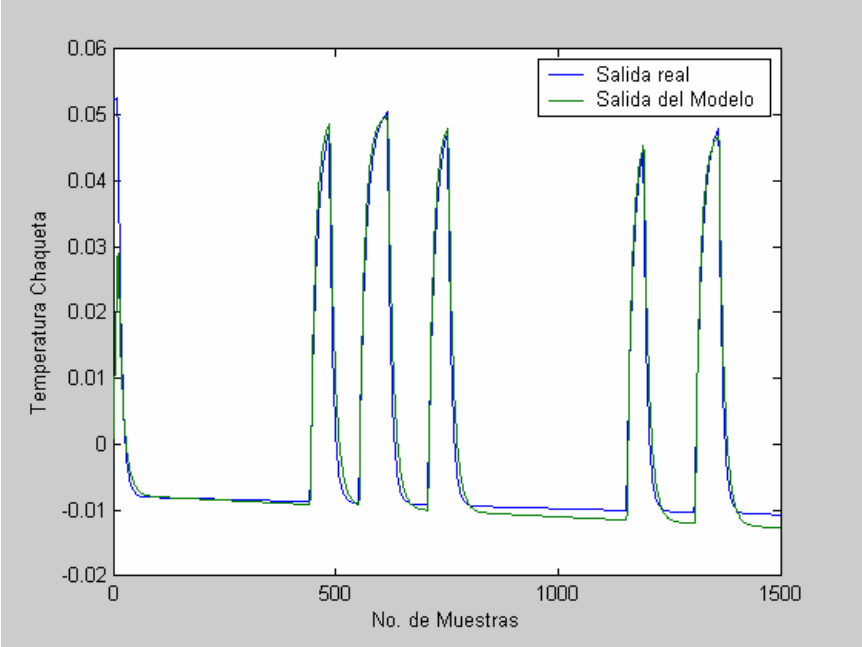

Fig. 4.21. Temperatura de la chaqueta del reactor (azul) y la temperatura del modelo (verde).

De la figura 4.21 se pueden extraer las siguientes observaciones:

1. Para las primeras 700 muestras hay una relativa alta correlación entre las dos señales de salida.
2. Para un número de muestras superiores a 700, la correlación entre las dos señales tiene a disminuir, especialmente en los puntos de menor pendiente.

En la figura 4.22 se ilustra la validación del modelo generado por el algoritmo, ante una entrada escalón unitario al sistema.

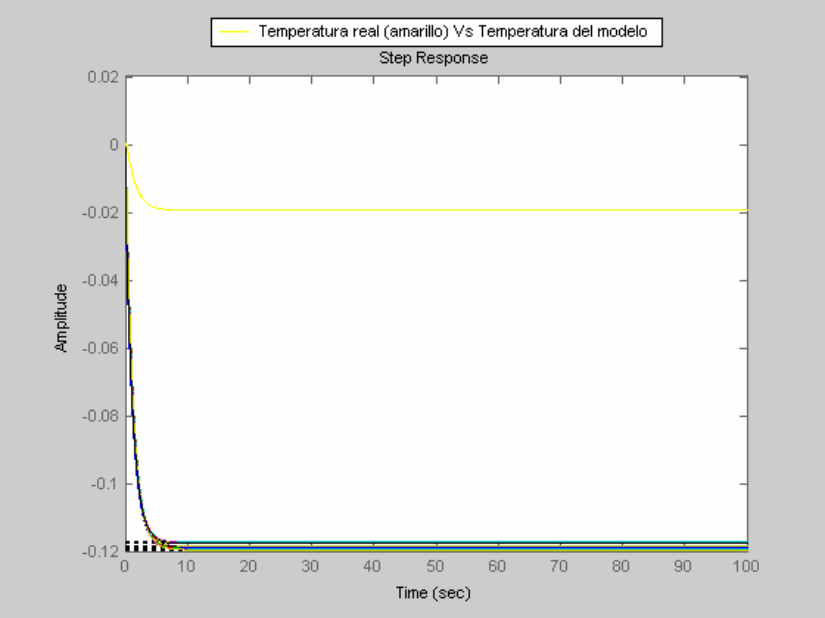

Fig. 4.22. Temperatura de la chaqueta para diferente orden del modelo.

De la figura 4.22 se pueden extraer las siguientes observaciones:

1. Hay una diferencia entre las amplitudes de las señales del modelo generado por el algoritmo y la señal de salida del reactor; este es un problema que puede ajustarse con la ganancia del modelo.
2. El comportamiento de las señales comparadas es similar dado que parten del mismo punto (cero) y con pendiente negativa se estabilizan alrededor de un punto (aproximadamente -0,12) a los 10 segundos.

4.4.2.3 Caso 3. Este caso se analiza para un tiempo de muestreo de 0.2 segundos. Durante un rango de observación de 300 segundos, se generaron 1500 datos, de los cuales se tomó la primera mitad de ellos (750) para la construcción del modelo y la segunda mitad para la validación. Los resultados se muestran en la tabla 4.10.

Tabla 4.10. Modelos ARX con Ts=0.2 s

Numero de datos para construir el modelo	Orden del sistema	MINIMOS CUADRADOS PONDERADOS	
		Tiempo ejecución (segundos)	FPE
750	2	0,7030	0,0478510
	3	2,0780	0,0564910
	4	3,4850	0,0582130
	5	4,9220	0,0573420
	6	6,5000	0,0566090
	7	8,0160	0,0566710
	8	9,6880	0,0575830
	9	11,3280	0,0592220
	10	13,1100	0,0614940

Del resultado de la simulación para una muestra de 750 datos (tabla 4.10) en la figura 4.23 se puede observar la relación que hay entre la temperatura de la chaqueta a partir de los datos de salida del reactor (salida real) y la temperatura del proceso a partir del modelo generado por el algoritmo (salida del modelo).

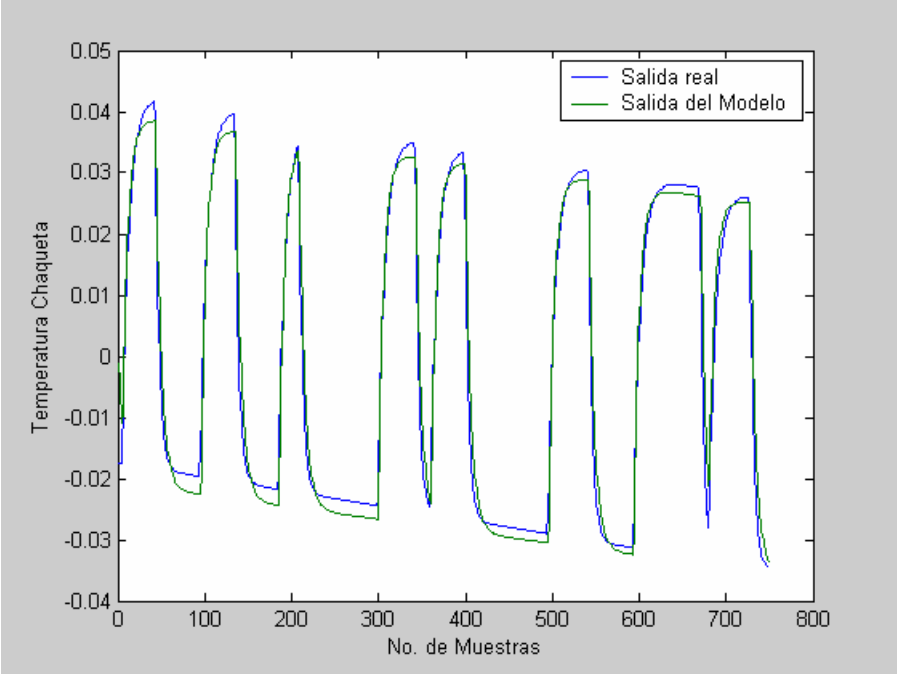

Fig. 4.23. Temperatura de la chaqueta del reactor (azul) y la temperatura del modelo (verde).

De la figura 4.23 se puede apreciar que la correlación es relativamente alta en los puntos de mayor pendiente, pero que en cambio es relativamente baja para los puntos donde las señales tienden hacia una pendiente pequeña o hacia un valor constante.

En la figura 4.24 se ilustra la validación del modelo generado por el algoritmo, ante una entrada escalón unitario al sistema. Como puede apreciarse, los modelos generados por el algoritmo presentan un comportamiento cualitativo parecido al de la salida del proceso.

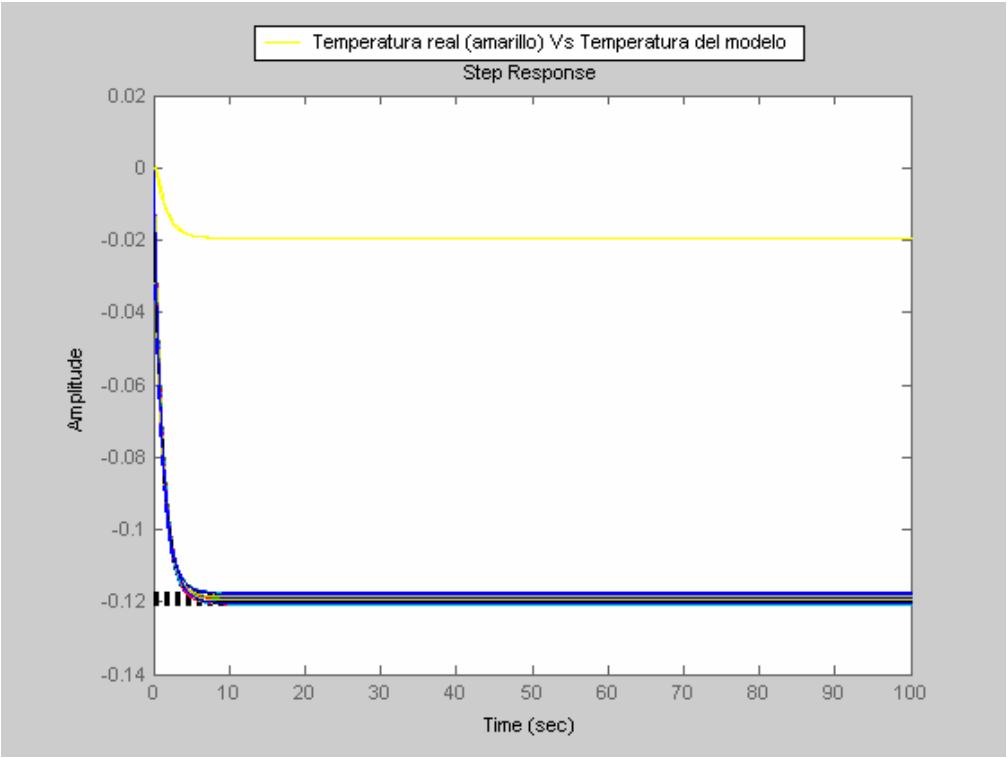

Fig. 4.24. Temperatura de la chaqueta para diferente orden del modelo.

4.4.2.4 Caso 4. Este caso se analiza para un tiempo de muestreo de 0.4 segundos. Durante un rango de observación de 300 segundos, se generaron 750 datos, de los cuales se tomó la primera mitad de ellos (375) para la construcción del modelo y la segunda mitad para la validación. Los resultados se muestran en la tabla 4.11.

Del resultado de la simulación para una muestra de 375 datos (tabla 4.11) en la figura 4.25 se puede observar la relación que hay entre la temperatura de la chaqueta a partir de los datos de salida del reactor (salida real) y la temperatura del proceso a partir del modelo generado por el algoritmo (salida del modelo).

Tabla 4.11. Modelos ARX con Ts=0.4 s

Numero de datos para construir el modelo	Orden del sistema	MINIMOS CUADRADOS	
		Tiempo ejecución (segundos)	FPE
375	2	0,4220	0,0910420
	3	3,1870	0,0917330
	4	5,8750	0,0909570
	5	8,4530	0,0930950
	6	11,1250	0,0977490
	7	13,8590	0,1044700
	8	16,6720	0,1127600
	9	19,5470	0,1222400
	10	22,5470	0,1326500

De la figura 4.25, al igual que en el caso anterior, se puede apreciar que la correlación es relativamente alta en los puntos de mayor pendiente, pero que en cambio es relativamente baja para los puntos donde las señales tienden hacia una pendiente pequeña o hacia un valor constante.

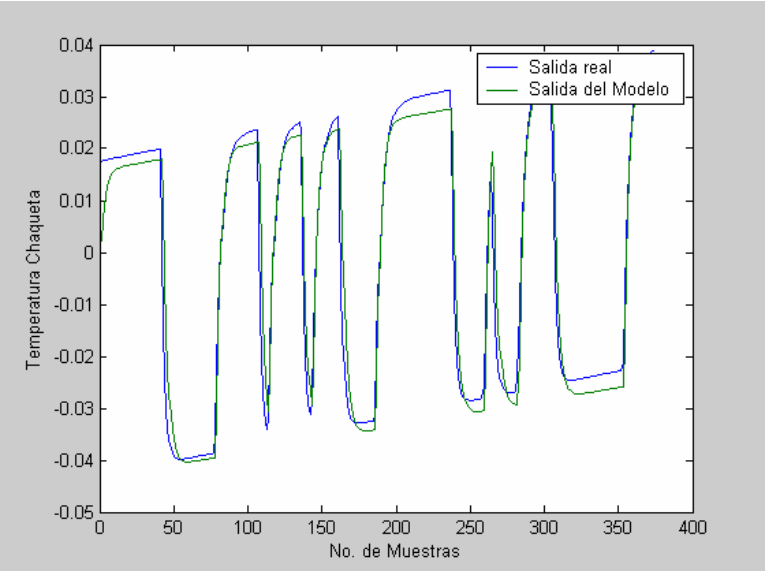

Fig. 4.25. Temperatura de la chaqueta del reactor (azul) y la temperatura del modelo (verde).

En la figura 4.26 se ilustra la validación del modelo generado por el algoritmo, ante una entrada escalón unitario al sistema.

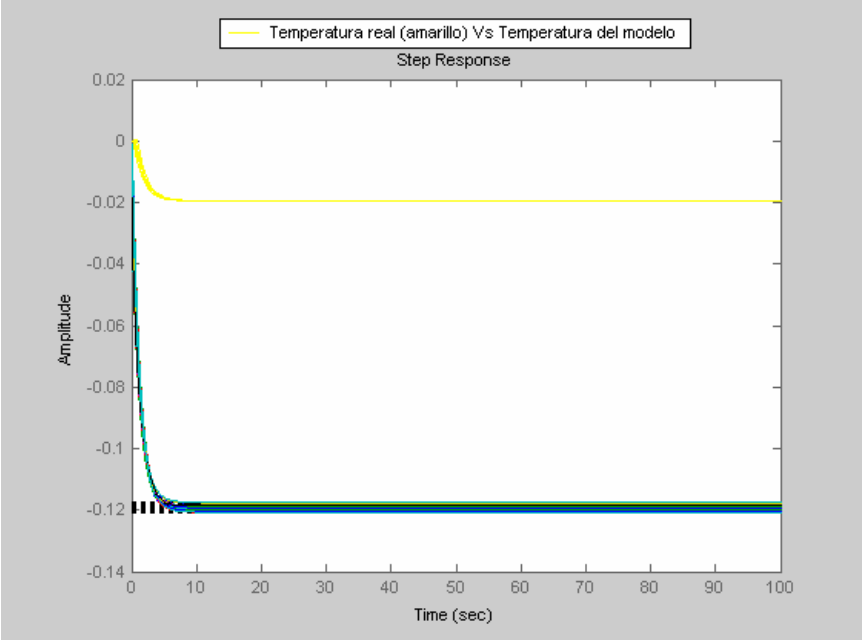

Fig. 4.26. Temperatura de la chaqueta para diferente orden del modelo.

De la figura 4.26 se pueden extraer las siguientes observaciones:

1. Hay una diferencia entre las amplitudes de las señales del modelo generado por el algoritmo y la señal de salida del reactor; este es un problema que puede ajustarse con la ganancia del modelo.
2. El comportamiento de las señales comparadas es similar dado que parten del mismo punto (cero) y con pendiente negativa se estabilizan alrededor de un punto (aproximadamente -0,12) a los 8 segundos.

4.4.2.5 Caso 5. Este caso se analiza para un tiempo de muestreo de 0.8 segundos. Durante un rango de observación de 300 segundos, se generaron

375 datos, de los cuales se tomó la primera mitad de ellos (188) para la construcción del modelo y la segunda mitad para la validación. Los resultados se muestran en la tabla 4.12.

Tabla 4.12. Modelos ARX con Ts=0.8 s

Numero de datos para construir el modelo	Orden del sistema	MINIMOS CUADRADOS	
		Tiempo ejecución (segundos)	FPE
188	2	0,5310	0,1542600
	3	5,9060	0,1629700
	4	11,3130	0,0185900
	5	16,8910	0,2170400
	6	22,5630	0,2535900
	7	28,3440	0,2922600
	8	34,2500	0,3347500
	9	40,2810	0,3796200
	10	46,3440	0,4268600

Del resultado de la simulación para una muestra de 188 datos (tabla 4.12) en la figura 4.27 se puede observar la relación que hay entre la temperatura de la chaqueta a partir de los datos de salida del reactor (salida real) y la temperatura del proceso a partir del modelo generado por el algoritmo (salida del modelo).

De la figura 4.27 se puede apreciar que la correlación es relativamente alta en los puntos de mayor pendiente, pero que en cambio es relativamente baja para los puntos donde las señales tienden hacia una pendiente pequeña o hacia un valor constante.

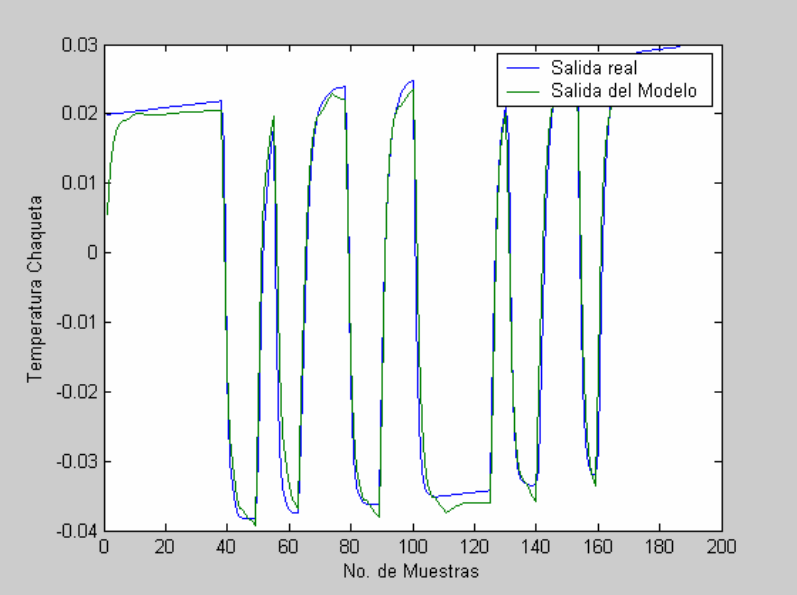

Fig. 4.27. Temperatura real (azul) y la temperatura del modelo (verde).

En la figura 4.28 se ilustra la validación del modelo generado por el algoritmo, ante una entrada escalón unitario al sistema.

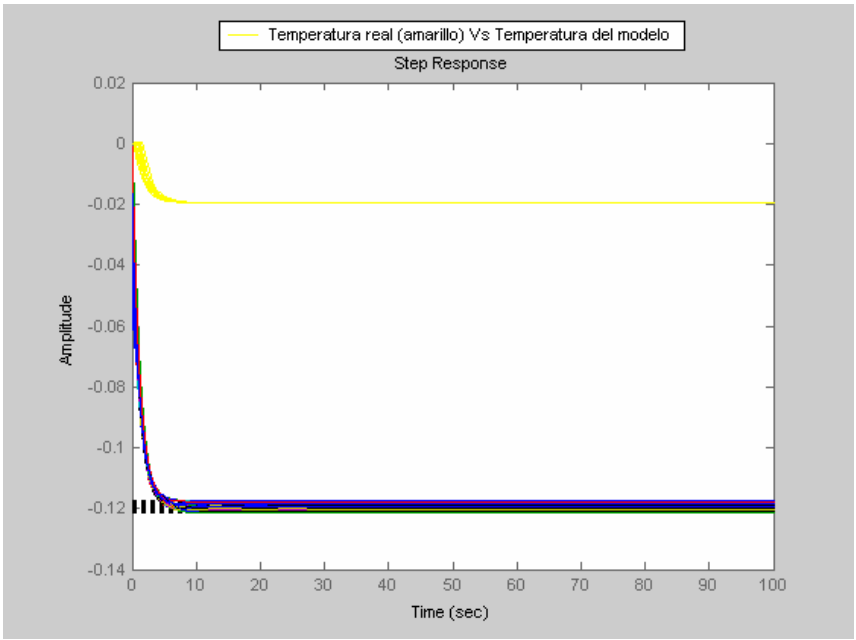

Fig. 4.28. Temperatura de la chaqueta para diferente orden del modelo.

De la figura 4.28 se pueden extraer las siguientes observaciones:

1. Hay una diferencia entre las amplitudes de las señales del modelo generado por el algoritmo y la señal de salida del reactor; este es un problema que puede ajustarse con la ganancia del modelo.
2. El comportamiento de las señales comparadas es similar dado que parten del mismo punto (cero) y con pendiente negativa se estabilizan alrededor de un punto (aproximadamente -0,12) a los 8 segundos.

Al hacer la comparación entre los dos algoritmos implementados, en términos de los tiempos de ejecución de cada uno de ellos, es evidente que el algoritmo de mínimos cuadrados requiere de un menor tiempo para llegar a la solución de un modelo de diferente orden, para el mismo número de datos, que el tiempo requerido por el algoritmo de mínimos cuadrados ponderado implementado.

4.5 EFECTO DE UNA PERTURBACIÓN EN LA SEÑAL DE ENTRADA

Las características estadísticas de los métodos de mínimos cuadrados han sido bien estudiadas en la literatura a través de formalismos de teoría de probabilidad. El lector puede consultar [18] y las referencias citadas allí para una descripción completa. En esta sección analizaremos cualitativamente el

comportamiento de los algoritmos de identificación ante perturbaciones aleatorias en la señal de entrada.

Para analizar el comportamiento de la robustez del algoritmo, en la generación del modelo, a la señal pseudoaleatoria se le adiciona una perturbación o señal de ruido. En este caso, la perturbación corresponde a una señal de ruido, la cual es comúnmente utilizada en este tipo de análisis. Las características de la perturbación son las siguientes:

Tipo de señal:	ruido con media cero y función de densidad de probabilidad normal.
Potencia de la señal:	0.0001
Tiempo de muestreo:	Tiempo de muestreo de la señal de entrada (Ts)

En la figura 4.29 se ilustra la perturbación adicionada al sistema.

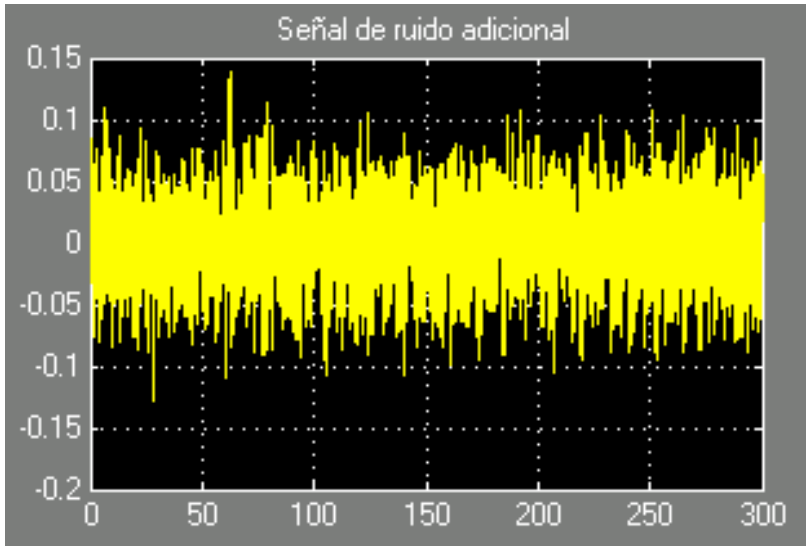

Fig. 4.29. Perturbación adicionada a la señal de entrada

Esta señal tiene una amplitud que oscila entre +- 0.13 en un rango de muestreo de 300 segundos.

Como resultado de adicionar esta perturbación a la señal de entrada pseudoaleatoria, en la figura 4.30 puede observarse la nueva señal de entrada al reactor con perturbaciones incluidas.

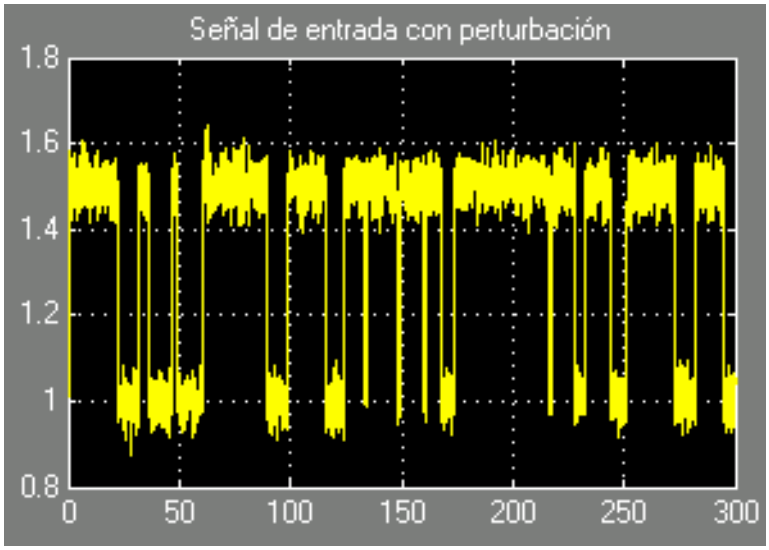

Fig. 4.30 Señal de entrada al CSTR con perturbaciones

Comparativamente, las amplitudes de la perturbación con la amplitud de la señal tienen una relación de 1 a 10 aproximadamente.

Para comparar los algoritmos se escogió un caso de los estudiados en las sesiones 4.4.1 y 4.4.2, el que corresponde a un tiempo de muestreo de 0.08 segundos.

4.5.1 Algoritmo de mínimos cuadrados. En este experimento, se seleccionaron los datos generados por el reactor ante una entrada con perturbaciones como la que se indica en la figura 4.30.

Se analiza la respuesta para un tiempo de muestreo de 0.08 segundos. En la figura 4.31 se observa la relación que hay entre la salida del modelo de predicción (verde) y la señal de salida del reactor (azul).

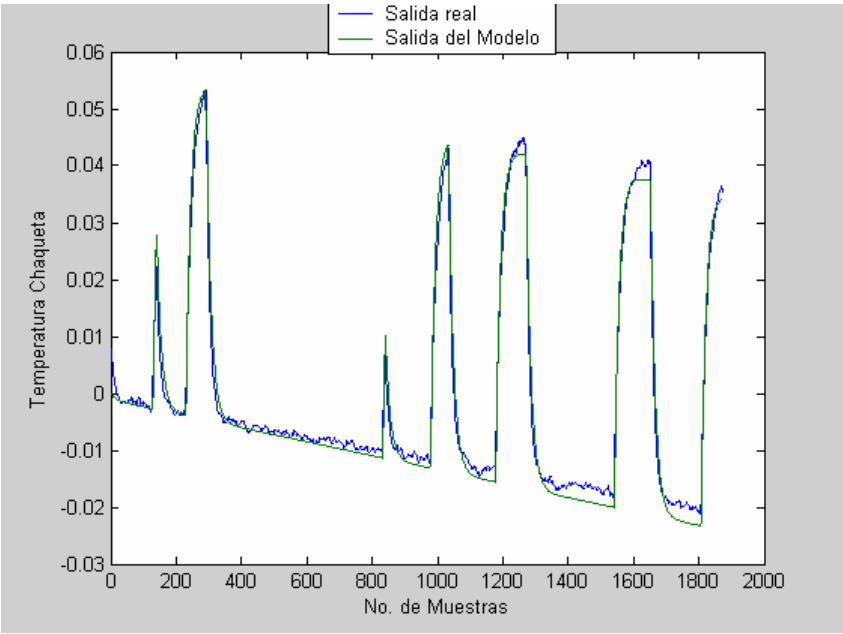

Fig. 4.31. Salida del modelo de predicción vs salida del reactor

Como puede observarse, se presenta una buena correlación de los datos del modelo de predicción, excepto en los puntos donde la señal tiende a tomar valores constantes. Esto significa que sigue adecuadamente la dinámica en los puntos de mayor variación.

Para hacer la validación del modelo de predicción obtenido y compararlo a su vez con la respuesta que proporciona CSTR, se les aplica una señal escalón tanto a los modelos de predicción para diferente orden como al modelo de la planta original. El resultado de la salida comparada de los modelos obtenidos se muestra en la gráfica 4.32

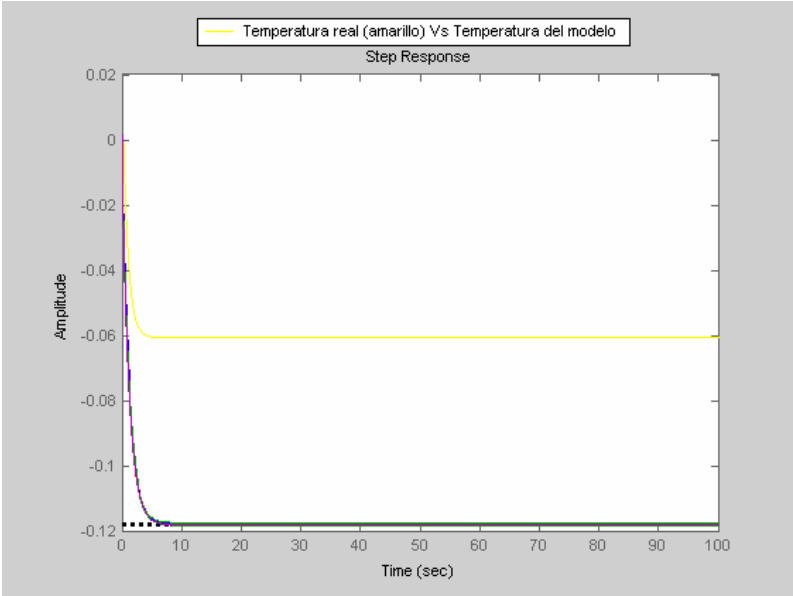

Fig. 4.32. Temperatura de la planta (amarillo) vs temperatura del modelo de predicción

El comportamiento de la temperatura del modelo de predicción generado por el algoritmo tiene la misma forma del comportamiento de la temperatura de la planta; sin embargo hay una diferencia en amplitud, la cual puede mejorarse con un ajuste de la ganancia en estado estacionario. Para mostrar un ejemplo,, si se ajusta la ganancia de los modelos de predicción por un factor de 0.5

(resultado de dividir -0.06/-0.12), se obtiene el resultado que se ilustra en la figura 4.33.

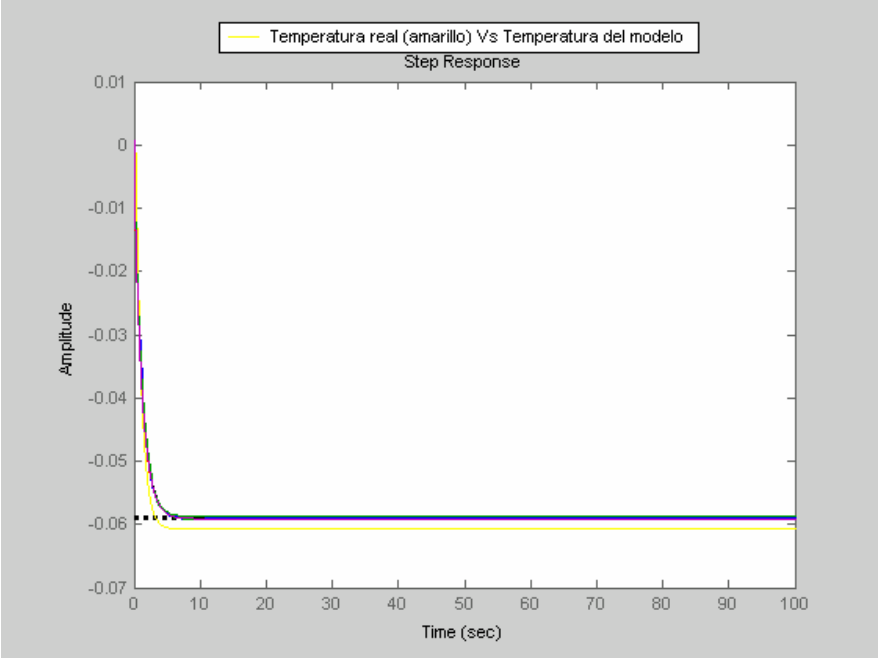

Fig. 4.33. Modelos de predicción con ganancia ajustada en 0.5

Como puede observarse en la figura 4.33, con el ajuste de la ganancia en los modelos de predicción, su salida tiende a seguir mucho más de cerca el comportamiento de la salida de la planta, el cual es el objetivo final

4.5.2 Algoritmo de mínimos cuadrados ponderados. En este experimento se seleccionaron los datos generados por el reactor ante una entrada con perturbaciones como la que se indica en la figura 4.30.

Se analiza la respuesta para un tiempo de muestreo de 0.08 segundos. Con la señal de entrada (figura 4.30) aplicada a la planta original se generan los datos para la construcción del modelo del CSTR. En la figura 4.34 se observa la relación que hay entre la salida del modelo de predicción (verde) y la señal de salida del reactor (azul).

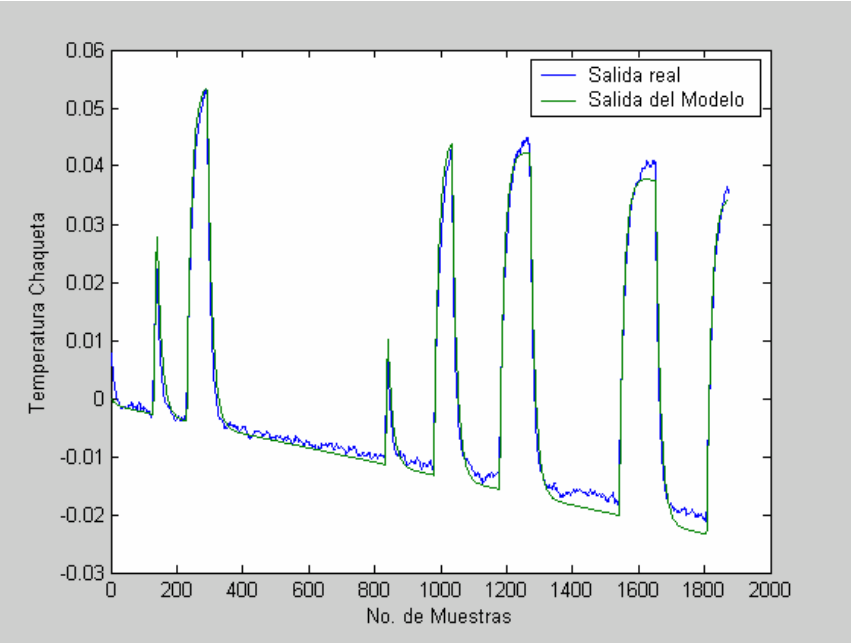

Fig.4.34. Salida del modelo de predicción (verde) vs salida del reactor (azul)

Como puede observarse en la figura 4.34, se presenta una buena correlación de los datos del modelo de predicción, excepto en los puntos donde la señal tiende a tomar valores constantes (máximos y mínimos). Esto significa que sigue adecuadamente la dinámica en los puntos de mayor variación, de manera similar a como lo registró el modelo obtenido con el algoritmo de mínimos cuadrados.

Para hacer la validación del modelo de predicción obtenido y compararlo a su vez con la respuesta que proporciona el CSTR, se le aplica una señal escalón tanto a los modelo de predicción para diferente orden como al modelo del reactor. El resultado de la salida comparada de los modelos obtenidos se muestra en la gráfica 4.35.

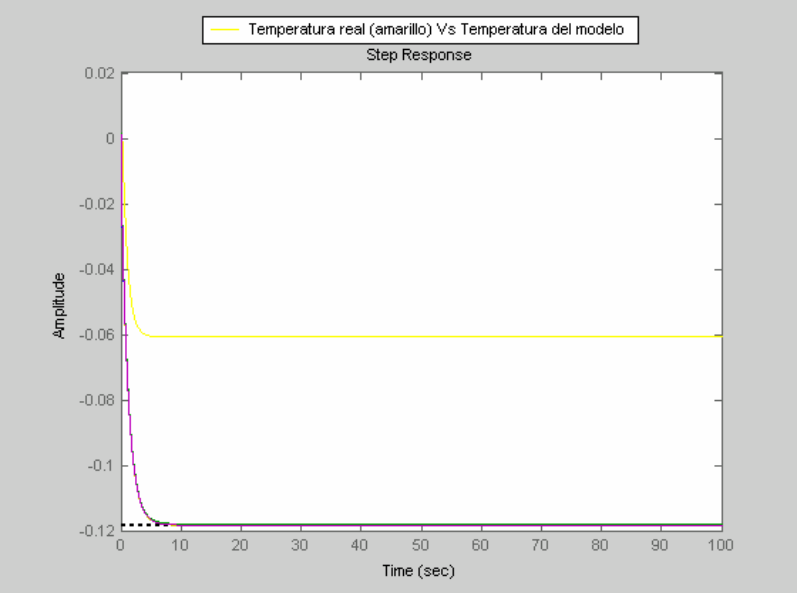

Fig. 4.35. Temperatura de la planta (amarillo) vs temperatura del modelo de predicción

Al igual que en el caso anterior, la respuesta del modelo de predicción ante una entrada escalón tiene la misma forma que la respuesta del reactor.

A partir de los resultados descritos en este apartado, se evidencia como la salida de los modelos ARX obtenidos por los algoritmos de mínimos cuadrados y mínimos cuadrados ponderados tienden a seguir la salida del reactor, en este caso la temperatura de la chaqueta.

4.6 VALIDACIÓN DE LOS MODELOS ANTE UNA ENTRADA IMPULSO

Para validar los modelos obtenidos con una señal de entrada diferente a la señal pseudoaleatoria, se aplicó una señal impulso a los modelos de predicción para observar el comportamiento de la salida y compararla con la salida del reactor. La señal impulso es una señal que se aplica tan solo por un instante de tiempo sumamente pequeño y luego desaparece. En sistemas lineales la validez de un modelo a una entrada impulso se extrapola directamente a una entrada escalón, rampa, entre otras, ver [57].

En la figura 4.36 se ilustra el resultado para la simulación del algoritmo de mínimos cuadrados con un tiempo de muestreo de 0.08 segundos aplicado al los modelos obtenidos en la sección 4.4.1.

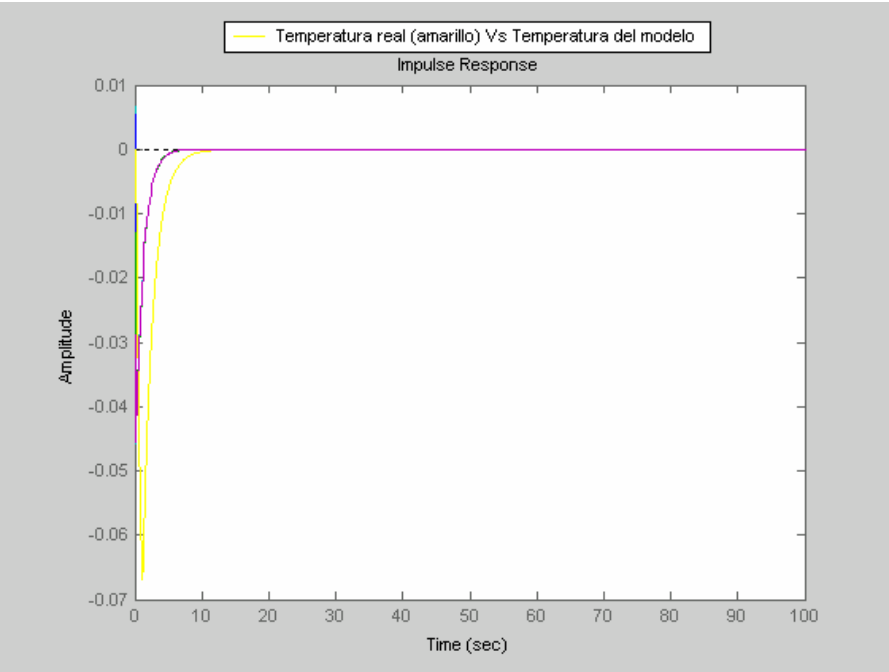

Fig. 4.36. Respuesta de los modelos ante una entrada impulso

Como se observa en la figura 4.36, nuevamente aparece una diferencia en amplitud entre la temperatura de la chaqueta del reactor y la temperatura de los modelos de predicción, la cual se corrige con un ajuste de la ganancia en estado estable como ya se ilustró en el caso para la figura 4.33. Las señales presentan un comportamiento similar tienden a estabilizarse alrededor de un valor de cero cuando han transcurrido aproximadamente 10 segundos.

Este resultado evidencia que los modelos obtenidos con la aplicación del algoritmo de mínimos cuadrados se pueden utilizar para predecir el comportamiento del sistema con una señal diferente de la señal pseudoaleatoria de entrada.

En este capítulo inicialmente se presentó el modelo de análisis con todos sus parámetros; posteriormente se desarrolló la aplicación de los algoritmos variando el número de datos muestreados. Se puede evidenciar de los resultados la obtención de modelos válidos y la robustez de los algoritmos, aún en el caso en el que a la señal de entrada se le agregan perturbaciones.

CAPITULO V.- ANALISIS DE LOS RESULTADOS

En este capítulo se hace un análisis de los resultados obtenidos en el capítulo IV enfocado especialmente a los aspectos computacionales y los criterios de comparación planteados en el capítulo anterior.

5.1 ANÁLISIS DE LA COMPLEJIDAD DE LOS ALGORITMOS

Para hacer al análisis y la determinación de la complejidad de los algoritmos hicimos un estudio para determinar su función de trabajo (análisis) y culminamos con la determinación de su eficiencia (complejidad).

Hallar una función de trabajo consiste en obtener una ecuación que permita calcular la tendencia de los resultados del algoritmo. El cálculo de la función de trabajo se hizo mediante el análisis estadístico de los datos colectados. A través del análisis de regresión se obtuvieron diferentes curvas de ajuste como resultado. Se pueden obtener tantas curvas como sea posible de acuerdo con el método que se utilice; por ejemplo, con el método de mínimos cuadrados se pueden obtener curvas

- Lineales $(y = cx + b)$
- Logarítmicas $(y = c \ln x + b)$
- Polinomiales $(y = c_n x^n + c_{n-1} x^{n-1} + ... + c_1 x + b$

- Potenciales $(y = cx^b)$
- Exponenciales $(y = ce^{bx})$

En esta sección se hace un análisis estadísticos de los datos para comparar los resultados de los algoritmos implementados.

5.1.1 Regresión. La *recta de regresión* (como también es llamado el caso lineal) es el ejemplo más sencillo de la aplicación del método de mínimos cuadrados. Para obtenerla sólo debe resolverse el sistema de ecuaciones

$$c\sum x + bn = \sum y \qquad (5.1)$$

$$c\sum x^2 + b\sum x = \sum xy \qquad (5.2)$$

Donde *n* es el número de datos de la muestra. Revisaremos una aproximación lineal para un sistema de orden dos tanto para el algoritmo de mínimos cuadrados como para el de mínimos cuadrados ponderados. Los demás resultados serán consignados en tablas.

5.1.1.1 Análisis de regresiones para determinación de complejidad computacional del algoritmo de mínimos cuadrados. Los datos tomados de la obtención del modelo paramétrico con el algoritmo de mínimos cuadrados se pueden observar en la tabla 5.1, donde y representa el tiempo que demora

en ejecutarse el algoritmo para un modelo de orden 2 y x representa el número de datos de entrada al algoritmo para ese orden del modelo.

Tabla 5.1. Datos obtenidos para un modelo ARX de orden 2

Datos	
x	**y**
167	0,453
188	0,484
214	0,516
250	0,516
300	0,422
375	0,469
500	0,484
750	0,641
1500	0,594
1875	0,719

Aplicando los datos de la tabla 5.1 en (5.1) y (5.2) para el caso polinomial de grado 3 ($y = c_3x^3 + c_2x^2 + c_1x + c_0$), tenemos:

Tabla 5.2. Valores para las ecuaciones

Sistema de ecuaciones							
Incógnitas		**C3**	**C2**	**C1**	**C0**		
	Σx^3	0,01563	6,9803	6,119	10	5,298 Σy	
Matriz de	Σx^4	17,8369	10,63	6,98	6,12	3,67 Σxy	**Vector de**
Constantes	Σx^5	31,0482	17,837	10,63	6,98	4,535 Σx^2y	**Resultados**
	Σx^6	55,0399	31,048	17,84	10,6	7,13 Σx^3y	

y resolviendo el sistema de ecuaciones obtenemos la ecuación buscada :

$$y = 0.0011x^3 - 0.0127x^2 + 0.0456 + 0.4326 \qquad (5.5)$$

En la figura 5.1 pueden observarse los valores reales y la aproximación para este caso particular.

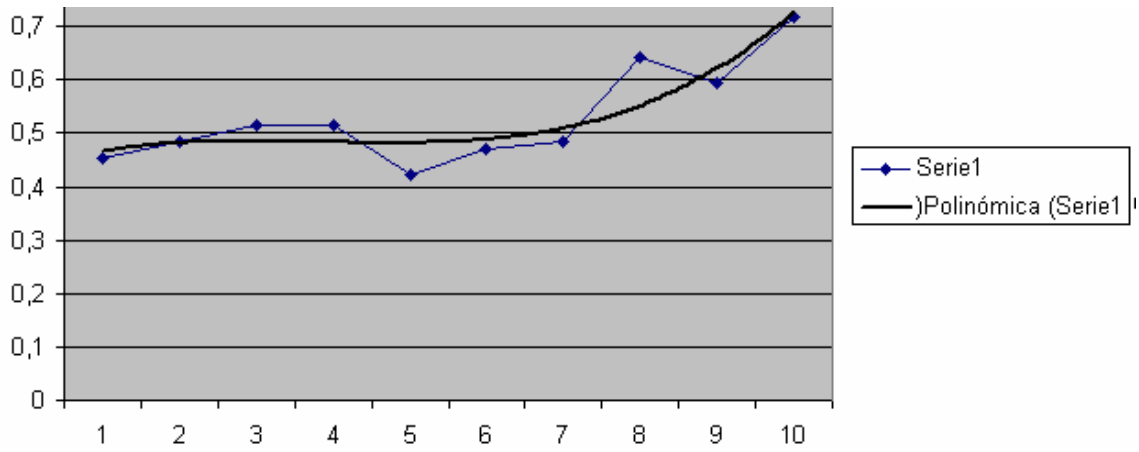

Fig. 5.1. Recta de regresión para modelo de orden 2.

Mediante el *factor de correlación*, *r*, puede determinarse qué tan bien explica la ecuación obtenida al conjunto de datos, dentro de un valor en el rango [-1, 1], que podemos determinar por la fórmula:

$$r = \frac{\sum (y - \bar{y})(\hat{y} - \bar{\hat{y}})}{\sqrt{\sum (y - \bar{y})^2 (\hat{y} - \bar{\hat{y}})^2}}$$

donde y corresponde a los valores originales, $\hat{y}$ a los valores calculados, y la barra sobre cada variable denota el promedio de cada conjunto de datos. Mientras más cercana sea *r* a ±1 más precisamente nuestra ecuación podrá

reproducir los datos reales. En la práctica se toma como aceptable un valor superior a ±0.8 pero todo depende del problema y precisión buscada, ver [51].

Este modelo de mínimos cuadrados se puede generalizar para los casos planteados en la sección 4.4.1. Después de aplicar la generalización se obtuvieron los resultados indicados en la tabla 5.3.

Tabla 5.3. Coeficiente de correlación para modelos de diferente orden

Orden del sistema	MINIMOS CUADRADOS				
	Lineal	Polinómica orden 3	Logaritmica	Exponencial	Potencial
2	0,540	0,800	0,360	0,520	0,360
3	0,870	0,990	0,949	0,914	0,915
4	0,884	0,996	0,960	0,927	0,905
5	0,880	0,998	0,966	0,927	0,909
6	0,885	0,998	0,972	0,936	0,911
7	0,898	0,998	0,977	0,945	0,913
8	0,894	0,999	0,979	0,952	0,912
9	0,901	0,999	0,978	0,958	0,912
10	0,904	0,999	0,980	0,962	0,912

De acuerdo con los datos obtenidos en la tabla 5.3 podemos concluir:

- La regresión que mejor representa los datos es la correspondiente a la ecuación polinomial de grado 3 para todos los órdenes del sistema. El valor del coeficiente de correlación R^2 es de 0.8 para un orden del sistema de 2 y para los órdenes del sistema de 3 a 10 permanece en el rango de 0.99 a 1.

- Para los órdenes del sistema superiores a 2 el comportamiento de la complejidad muestra que esta decrece a medida que aumenta el número de datos hasta que se consigue un mínimo en el rango de 1500 y 1875 número de datos.

5.1.1.2 Análisis de regresiones para determinación de complejidad computacional del algoritmo de mínimos cuadrados ponderados. Aplicando el mismo concepto desarrollado en la sección 5.1.1.1, se obtuvieron los siguientes resultados para el algoritmo de mínimos cuadrados ponderados:

Tabla 5.4. Coeficiente de correlación para modelos de diferente orden

Orden del sistema	MINIMOS CUADRADOSPONDERADOS				
	Lineal	Polinómica orden 3	Logaritmica	Exponencial	Potencial
2	0,377	0,974	0,341	0,606	0,377
3	0,650	0,976	0,815	0,578	0,693
4	0,750	0,985	0,880	0,687	0,761
5	0,789	0,989	0,905	0,937	0,794
6	0,886	0,991	0,915	0,765	0,810
7	0,814	0,991	0,918	0,773	0,814
8	0,818	0,991	0,921	0,780	0,819
9	0,822	0,991	0,923	0,785	0,821
10	0,824	0,991	0,926	0,790	0,826

De acuerdo con los datos obtenidos en la tabla 5.4 podemos concluir:

- La regresión que mejor representa los datos es la correspondiente a la ecuación polinomial de grado 3 para todos los órdenes del sistema. El valor del coeficiente de correlación R^2 permanece en el rango de 0.97 a 0.99.

- Para los órdenes del sistema superiores a 2 el comportamiento de la complejidad muestra que esta decrece a medida que aumenta el número de datos hasta que se consigue un mínimo en el rango de 750 a 1500 número de datos.

5.1.2 Conclusiones sobre resultados. La complejidad computacional del algoritmo es indicada por el *orden* de su función de trabajo. Existen dos formas de cómo denotar este orden y que son conocidas como "gran *O*" y "pequeña *o*". En la práctica se suele identificar a la *gran O* como sinónimo del orden de la función de trabajo del algoritmo, sin embargo ambas medidas son válidas e identifican la forma en que una función varía en magnitud a medida que sus parámetros tienden a un límite. Su definición es como sigue. Una función *f*(*n*) se dice que es de orden *O*(*n*) si cumple la siguiente condición

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = cons\tan te \neq 0$$

en ambas expresiones *f*(*n*) denota la función de trabajo del algoritmo, *g*(*n*) son funciones propuestas que se obtienen, respectivamente, del término de la función con mayor crecimiento (el que define propiamente el comportamiento del algoritmo) y de cualquier otra función con un crecimiento mayor.

Para nuestro caso se tiene que de acuerdo al análisis de regresión se obtiene que la complejidad de los algoritmos implementados se pueda representar con una ecuación del tipo:

$$f(n) = a\chi^3 + b\chi^2 + c\chi + d$$

Y tomando:

$$g(n) = \chi^3$$

Al calcular el límite de la ecuación anterior se obtiene que el valor límite es a, por lo que se puede concluir que el algoritmo presenta una complejidad $O(n^3)$ y es un algoritmo polinomial tipo P. Todo algoritmo tipo P se considera eficiente debido a que su complejidad es inferior a los algoritmos exponenciales (tipo NP).

En resumen, ambos algoritmos presentan la misma complejidad computacional $O(n^3)$ por lo que el desempeño de ellos debe medirse por el que genere un menor error de predicción.

Lo ideal es utilizar el número de datos que nos de el valor mínimo de complejidad, siempre y cuando el valor del FPE sea adecuado.

5.2 CRITERIO DEL ERROR FINAL DE PREDICCIÓN (FPE)

El FPE fue explicado ampliamente en la sección 3.1.1 de este trabajo. El FPE se tomó como uno de los criterios de comparación de los algoritmos implementados, por lo que en esta sección se presentan los resultados obtenidos con su aplicación.

5.2.1 Error final de predicción (FPE) para el algoritmo de mínimos cuadrados. En la tabla 5.5 se presentan los resultados del error de predicción final en la ejecución del algoritmo de Mínimos Cuadrados.

Tabla 5.5 Orden del sistema, número de datos y FPE (Mínimos Cuadrados)

ORDEN	2	3	4	5	6	7	8	9	10	DATOS
FPE PARA MINIMOS CUADRADOS	0,0126750	0,0166670	0,0194490	0,0213080	0,0224870	0,0023180	0,0023540	0,0236890	0,0237230	1875
	0,0205860	0,0266480	0,0307410	0,0333970	0,0350330	0,0035970	0,0364460	0,0366420	0,0366990	1500
	0,0443570	0,0509820	0,0535320	0,0542130	0,0542830	0,0544860	0,0055150	0,0563520	0,0580940	750
	0,0065660	0,0706180	0,0712910	0,0071460	0,0725070	0,0747020	0,0077970	0,0821640	0,0871250	500
	0,0867650	0,0088030	0,0880050	0,0900210	0,0941520	0,1000700	0,1072900	0,1154900	0,1244400	375
	0,0120900	0,1219800	0,1235700	0,1278800	0,1344800	0,1427300	0,1521200	0,1623500	0,1732600	300
	0,1533800	0,1532600	0,1566800	0,1635900	0,1727200	0,1832800	0,1948300	0,2071600	0,2201500	250
	0,1730400	0,1757700	0,1816200	0,1906700	0,2021300	0,2150100	0,2289100	0,2436800	0,2591500	214
	0,1539100	0,1628600	0,1851400	0,2138900	0,2461400	0,2794400	0,3155400	0,3533800	0,3930100	188
	0,1643200	0,1892300	0,2310200	0,2791500	0,3330900	0,3883600	0,4477100	0,5087300	0,5744500	167
	0,2834200	0,3062800	0,3428300	0,3853100	0,4318600	0,4776100	0,5293200	0,5846500	0,6420100	150

De los resultados de la tabla 5.5, se pueden extraer las siguientes observaciones:

1. Para un número de datos iguales o superiores a 750, el menor valor del FPE recae sobre modelos de orden superior a 6; específicamente para la prueba, este calor corresponde a los modelos de orden 7 y 8.
2. Para un número de datos que oscila en un rango entre 250 y 500, el menor valor del FPE oscila entre modelos de orden 2 y 3.
3. Para un número de observaciones iguales o menores a 214, el mínimo valor del FPE recae en los modelos de orden 2.

De acuerdo con estas observaciones se nota una tendencia a que para un gran número de datos de entrada (superiores a 750), los menores valores del FPE recaen en modelos de orden grande (de 7 en adelante); y que para un número bajo de datos de entrada (inferiores a 500), los menores valores del FPE recaen sobre modelos de orden pequeño (entre 2 y 3).

5.2.2 Error final de predicción (FPE) para el algoritmo de mínimos cuadrados ponderados. En la tabla 5.6 se presentan los resultados del error de predicción final en la ejecución del algoritmo de Mínimos Cuadrados ponderados. Para este caso al igual que en los anteriores, se obtuvieron modelos desde orden dos hasta orden 10, variando el número de datos muestreados desde 150 hasta 1875.

Tabla 5.6 Orden del sistema, número de datos y FPE (Mínimos cuadrados ponderados)

ORDEN	2	3	4	5	6	7	8	9	10	DATOS
FPE PARA MINIMOS CUADRADOS PONDERADOS	0,0137610	0,0189380	0,0224270	0,0244960	0,0254000	0,0254170	0,0249670	0,0244750	0,0241060	1875
	0,0223260	0,0302160	0,0352960	0,0381080	0,0391170	0,0389140	0,0382630	0,0376710	0,0037250	1500
	0,0478510	0,0564910	0,0582130	0,0573420	0,0566090	0,0566710	0,0575830	0,0592220	0,0614940	750
	0,0695760	0,0753110	0,0745440	0,0740850	0,0752850	0,0779550	0,0818570	0,0867990	0,0926120	500
	0,0910420	0,0917330	0,0909570	0,0930950	0,0977490	0,1044700	0,1127600	0,1222400	0,1326500	375
	0,1250500	0,1256700	0,1269400	0,1315600	0,1388700	0,1481500	0,1588400	0,1705700	0,1831400	300
	0,1575400	0,1564800	0,1597600	0,1670800	0,1770400	0,1887500	0,2017100	0,2156500	0,2304500	250
	0,1756600	0,1781700	0,1841800	0,1937800	0,0206200	0,0220300	0,2356400	0,2520400	0,2693300	214
	0,1542600	0,1629700	0,0185900	0,2170400	0,2535900	0,2922600	0,3347500	0,3796200	0,4268600	188
	0,1644400	0,0189300	0,2322900	0,2867700	0,3506900	0,4189100	0,4940900	0,0568900	0,6507400	167
	0,2834200	0,3062800	0,3434800	0,3894200	0,4419800	0,4956800	0,5566900	0,6233700	0,6931500	150

De los resultados de la tabla 5.6, se pueden extraer las siguientes observaciones:

1. En cinco de los once casos (para 1875, 750, 500, 300 y 150 datos) el menor valor del FPE recae sobre modelos de orden 2.
2. En dos de los once casos (para 250 y 167 datos) el menor valor del FPE recae sobre modelos de orden 3.
3. En dos de los once casos (para 375 y 268 datos) el menor valor del FPE recae sobre modelos de orden 4.
4. En un solo caso (para 1500 datos) el menor valor del FPE recae sobre un modelo de orden 10.
5. En un solo caso (para 214 datos) el menor valor del FPE recae sobre un modelo de orden 6.

De acuerdo con estas observaciones se nota una tendencia a que para un gran número de datos de entrada (iguales o superiores a 500), con la excepción del

segundo caso que corresponde a 1500 datos, los menores valores del FPE recaen en modelos de orden 2; y que para un número bajo de datos de entrada (inferiores a 188), los menores valores del FPE recaen sobre modelos de orden pequeño (entre 2 y 3).

En este capítulo se presentó un análisis de los resultados de la aplicación de los algoritmos de mínimos cuadrados y de mínimos cuadrados ponderados. Se evidencia a partir de los resultados que los modelos de predicción obtenidos con la aplicación del método de mínimos cuadrados tienen un valor del FPE ligeramente menor a los obtenidos con la aplicación del método de mínimos cuadrados ponderados.

CONCLUSIONES

Este trabajo constituye un estudio inicial en el análisis de aspectos computacionales, como la complejidad computacional asociado a algoritmos de identificación paramétrica aplicados al caso específico de reactores químicos de polimerización, tipo CSTR.

Este estudio se centra en los métodos mayormente utilizados en la literatura de modelamiento paramétrico de un CSTR. Estos métodos son los de Mínimos Cuadrados y Mínimos Cuadrados Ponderados aplicados a un modelo ARX, que es una estructura de modelos paramétricos lineales.

Los experimentos desarrollados en este trabajo evidencian la necesidad de privilegiar un bajo ancho de banda para la señal de entrada del sistema con el objetivo de lograr una mejor respuesta del modelo con respecto al comportamiento real del reactor en estudio.

Aunque el modelamiento paramétrico tiene asociadas herramientas bien conocidas desde el punto de vista de análisis de sistemas dinámicos, nos hemos enfocado en al análisis computacional, partiendo de una validación previa de cada modelo obtenido. Tanto el algoritmo de mínimos cuadrados como el algoritmo de mínimos cuadrados ponderados estudiado presentan una complejidad computacional de orden 3 $O(n^3)$.

En términos generales y de acuerdo con los criterios de comparación presentados en este trabajo, el algoritmo de Mínimos Cuadrados presenta un menor tiempo de ejecución comparado con los tiempos de ejecución del algoritmo de Mínimos Cuadrados Ponderados. Adicionalmente, los FPE obtenidos en general fueron bastante similares para ambos casos, siendo ligeramente menores para el caso del método de Mínimos Cuadrados.

Por lo general, en el algoritmo de mínimos cuadrados los modelos de orden menor (de 2 a 4) son los que tienen los valores más pequeños del FPE, en tanto que con el algoritmo de los mínimos cuadrados ponderados esta tendencia se desplaza hacia los modelos de orden intermedio (de 2 a 6) en los experimentos desarrollados.

Los dos algoritmos implementados en esta tesis permiten obtener modelos de predicción que siguen el comportamiento del proceso original y que además evidencian una buena predicción aún en presencia de perturbaciones. Las diferencias en amplitud de las señales se ajustaron finalmente con la adecuación de la ganancia de la señal de salida.

Este trabajo permitió evidenciar que la salida del modelo de predicción ante una entrada diferente a la señal de prueba original, sigue un comportamiento muy similar al del proceso original; esta afirmación quedó demostrada con las

pruebas de un modelo de predicción obtenido con la aplicación del algoritmo de mínimos cuadrados.

Siendo este un primer trabajo en análisis computacional, el siguiente paso consiste en probar en modelos lineales OE y ARMAX, al igual que en modelos no lineales paramétricos, los cuales son potencialmente útiles en aplicaciones de simulación y control.

Los modelos de predicción del proceso de polimerización en un CSTR desarrollados en este trabajo, tienen ciertas limitaciones dado que son modelos lineales de un proceso altamente no lineal. De ahí que la señal de prueba deberá ser positiva ya que en caso contrario la respuesta de predicción se alejará de la respuesta original.

REFERENCIAS BIBLIOGRAFICAS

[1] Ogunnaike, B. and Harmon R., W., Process Dynamics, Modeling and Control. Published by Oxford University Press Inc., páginas 363-364, 1994.

[2] Bequette, B.W., Process Dynamics, modeling, analysis and simulation. Prentice Hall PTR. 1998.

[3] Van den Bosch, P.P.J. and Van der Klauw, A.C., Modeling, Identification and Simulation of Dynamical Systems. CRC Press, Inc. Florida. 1994.

[4] Choi, K.Y., Análisis of Steady State of free Radical Solution Polymerization in a Continuous Stirred Tank Reactor. Department of Chemical and Nuclear Engineering, University of Maryland, pp 975-981.

[5] Soroush, M. and Zambare, N., "Nonlinear Output Feedback Control of a Class of Polymerization Reactors", *IEEE Transactions on Control Systems Technology, vol. 8, no. 2, march 2000, pp.310-320.*

[6] Zambare, N., Soroush, M. and Ogunnaike, B.A., "Multi-Rate Control of a polymerization Reactor: a Comparative Study", *Proceedings of the American Control, San Diego, California, June 1999, pp. 2553-2557.*

[7] Wisner, D.A., "Conversion Control in a Continuous Reactor Train: Synthesis and Computer Simulation", *IEEE Transactions on Automatic Control, Vol AC-10, No. 4, October, 1965, pp. 455-460.*

[8] Svolos, A.E. and Todd-Pokropek, A. , “Time and space results of dynamic texture feature extraction in MR and CT image analysis”, Information Technology in Biomedicine, IEEE Transactions on, June, Volume: 2 , Issue: 2, 1998, 48 – 54.

[9] Chong Jin Ong and Gilbert, E.G., “Fast versions of the Gilbert-Johnson-Keerthi distance algorithm: additional results and comparisons”, Robotics and Automation, IEEE Transactions on, August, Volume: 17 , Issue: 4, 2001, 531 – 539.

[10] Floreen, P., “The convergence of Hamming memory networks”, Neural Networks, IEEE Transactions on, July 1991, Volume: 2, Issue: 4, 1991, 449 – 457.

[11] Chikkula, Y. and Jay H. Lee, “Input Sequence Design for Parametric Identification of Nonlinear Systems”, Proceedings of the American Control Conference Albuquerque, New Mexico June 1997, pp. 3037-3041.

[12] Kahler, G.R., Vajda, F. and Della Torre, E. “Parameter Estimation Techniques for Recording Media”, IEEE Transactions on Magnetics, vol 32, no. 5, September 1996, pp. 4240-4242.

[13] Garrido, S., “Identificación, Estimación y Control de Sistemas No-lineales mediante RGO”. PhD thesis, Universidad Carlos III de Madrid, España, 1999.

[14] Leonaritis, I. J. and Billings, S. A., "Input-output parametric models for non-linear systems", International Journal of Control. 41, 1985, pp 303-344..

[[15]] Kristinson, K. and Dumont, G., System identification and control using genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(5):1033–1046, 1992.

[16] Flockton, S. J. and White, M. J., Pole-zero system identification using genetic algorithms. In *Proceedings 5th International Conference on Genetic Algorithms*, pages 531–535. University of Illinois at Urbana Champaign, USA, 17-21 July 1993.

[17] Tan, K. C., Li, Y., Murray-Smith, D. J. and Sharman, K. C., System identification and linearization using genetic algorithms with simulated annealing. In *Proc. 1st IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*, Sheffield, UK., 1995.

[18] LJUNG Lenart, System Identification – Theory For the User, Second Edition, PTR Prentice Hall, Upper Saddle River,N.J., 1999.

[19] Gupta, M.M. and Yamakawa, T., Fuzzy Computing, Theory, Hardware and applications, First edition, Elsevier Science Publishers B.V., 1988.

[20] Van Oversche, P. and De Moor, B., Subespace Identification for Linear Systems, Theory-implementation.applications. First Edition, Kluwer Academiv Publishers.1997.

[21] Willems J. *From time series to linear systems.* Automatica, Part I: Vol. 22, no. 5, pp. 561580, 1986, Part II: Vol. 22, no. 6, pp. 675694, 1986, Part III: Vol. 23, no. 1, pp. 87115, 1987.

[22] Kung S.Y. *A new identification method and model reduction algorithm viasingular value decomposition.* 12th Asilomar Conf. on Circuits, Systems and Comp., pp. 705714, Asilomar, CA, 1978.

[23] Moonen M., De Moor B., Ramos J., Tan S. *A subspace identification algorithm for descriptor systems.* Systems & Control Letters, Vol. 19, pp. 4752, 1992.

[24] MacGregor, J. F., Penlidis, A. and Hamielec, A. E., Control of polymerization reactors: A review. *Polymer Process Engineering*, 2(2 & 3):179–206, 1984.

[25] Kiparissides, C., Polymerization reactor modeling: A review of recent developments and future directions. *Chemical Engineering Science*, 51(10):1637–1659, 1996.

[26] Embirucu, M., Lima, E.L. and Pint, J.C., A survey of advanced control of polymerization reactors. *Polymer Engineering & Science*, 36(4):433–447, 1996.

[27] Maner, B.R and Doyle III, F.J, Babatunde A. Ogunnaike and Ronald K. Pearson, Nonlinear Model Predictive Control of a Simulated Multivariable Polymerization Reactor Using Second-order Volterra Models. School of Chemical Engineering, Purdue University, West Lafayette, IN 47907-1283.

[28] Bard,Y., *Nonlinear Parameter Estimation*. Academic Press, New York, 1974.

[29] Stewart, W. E., Caracotsios, M. and Sørensen, J. P., Parameter estimation from multiresponse data. *AIChE Journal*, 38(5):641–650, 1992.

[30] Oliveira, D. y Moreira, E., A Chemical Reactor Benchmark for Parallel Adaptive Control Using Feedforward Neural Networks, IEEE, 2000.

[31] Sawattanakit, N. and Jaovisidha, V., Process Fault Detection and Diagnosis in CSTR System Using On-line Approximator, Department of Electrical Engineering, Chulalongkorn University Phaya Thai Road, Bangkok, IEEE , 1998.

[32] Anderson, H. L., Kemmler, A. and Strey, R., The basic processes of polymerization and its real kinetic analysis. *Journal of Thermal Analysis*, 47(4):1041–1062, 1996.

[33] Sirohi, A. and Yong Choi, K., On-line parameter estimation in a continuous polymerization process. *Industrial and Engineering Chemistry Research*, 35(4):1332–1343, 1996.

[34] Ogunnaike, B., On-line modeling and predictive control of an industrial terpolymerization reactor. *International Journal of Control*, 59(3):711–729, 1994.

[35] Soroush, M. and Kravaris, C., Nonlinear control of a batch polymerization reactor: an experimental study. *AIChE Journal*, 38(9):1429–1448, 1992.

[36] Lewin, D.R., Modeling and control of an industrial pvc suspension polymerization reactor. *Computers & Chemical Engineering*, 20(Suppl.):S865–S870, 1996.

[37] Crowley, T. and Choi, K., On-line monitoring and control of a batch polymerization reactor. *Journal of Process Control*, 6(2/3):119–127, 1996.

[38] Jyh-Shyong Chang and Po-Hsun Liao. Molecular weight control of a batch polymerization reactor: Experimental study. *Industrial and Engineering Chemistry Research*, 38(1):144–153, 1999.

[39] Arora, N. y Biegler, L., Parameter Estimation for a Polymerization Reactor Model with a Composite-Step Trust-Region NLP Algorithm, *Ind. Eng. Chem. Res.,* 43 (14), 3616 -3631, 2004

[40] Ljung, L., System Identification Toolbox For use with Matlab, User's Guide, The mathworks, Inc. (1997).

[41] De la Cruz, J.M., Aranda, J., Ruipérez, P. y Diaz, J.M. "Identificación de sistemas multivariables acoplados con restricciones". Pp.:239-246. III Congreso de usuarios de Matlab – 17-19 de Noviembre de 1999.

[42] Ray, W., New approaches to the dynamic of nonlinear systems with implications for process and control system design. Chemical Process Control 2, edited by D. E. Seborg & T. F. Edgar, United Engineering Trustees, New York, 246-267, 1982.

[43] Zambare, N., Soroush, M. and Grady, M.C., "Multi-Rate Nonlinear State Estimation in a Polymerization Reactor: a Real Time Study", *Proceedings of the American Contro Conferences, Anchorage, Mayo 2002, pp. 2553-2557.*

[44] Chitanov, V., Kiparissides, M. and Petrov, M., "Neural-Fuzzy Modelling of polymer Quality in Batch Polymerization Reactors", Second IEEE International Conference Intelligent Systems, June 2004, pp. 67-72.

[45] Tyagunov, A.A, “High-performance Model Predictive Control for Process Industry”, Doctoral Thesis, Technische universiteit Eindhoven, june 2004.

[46] Cormen, T.H., Leiserson, C.E., Rivest, R.L. y Stein, C.,Introduction to Algorithms. Second Edition, 2001. Mc Graw Hill.

[47] Aho, A.V., Hopcroft, J. E. y Ullman, J.D., Estructura de datos y algoritmos, Primera edición, 1998. Pearson Addison Wesley.

[49] Vilar, J.M., "Modelos Estadísticos Aplicados", Junio, 2003, Publicaciones de la UDC, monografía 101. www.udc.es/publicaciones.

[50] Shampine, L. F. and Reichelt, M. W., The MATLAB ODE Suite, SIAM Journal on Scientific Computing, 18-1, 1997.

[51] Arce, R., Conceptos básicos sobre la heterocedasticidad en el modelo básico de regresión lineal, tratamiento con e-views, Universidad Autónoma de Madrid, Abril de 2001, pp. 2-19.

[52] Levenspiel, O., Ingeniería de las reacciones químicas. Editorial Reverté S.A. 1,981.

[53] European IPPC bureau. Best Available Techniques in the production of polymers. 2005.

[54] Sayer, C. y Guiudici, R., A Comparison of Different Modeling Approaches For The Simulation Of The Transient And Steady-State Behavior of Continuous Emulsion Polymerizations In Pulsed Tubular Reactors. 2001.

[55] Nising Phillip. High-temperature radical polymerization of methyl methacrylate in a continuous pilot scale process. École Polytechnique Fédérale De Lausanne. Tésis número:3460. 2006.

[56] Russo L., Bequette W. Operability of chemical reactors: Multiplicity behaviors of extended CSTR models. Department of chemical engineering of Rensselaer Institute. 1,996.

[57] Alan V. Oppenheim et al. Signals and Systems. Second Edition. Prentice-Hall Int. 1998

ANEXO A. IMPLEMENTACIÓN DEL ALGORITMO DE MINIMOS CUADRADOS PARA LA OBTENCIÓN DE UN MODELO PARAMETRICO ARX

A través de la siguiente secuencia de comandos en Matlab se realiza la identificación y obtención de parámetros del modelo, usando el método de mínimos cuadrados. Este algoritmo permite realizar distintas pruebas basadas en la secuencia anterior de Matlab con el fin de obtener el modelo que se considera responde de la forma mas semejante a la planta original.

```
%********************** ALGORITMO DE MINIMOS CUADRADOS ***************
%
%   Este archivo contiene el codigo del algoritmo de Minimos Cuadrados
para la
%   obtencion de modelos parametricos con estructura ARX. Se aplica a
un
%   proceso de polimerizacion de estireno en un CSTR. El modelo y los
parametros
%   fueron tomados de:
%   "A Chemical Reactor Benchmark for Parallel Adaptive Control Using
%   Feedforward Neural Networks", Oliveira et al.
%   IEEE, 2000
%
%   Elaborado por Oscar Segundo Acuña Camacho / Noviembre de 2005
%   Universidad Autonoma de Bucaramanga
%   Universidad Tecnologica de Bolivar - Cartagena de Indias
%
%
cstr_parameters                      % El archivo cstr_parameters
establece los parametros del CSTR
Ts=0.08;                             % Tiempo de muestreo
sim('ReactorBenchmarkDataId');       % Corre el archivo que genera los
datos del CSTR_benchmark
sim('valida1');                      % Corre los datos para el calculo
de la validacion con entrada escalon
% Se cargan los datos de la simulacion extrayendolos del workspace
tini=clock
u1=simout(:,1);
y1=simout(:,2);
nd=length(simout)                   % Numero total de datos
u=u1(1:(nd-1)/2)                    % Seleccionamos .... datos para
construir el modelo
```

```
y=y1(1:(nd-1)/2)
z2=[y1 u1];                         % Seleccionamos .... datos para
construir el modelo
figure(1)
idplot(z2)                          % Se observan los datos
seleccionados para la construccion del modelo
title('Señal pseudoaleatoria')
u=detrend(u);                       % Remueve los niveles constantes de
u (entrada) y hace el valor medio igual a cero
y=detrend(y);                       % Remueve los niveles constantes
de y (salida) y hace el valor medio igual a cero
Y=y;
tY=size(Y);                         % vector de dimensiones de la
matriz de salidas
tu=size(u);                         % vector de dimensiones de la
matriz de entradas
for n=2:10;                         %orden de la matriz de parametros
    Z=zeros(length(y),2*n+1);       %Inicializacion de la matriz Z
    Zext=zeros(1,2*n+1);            %Inicializacion de matriz auxiliar
    for k=n+1:length(y),            %comienza en n+1 para que no busque
una posicion de indice negativo en la matriz
        Em=Zext(tY(2)+tu(2):length(Zext)-(tY(2)+tu(2)));
        %dependiendo del numero de entradas es el numero de indices
que se corren
        %por eso son tY=columnas del vector de entradas
        %mas tu=columnas del vector de salidas, que dependen del
        %numero de entradas y salidas respectivamente.
    Z(k,:)=[u(k) y(k-1) u(k-1) Em];
    %la matriz se construye asi:
    %k corresponde al numero de la fila,
    %la primera columna siempre corresponde con u(k)
    %las tY*yu columnas corresponden a las salidas correspondientes a
k-1
    %seguidas por las entradas k-1
    %y las ultimas columnas van desde la columna tY+tu hasta la
(2*n+1)-(tY+tu)
    %de la fila anterior, por eso se utiliza la matriz auxiliar
    %para almacenar la fila anterior.
    %tambien podria utilizarce:
    %       Zext=Z(k-1,:) como primera linea del 'for'
    %       con lo cual no es necesario inicializar esta matriz.
    Zext=Z(k,:);
    end
% De acuerdo con la solucion matricial para el algoritmo de minimos
cuadrados
% se tiene que el valor del vector de parametros P es igual a:
% P=inv(Z'*Z)Z'*Y, donde:
% Z:    Matriz de Observacion
% Z':   Inversa de la matriz de observacion
% Y:    Matriz de datos del experimento
    P=inv(Z'*Z)*Z'*Y;
    Y2=(P'*Z')';                %se halla la salida con los parametros
obtenidos

    t=Ts:Ts:length(y)*Ts;
    figure(2)
```

```
    plot(t,Y,'r*')              %se grafica para verificacion
    hold
    plot(t,Y2,'b+')
    legend('Salida experimental','Salida
ajustada'),xlabel('Time(seg)'),ylabel('Temperatura Chaqueta')
    hold off

% Funcion de transferencia
% denp=[P(2) P(4)];
p=length(P);
den=[1 -P(2)];
for i=4:2:p,
    den=[den -P(i)];
end
num=[P(1)];
for i=3:2:p,
    num=[num P(i)];
end
TFz=tf(num,den,Ts);
present(TFz)

% Ahora se procede a la validacion, escogiendo un conjunto de datos
que no se usaron en la
% construccion del modelo. La funcion detrend Remueve los niveles
constantes  hace el valor
% medio igual a cero.
u2=detrend(u1((nd-1)/2:nd-1));              % Se escoge la entrada
aleatoria en un rango diferente al usado en la construccion del modelo
y2=detrend(y1(nd/2:nd-1));                  % Igual con los datos de
salida.

th=poly2th(den,num)                         % poly2th convierte el
polinomio del modelo encontrado en formato theta
ysim=idsim(u2,th);                          % idsim simula el sistema
especificado en el formato theta
figure(3)
plot([y2(1:(nd-1)/2) ysim(1:(nd-1)/2)]);    %Se compara la salida del
modelo y2 y la salida simulada ysim
legend('Salida real','Salida del Modelo'),xlabel('No. de
Muestras'),ylabel('Temperatura Chaqueta')
tfin=clock
tt=tfin-tini;
tt
% Calculo del FPE (Factor de prediccion del error)
% Se define como: FPE = Akaikes Final Predition Error = V*(1+d/N)/(1-
d/N)
% where V is the loss function, d is the number of estimated
parameters
% and N is the number of estimation data.

N=length(y);                                % Numero de datos
estimados
d=2*n+1;                                    % Numero de parametros
estimados
fp=sum((y-Y2).^2)/N;
```

```
fpl(n)=(1+d/N)*fp/(1-d/N);                  % fpl es el Factor de
Prediccion del Error final
n                                           % orden del sistema

%Validacion de la respuesta del modelo ante una entrada escalon
unitario
sysd=tf(num,den,Ts);
sysc = d2c(sysd);
figure(4)
%sim('valida1')
uv=simout1(:,1);
yv=simout1(:,2);
nv=length(simout1);                         % Numero total de datos
uv1=uv(1:(nv-1)/2) ;                        % Seleccionamos .... datos
para la entrada
yv1=yv(1:(nv-1)/2) ;                        % Seleccionamos .... datos
para la salida al escalon de entrada
plot(t,yv1,'y')
hold on
step(sysc,100)
legend('Temperatura real (amarillo) Vs Temperatura del modelo')
end
```

ANEXO B. IMPLEMENTACIÓN DEL ALGORITMO DE MINIMOS CUADRADOS PONDERADOS PARA LA OBTENCIÓN DE UN MODELO PARAMETRICO ARX

Este algoritmo permite realizar distintas pruebas basadas en la secuencia de Matlab con el fin de obtener el modelo que se considera responde de la forma mas semejante a la planta original.

A través de la siguiente secuencia de comandos en Matlab se realiza la identificación y obtención de parámetros del modelo, usando el método de mínimos cuadrados:

```
%***************** ALGORITMO DE MINIMOS CUADRADOS PONDERADOS*********
%
%   Este archivo contiene el codigo del algoritmo de Minimos Cuadrados
Ponderados para la
%   obtencion de modelos parametricos con estructura ARX. Se aplica a
un proceso de
%   polimerizacion de estireno en un CSTR. El modelo y los parametros
fueron tomados de:
%   "A Chemical Reactor Benchmark for Parallel Adaptive Control Using
%   Feedforward Neural Networks", Oliveira et al.
%   IEEE, 2000
%
%   Elaborado por Oscar Segundo Acuña Camacho / Noviembre de 2005
%   Universidad Autonoma de Bucaramanga
%   Universidad Tecnologica de Bolivar - Cartagena de Indias
%

cstr_parameters                      % El archivo cstr_parameters
establece los parametros del CSTR
Ts=0.08;                              % Tiempo de muestreo
sim('ReactorBenchmarkDataId');        % Corre el archivo que genera los
datos del CSTR_benchmark
sim('valida1');                       % Corre los datos para el calculo
de la validacion con entrada escalon
% Se cargan los datos de la simulacion extrayendolos del workspace
tini=clock
u1=simout(:,1);
y1=simout(:,2);
nd=length(simout);                    % Numero total de datos
```

```
u=u1(1:(nd-1)/2);                    % Seleccionamos la mitad de los
datos para construir el modelo
y=y1(1:(nd-1)/2);
z2=[y1 u1];                          % Seleccionamos .... datos para
construir el modelo
figure(1)
idplot(z2)                           % Se observan los datos
seleccionados para la construccion del modelo
title('Señal pseudoaleatoria')
u=detrend(u);                        % Remueve los niveles constantes
de u (entrada) y hace el valor medio igual a cero
y=detrend(y);                        % Remueve los niveles constantes
de y (salida) y hace el valor medio igual a cero
Y=y;
tY=size(Y);                          % vector de dimensiones de la
matriz de salidas
tu=size(u);                          % vector de dimensiones de la
matriz de entradas

for n=2:10;                          % orden de la matriz de parametros
    Z=zeros(length(y),2*n+1);        % Inicializacion de la matriz Z
    Zext=zeros(1,2*n+1);             % Inicializacion de matriz
auxiliar
    for k=n+1:length(y),             % comienza en n+1 para que no
busque una posicion de indice negativo en la matriz
        Em=Zext(tY(2)+tu(2):length(Zext)-(tY(2)+tu(2)));
        %dependiendo del numero de entradas es el numero de indices
que se corren
        %por eso son tY=columnas del vector de entradas
        %mas tu=columnas del vector de salidas, que dependen del
        %numero de entradas y salidas respectivamente.
        Z(k,:)=[u(k) y(k-1) u(k-1) Em];
    %la matriz se construye asi:
    %k corresponde al numero de la fila,
    %la primera columna siempre corresponde con u(k)
    %las tY*yu columnas corresponden a las salidas correspondientes a
k-1
    %seguidas por las entradas k-1
    %y las ultimas columnas van desde la columna tY+tu hasta la
(2*n+1)-(tY+tu)
    %de la fila anterior, por eso se utiliza la matriz auxiliar
    %para alamacenar la fila anterior.
    %tambien podria utilizarce:
    %       Zext=Z(k-1,:) como primera linea del 'for'
    %       con lo cual no es necesario inicializar esta matriz.
        Zext=Z(k,:);
    end
% De acuerdo con la solucion matricial para el algoritmo de minimos
cuadrados
% se tiene que el valor del vector de parametros P es igual a
% P=inv(Z'*Z)Z'*Y, donde:
% Z:    Matriz de Observacion
% Z':   Inversa de la matriz de observacion
% Y:    Matriz de datos del experimento

    P=inv(Z'*Z)*Z'*Y;
```

```
    Y2=(P'*Z')';                                    % se halla la salida
con los parametros obtenidos

% Determinacion de Matriz de ponderacion
    N=length(y);                                    % Numero de datos
estimados
    d=2*n+1;                                        % Numero de
parametros estimados
    A=sum((y(1:N-1)-Y2(1:N-1)).*(y(2:N)-Y2(2:N)));  % Calculo del
numerador del estimado de rho=r
    B=sum((y-Y2).^2);                               % Calculo del
denominador del estimado de rho=r
    r=A/B;                                          % Calculo del
estimado rho=r
    T=ones(N);
    T1=tril(T,1).*triu(T,1)*(-r);
    T2=tril(T,-1).*triu(T,-1)*(-r);
    T3=eye(N)*(1+r^2);
    T3(1)=1;
    T3(N*N)=1;
    W=T1+T2+T3;                                      % inversa del
estimador de los pesos W^-1

% Calculo de la salida por WLS

    Pw=inv(Z'*W*Z)*Z'*W*Y;
    Y2=(Pw'*Z')';                                   % se halla la
salida con los parametros obtenidos
    t=Ts:Ts:N*Ts;
    figure(5)
    plot(t,Y,'r*')                                  %se grafica para
verificacion
    hold
    plot(t,Y2,'b+')
    legend('Salida experimental','Salida
ajustada'),xlabel('Time(seg)'),ylabel('Temperatura Chaqueta')
    hold off

%   Funcion de transferencia
%   denp=[P(2) P(4)];
    p=length(Pw);
    den=[1 -Pw(2)];
    for i=4:2:p,
        den=[den -Pw(i)];
    end
    num=[Pw(1)];
    for i=3:2:p,
        num=[num Pw(i)];
    end
    TFz=tf(num,den,Ts);
    present(TFz)

% Ahora se procede a la validacion, escogiendo un conjunto de datos
que no se usaron en la
% construccion del modelo. La funcion detrend Remueve los niveles
constantes  hace el valor  medio igual a cero.
```

```
    u2=detrend(u1((nd-1)/2:nd-1));              % Se escoge la entrada
aleatoria en un rango diferente al usado en la construccion del modelo
    y2=detrend(y1(nd/2:nd-1));                  % Igual con los datos
de salida.

    th=poly2th(den,num);                         % poly2th convierte
el polinomio del modelo encontrado en formato theta
    ysim=idsim(u2,th);                          % idsim simula el
sistema especificado en el formato theta
    figure(6)
    plot([y2(1:(nd-1)/2) ysim(1:(nd-1)/2)]);    % Se compara la salida
del modelo y2 y la salida simulada ysim
    legend('Salida real','Salida del Modelo'),xlabel('No. de
Muestras'),ylabel('Temperatura Chaqueta')
    tfin=clock;
    ttw=tfin-tini;
    ttw
% Calculo del FPE (Factor de prediccion del error)
% Se define como: FPE = Akaikes Final Predition Error = V*(1+d/N)/(1-
d/N)
% where V is the loss function, d is the number of estimated
parameters
% and N is the number of estimation data.

    fp=sum((y-Y2).^2)/N;
    fpw(n)=(1+d/N)*fp/(1-d/N);                   % fpw es el Factor de
Prediccion del Error final
    n                                           % orden del sistema

%Validacion de la respuesta del modelo ante una entrada escalon
unitario

    sysd=tf(num,den,Ts);
    sysc = d2c(sysd);
    figure(7)
%sim('valida1')
    uv=simout1(:,1);
    yv=simout1(:,2);
    nv=length(simout1);                         % Numero total de
datos
    uv1=uv(1:(nv-1)/2) ;                        % Seleccionamos ....
datos para la entrada
    yv1=yv(1:(nv-1)/2) ;                        % Seleccionamos ....
datos para la salida al escalon de entrada
    plot(t,yv1,'y')
    hold on
    step(sysc,100)
    legend('Temperatura real (amarillo) Vs Temperatura del modelo')
end
```