ESCUELA POLITECNICA NACIONAL

FACULTAD DE INGENIERIA ELECTRICA

IDENTIFICACION PARAMETRICA DISCRETA

TESIS PREVIA A LA OBTENCION DEL TITULO DE INGENIERO EN ELECTRONICA Y CONTROL

BYRON PATRICIO ZARATE MORA

MARZO, 1996

Certifico que el presente trabajo ha sido elaborado en su totalidad por el Sr. Byron Patricio Zárate Mora.

Ing. Patricio Burbano

Director

Agradecimiento

Un especial y sincero agradecimiento al Ing. Patricio Burbano por su valiosa colaboración en la elaboración de este trabajo.

Dedicatoria

A mis padres y a mi hermano por el apoyo brindado en todos los momentos de mi vida.

CONTENIDO

CAPÍTULO I : INTRODUCCIÓN

1.1 INTRODUCCIÓN

El objetivo de esta tesis es el de implementar un programa computacional para identificación paramétrica discreta utilizando modelos a ecuación de diferencias y mediante el método de mínimos cuadrados recursivos y sus diferentes variantes.

Se realiza un estudio de las características de los mínimos cuadrados ordinarios (MCO), como punto de partida para el desarrollo de los mínimos cuadrados recursivos (MCR), para concluir con las características de los mínimos cuadrados estocásticos (MCS); y, con el algoritmo de máximo de verosimilitud (máximo de Likelihood MLH).

Dentro de los algoritmos a implementar se considera el de mínimos cuadrados recursivos con factor de olvido, mínimos cuadrados extendido (MCE) y máximo de verosimilitud como un método estadístico. Se considera la presencia de ruido, así como también el trabajo a nivel de simulación y en tiempo real.

Se utiliza el lenguaje de programación C para simulación, y se manejan las rutinas pertinentes junto con la adquisición y salida de datos para su aplicación en tiempo real.

A nivel de simulación se tiene la posibilidad de manejar modelos a ecuación de diferencias, diferentes tipos de ruidos y diferentes algoritmos para identificación. En tiempo real se utiliza la tarjeta de adquisición de datos DAS-128 existente en el Laboratorio de Control, diferentes algoritmos desarrollados previamente en simulaciones y se

aplica en dispositivos circuitales o prototipos a pequeña escala.

La importancia de este trabajo radica en el hecho de que en la actualidad la tendencia del control es hacia un control computarizado en donde las acciones de identificación y control son realizadas en línea, es decir el computador forma parte del lazo de identificación y control.

Adicionalmente las técnicas de identificación se utilizan hoy en día debido a la complejidad de los sistemas y a la presencia de ruido por lo que la aplicación de técnicas clásicas de modelación resultan limitadas.

Cabe señalar que se desarrollará un paquete computacional modular y didáctico utilizando ayudas gráficas para visualizar la convergencia de parámetros; y, que permita ilustrar las características y aplicación de los diferentes métodos.

Si bien es cierto que en trabajos anteriores se han desarrollado diferentes temáticas, en este trabajo de tesis se pretende unificar y completar dichos trabajos, en el ambiente de un solo paquete computacional a nivel de simulación y con la opción de poder trabajar en tiempo real.

A manera de introducción, brevemente, se presentan los conceptos de sistema y modelo; y, de modelación, identificación y simulación. Un sistema esta formado por un conjunto de componentes que interaccionan entre sí y tal vez con el medio ambiente; y que cumplen con una tarea específica [4]. Para analizar el comportamiento dinámico de un sistema así como para efectuar un control sobre el mismo,

no se trabaja en forma directa sobre este, sino que se utiliza un modelo del mismo.

El modelo es una representación de las características más importantes del sistema. Puede ser físico (a pequeña escala), abstracto (matemático o numérico). El modelo matemático puede ser a variables de estado, función de transferencia y ecuaciones de diferencia si es discreto.

En la presente tesis se utiliza modelo a ecuaciones de diferencia pues se esta tratando con sistemas discretos.

Existen dos métodos de obtener el modelo matemático o analítico de un sistema así: estructuralista y globalistas [25].

El método estructuralista parte del conocimiento de la estructura física y los componentes del sistema, se conoce como modelación, aquí se parte del conocimiento de las leyes físicas, químicas, y de otras ciencias así como de principios de ingeniería; y, se los aplica a los componentes que conforman un sistema a través de las leyes de Kirchoff, de Newton, de conservación de masa y energía (balance de materia y energía) para obtener un modelo dinámico del mismo.

Los métodos globalistas no requieren del conocimiento de la estructura ni de los componentes de los sistemas. Considera al sistema como un bloque (caja negra) al cual se le aplica una entrada y se obtiene una salida o respuesta. Entonces el modelo se obtiene a partir de mediciones de entrada y salida (que reflejan la dinámica del sistema) y se procesa dichas mediciones mediante un paquete computacional; y,

considerando ciertas condiciones preestablecidas. A este proceso se conoce con el nombre de identificación.

Luego de que un sistema ha sido modelado es necesario realizar la validación del modelo; esto es, averiguar si la descripción del sistema es adecuada para lo cual se simula el funcionamiento de dicho sistema, y se compara con los respuestas del funcionamiento real del mismo. En general la simulacion se realiza en un computador digital debido a la facilidad que este presta, así como la versatilidad para simular condiciones que de otro modo serían difíciles de conseguir.

La simulación se puede definir como un equivalente computacional del modelo. Puede ser mediante computador análogo o digital [17].

Cuando el sistema es complejo es muy difícil manejar las ecuaciones que describen la dinámica de los componentes, inclusive puede haber componentes cuyas relaciones matemáticas no se conozcan o sean no lineales por ejemplo, lo cual dificulta aún más la obtención del modelo analítico.

La identificación de sistemas es más adecuada cuando el sistema es complejo y existe presencia significativa de ruido.

El método de identificación paramétrica que se utiliza en esta tesis se puede realizar fuera de línea (simulación) o sea generando la información independientemente del sistema ó en línea; esto es, simultáneamente con la operación del sistema (tiempo real).

A continuación se presenta un breve resumen de los tópicos más importantes de cada capítulo.

En el Capítulo I se presentan los conceptos básicos de identificación paramétrica discreta, el modelo general que se va a utilizar (modelo ARMA Auto Regressive Moving Average), así como el algoritmo inicial de mínimos cuadrados del cual se va a partir para implementar la identificación.

En el Capítulo II se estudian los distintos algoritmos de identificación: el método de mínimos cuadrados recursivos, mínimos cuadrados generalizados, que son métodos determinísticos, el de mínimos cuadrados estocásticos y el método de máximo de verosimilitud que son métodos estadísticos.

En el Capítulo III se dedica a la implementación de los algoritmos estudiados en un programa computacional que pueda ser utilizado en un computador personal, bajo el ambiente Windows versión 3.1.

El lenguaje escogido para elaborar el programa es el "C", debido a su facilidad para comunicarse a bajo y alto nivel, lo que lo hace muy útil en este trabajo en donde se necesita rutinas de adquisición y salida de datos, y permite crear un ambiente agradable de trabajo para el usuario [18].

El Capítulo IV se presenta los resultados de la simulacion, así como también se presenta los resultados de la aplicación del programa a prototipos en tiempo real.

Se completa el desarrollo de la tesis con los apéndices y las referencias que constan en la bibliografía.

1.2 IDENTIFICACIÓN PARAMÉTRICA DISCRETA

La identificación de sistemas puede ser paramétrica y no paramétrica. Cuando se obtiene un modelo a variables de estado, función de transferencia o ecuaciones de diferencias se dice que se utiliza un método de identificación paramétrica dada cierta estructura preestablecida del modelo, por ejemplo a ecuaciones de diferencia y su orden, se trata de calcular los coeficientes de dicha ecuación o sea los parámetros del modelo.

Cuando se utilizan las curvas de respuesta de frecuencia, o respuesta en el tiempo, esto es información con un número no limitado de datos se tiene identificación no paramétrica.

En el presente trabajo de tesis se restringe el estudio a la identificación paramétrica discreta pues el objetivo es tener modelos para control discreto.

Al analizar el problema de estimación se tiene dos casos:

- Estimación de estado cuando se conoce los parámetros del modelo.

- Estimación de parámetros del modelo (identificación) cuando se conoce el estado del sistema.

El problema de identificación paramétrica discreta en esta tesis esta basado en obtener los parámetros del modelo conociendo los valores de entrada y salida del sistema (conociendo los estados).

1.2.1 Modelo ARMA [13]

El modelo ARMA (Auto Regressive Moving Average) es una representación canónica en la cual se tiene el mínimo número de parámetros necesarios para describir totalmente a un sistema.

El modelo ARMA se puede expresar como:

$$y(k) + a_1 y(k-1) + \ldots\ldots + a_n y(k-n) = b_r u(k-r) + \ldots\ldots + b_m u(k-m) \quad (1.1)$$

donde r es el retardo entre la salida y la entrada.

La ecuación (1.1) en forma simplificada esta dada por

$$y(k) = x(k)\Theta(k) \quad (1.2)$$

en donde:

$$x(k)=\left[y(k-1) \quad y(k-2) \quad \cdots \quad y(k-n) \quad u(k-r) \quad u(k-r-1) \quad \cdots \quad u(k-m)\right] \quad (1.3)$$

$$\Theta(k) = \begin{bmatrix} a_1 \\ \vdots \\ a_n \\ b_r \\ \vdots \\ b_m \end{bmatrix} = [a_1 \quad \cdots \quad a_n \quad b_r \quad \cdots \quad b_m]^T \quad (1.4)$$

El vector fila $x(k)$ contiene la información de los valores anteriores de salida $y(k)$, y de entrada $u(k)$ que constituyen

los estados del sistema, por lo tanto el problema radica en base a estas mediciones identificar o estimar los parámetros del vector $\Theta(k)$.

Los valores estimados de $\Theta(k)$ se les denota como $\hat{\Theta}(k)$, mientras que a los verdaderos valores de los parámetros del sistema se les conoce como $\Theta^{\circ}(k)$.

En la figura 1 se muestra un esquema en bloques del modelo ARMA.

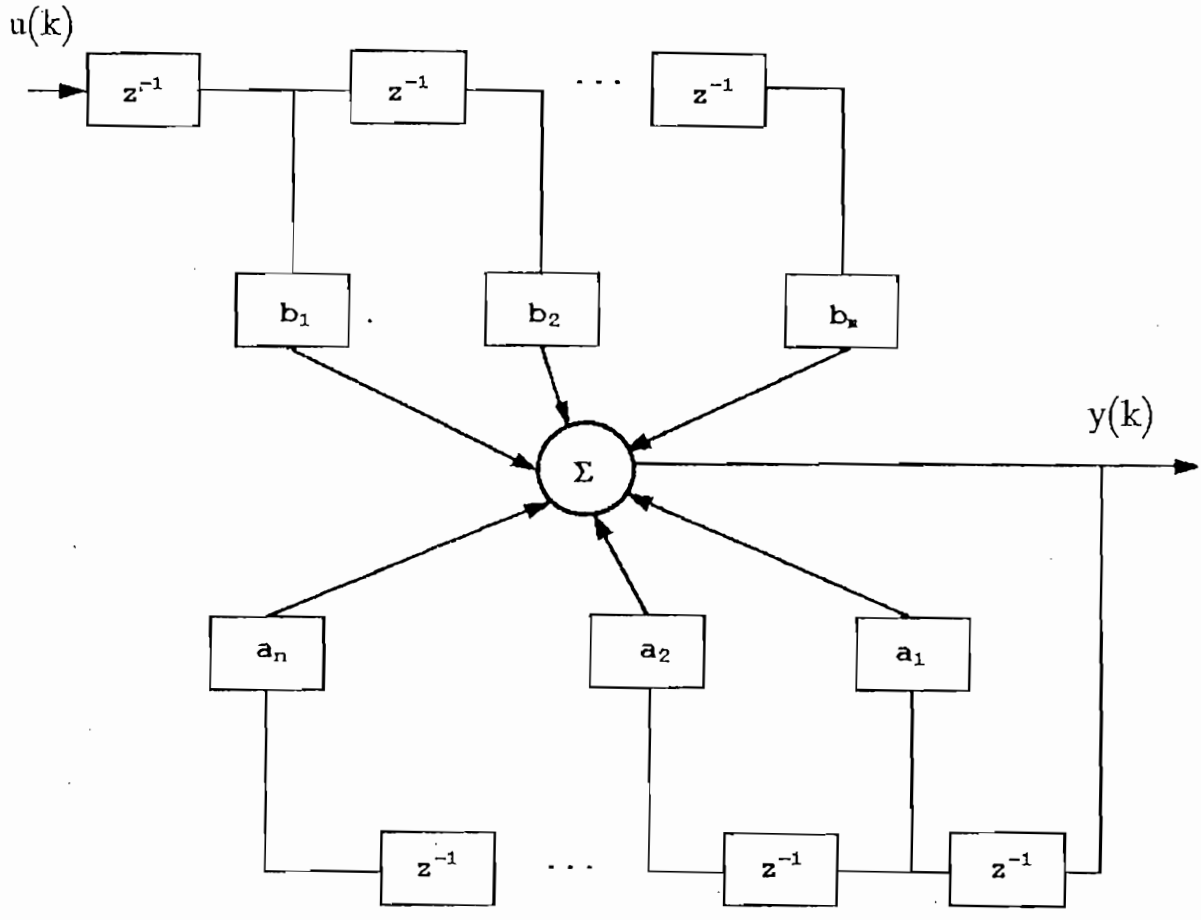

Fig. 1 Modelo ARMA

El nombre de modelo ARMA viene de la combinación de un modelo Auto Regressive (Autoregresivo AR) y un modelo Moving Average (Media Móvil MA).

Un modelo AR, esta definido por la ecuación:

$$y(k) + a_1 y(k-1) + \ldots + a_n y(k-n) = b_0 u(k) \qquad (1.5)$$

y se llama autoregresivo porque la ecuación (1.5) usa los valores anteriores de la salida, para encontrar un nuevo valor de $y(k)$ al instante k.

Un modelo MA, esta definido por la siguiente ecuación:

$$y(k) = b_0 u(k) + b_1 u(k-1) + \ldots + b_m u(k-m) \qquad (1.6)$$

se llama de media móvil porque la media cambia según se tenga más coeficientes b_i en la ecuación (1.6).

Por lo tanto si se combina el modelo AR (Auto Regressive) con el modelo MA (Moving Average) se tiene el modelo ARMA anteriormente definido.

1.3 MÍNIMOS CUADRADOS ORDINARIOS [13]

El método de Mínimos Cuadrados Ordinarios (MCO) es un algoritmo para procesamiento en lote; esto es, no es recursivo, sin embargo es el algoritmo de partida para el estudio de la identificación de sistemas mediante un modelo ARMA a ecuación de diferencias.

1.3.1 Error de la Ecuación

Al analizar la ecuación (1.2) que describe a un sistema, se observa que para tener una adecuada identificación los parámetros estimados $\hat{\Theta}(k)$, deben tender a los verdaderos valores de los parámetros $\Theta^{\circ}(k)$, es decir se busca que

$$\left[\hat{\Theta}(k) - \Theta^{\circ}(k)\right] \rightarrow 0$$

Sin embargo al no conocer el valor de $\Theta^{\circ}(k)$ no es posible conocer el error en los parámetros. La ecuación (1.2) se cumple para los parámetros verdaderos $\Theta^{\circ}(k)$, pero para los valores estimados $\Theta(k)$ tiene un error que depende tanto de la medición $y(k)$ como de los parámetros $\Theta(k)$, por lo cual, en este caso se tiene un error de la ecuación (1.2) y puede escribirse como:

$$y(k) = x(k)\Theta(k) + e(k,\Theta) \qquad (1.7)$$

La ecuación (1.7) es la base para el desarrollo del algoritmo de identificación basado en el modelo ARMA.

Al empezar el proceso de identificación se toman k mediciones de entrada y salida, como $y(k)$ depende de n valores anteriores se parte desde el enésimo término de las mediciones.

Entonces:

$$
\begin{aligned}
y(n) &= x(n)\Theta(n) + e(n) \\
y(n+1) &= x(n+1)\Theta(n+1) + e(n+1) \\
&\vdots \\
y(k) &= x(k)\Theta(k) + e(k)
\end{aligned}
\tag{1.8}
$$

Para tener el sistema de ecuaciones (1.8) de una forma más compacta se define:

El vector de mediciones $Y(k)$ como:

$$
Y(k) = \begin{bmatrix} y(n) \\ y(n+1) \\ \vdots \\ y(k) \end{bmatrix} \tag{1.9}
$$

El vector de errores $E(k)$ como:

$$
E(k) = \begin{bmatrix} e(n) \\ e(n+1) \\ \vdots \\ e(k) \end{bmatrix} \tag{1.10}
$$

Y la matriz de información $X(k)$ como:

$$
X(k) = \begin{bmatrix} x(n) \\ x(n+1) \\ \vdots \\ x(k) \end{bmatrix} \tag{1.11}
$$

Entonces se puede expresar en forma compacta el sistema de ecuaciones (1.8) como:

$$Y(k) = X(k)\Theta(k) + E(k) \qquad (1.12)$$

De la ecuación (1.12) se tiene que el error de la ecuación esta dado por:

$$E(k) = Y(k) - X(k)\Theta(k) \qquad (1.13)$$

1.3.2 Función de Costo

El objetivo es identificar los parámetros $\Theta(k)$ con el mínimo error. Para ello se define un índice de funcionamiento o de minimización cuadrático:

$$J = \sum_{k=n}^{k} e^2(k) = E(k)^T E(k) \qquad (1.14)$$

Se llama índice de funcionamiento mínimo cuadrático debido a que se minimiza el cuadrado del error.

Se calcula el índice de funcionamiento J en base a las ecuaciones (1.13) y (1.14) como:

$$J(\Theta) = \left[Y(k) - X(k)\Theta(k)\right]^T \left[Y(k) - X(k)\Theta(k)\right]$$

multiplicando y suprimiendo la notación dependiente de k se tiene:

$$J(\Theta) = Y^T Y - \Theta^T X^T Y - Y^T X\Theta + \Theta^T X^T X\Theta \qquad (1.15)$$

La minimización se realiza igualando a cero la derivada de J respecto a $\Theta(k)$ y resolviendo la ecuación correspondiente:

$$\frac{dJ}{d\Theta} = 0$$

$$\frac{dJ}{d\Theta} = \frac{d}{d\Theta}\left[Y^T Y - \Theta^T X^T Y - Y^T X\Theta + \Theta^T X^T X\Theta\right] \qquad (1.16)$$

1.3.3 Algoritmo de Mínimos Cuadrados Ordinarios (MCO)

Luego de haber definido el error de la ecuación que es la base para establecer el índice de funcionamiento se procede a hallar el valor de los parámetros estimados, mediante la minimización de la función de costo J. Se tiene entonces:

$$\frac{dJ}{d\Theta} = -X^T Y - X^T Y + X^T X\Theta + X^T X\Theta = 0$$

agrupando términos semejantes, simplificando y asignando $\Theta = \hat{\Theta}$ se tiene:

$$X^T Y - X^T X\hat{\Theta} = 0$$

Entonces al despejar $\hat{\Theta}$ se llega a obtener la expresión que permite encontrar el valor de los parámetros estimados:

$$\hat{\Theta}(k) = \left(X^T(k)X(k)\right)^{-1} X^T(k)Y(k) \qquad (1.17)$$

La ecuación (1.17) permite hallar el valor de los parámetros de la ecuación (1.2) al utilizar el método de los Mínimos Cuadrados Ordinarios.

CAPITULO II: MÉTODOS DE IDENTIFICACIÓN PARAMÉTRICA DISCRETA

2.1 MÍNIMOS CUADRADOS RECURSIVOS (MCR)

2.1.1 ALGORITMO RECURSIVO

2.1.2 CARACTERÍSTICAS Y CONSIDERACIONES COMPUTACIONALES DEL MÉTODO DE MCR

2.1.2.1 VENTAJAS DEL MCR

2.1.2.2 IDENTIFICABILIDAD

2.1.2.3 EXCITACIÓN PERSISTENTE

2.1.2.4 CONDICIONES INICIALES

2.1.2.5 FACTOR DE OLVIDO

2.1.2.6 ALGORITMO FINAL

2.2 MÍNIMOS CUADRADOS ESTOCÁSTICOS (MCS)

2.2.1 ASPECTOS ESTOCÁSTICOS DEL MÉTODO DE MCR

2.2.1.1 RUIDO BLANCO

2.2.1.2 RUIDO CORRELACIONADO

2.2.1.3 MODELO ARMAX

En el capítulo anterior se desarrolló el algoritmo de mínimos cuadrados ordinarios. Este algoritmo si bien es sencillo adolece de las siguientes desventajas.

- Para encontrar los parámetros estimados es necesario invertir la matriz $(X^T X)$.

- Los parámetros se calculan de un lote de datos, lo que hace que no se pueda ver variaciones de los parámetros al ir agregando mediciones.

- La cantidad de memoria utilizada depende del número de mediciones.

- No es implementable en forma recursiva en línea.

- En general el método de MCO es un algoritmo básico de identificación de parámetros, por lo tanto no permite mayores refinamientos.

Por tales motivos se busca realizar variantes a este algoritmo básico. Las variantes más importantes por su implementación y aplicación son:

- MCR (Mínimos Cuadrados Recursivo): Este método permite un procesamiento recursivo, es sencillo y su aplicación es adecuada en tiempo real y para la mayoría de casos prácticos.

- MCS (Mínimos Cuadrados Estocásticos): En este método se considera la presencia de ruido a la salida a través del modelo ARMAX. Permite analizar las características de

identificación con respecto a desviación, consistencia y mejor estimador linealidad.

- MCE (Mínimos Cuadrados Extendidos): Se considera la presencia de ruido correlacionado a la salida, por lo cual es necesario extender el vector de información, a los valores de ruido los cuales se aproximan al error de predicción $\varepsilon(k)=y(k)-x(k)\Theta(k-1)$ y se trabaja como si fuese un sistema determinístico.

- MLH (Máximo de Likelihood): Es un método estadístico para la identificación de los parámetros de la planta a_i, b_i; y, c_i del proceso de ruido utilizando el residual $\eta(k)=y(k)-x(k)\Theta(k)$. Se trabaja con el gradiente del error.

A continuación se analiza con detalle el algoritmo recursivo.

2.1 MÍNIMOS CUADRADOS RECURSIVOS (MCR) [13], [3], [28].

Se trata de desarrollar un algoritmo recursivo como una variante del método de MCO; y, realizar un análisis de las características del método y consideraciones para su implementación computacional.

2.1.1 ALGORITMO RECURSIVO

Cuando se desarrolla el método de mínimos cuadrados recursivos, se parte del algoritmo inicial de mínimos

cuadrados ordinarios al instante k, y lo que se hace es añadir una medición. En la ecuación (1.17) del vector de parámetros al instante k está dada por:

$$\hat{\Theta}(k) = \left[X^T(k)X(k)\right]^{-1} X^T(k)Y(k)$$

se añade información al instante $(k+1)$ en la matriz de información ecuación (1.11) y en el vector de la salida ecuación (1.9), por lo que se expresa la matriz de información y el vector de salida como:

$$X(k+1) = \begin{bmatrix} x(n) \\ x(n+1) \\ \cdot \\ \cdot \\ x(k) \\ \hline x(k+1) \end{bmatrix} = \begin{bmatrix} X(k) \\ \hline x(k+1) \end{bmatrix} \qquad (2.1)$$

$$Y(k+1) = \begin{bmatrix} y(n) \\ y(n+1) \\ \cdot \\ \cdot \\ y(k) \\ \hline y(k+1) \end{bmatrix} = \begin{bmatrix} Y(k) \\ \hline y(k+1) \end{bmatrix} \qquad (2.2)$$

Reemplazando las ecuaciones (2.1) y (2.2) en la ecuación (1.17) de parámetros estimados se tiene:

$$\hat{\Theta}(k+1) = \left[X^T(k+1)X(k+1)\right]^{-1} X^T(k+1)Y(k+1) \qquad (2.3)$$

al desarrollar la matriz de información y el vector de salida en función de sus valores anteriores se puede escribir la ecuación (2.3) como:

$$\hat{\Theta}(k+1) = \left(\begin{bmatrix} X(k) \\ \hline x(k+1) \end{bmatrix}^T \begin{bmatrix} X(k) \\ \hline x(k+1) \end{bmatrix}\right)^{-1} \left(\begin{bmatrix} X(k) \\ \hline x(k+1) \end{bmatrix}^T \begin{bmatrix} Y(k) \\ \hline y(k+1) \end{bmatrix}\right)$$

$$= \left(X^T(k)X(k) + x^T(k+1)x(k+1)\right)^{-1}\left(X^T(k)Y(k) + x^T(k+1)y(k+1)\right)$$

Se define la matriz:

$$P(k) = \left(X^T(k)X(k)\right)^{-1} \tag{2.4}$$

Al volver a escribir las ecuaciones (1.17) y (2.3) en función de la ecuación (2.4) se tiene:

$$\hat{\Theta}(k) = P(k)X^T(k)Y(k) \tag{2.5}$$

$$\hat{\Theta}(k+1) = P(k+1)\left(X^T(k)Y(k) + x^T(k+1)y(k+1)\right) \tag{2.6}$$

Si en la ecuación (2.4) se reemplaza la ecuación (2.1) se tiene:

$$\begin{aligned} P(k+1) &= \left(X^T(k+1)X(k+1)\right)^{-1} \\ &= \left(X^T(k)X(k) + x^T(k+1)x(k+1)\right)^{-1} \\ &= \left(P^{-1}(k) + x^T(k+1)x(k+1)\right)^{-1} \end{aligned} \tag{2.7}$$

De la ecuación (2.3) se observa que para encontrar los valores de los parámetros estimados es necesario invertir la ecuación (2.7). Utilizando el Lema de Inversión de Matrices

se puede realizar dicha inversión de una manera más sencilla. Según dicho lema se tiene:

$$(A+BCD)^{-1} = A^{-1} - A^{-1}B\left(C^{-1} + DA^{-1}B\right)^{-1}DA^{-1}$$

Aplicando en la ecuación (2.7) de la matriz $P(k+1)$ se tiene:

$$A = P^{-1}(k)$$
$$B = x^T(k+1)$$
$$C = I$$
$$D = x(k+1)$$

$$P(k+1) = P(k) - P(k)x^T(k+1)\left(I + x(k+1)P(k)x^T(k+1)\right)^{-1}x(k+1)P(k)$$

$$= \left(I - \frac{P(k)x^T(k+1)x(k+1)}{1+x(k+1)P(k)x^T(k+1)}\right)P(k) \qquad (2.8)$$

En la ecuación (2.6) al reemplazar la ecuación (2.8) se tiene:

$$\hat{\Theta}(k+1) = \left(I - \frac{P(k)x^T(k+1)x(k+1)}{1+x(k+1)P(k)x^T(k+1)}\right)P(k)\left(X^T(k)Y(k) + x^T(k+1)y(k+1)\right)$$

$$\hat{\Theta}(k+1) = \left(I - \frac{P(k)x^T(k+1)x(k+1)}{1+x(k+1)P(k)x^T(k+1)}\right)\left(P(k)X^T(k)Y(k) + P(k)x^T(k+1)y(k+1)\right)$$

reemplazando la ecuación (2.5) en la ecuación anterior y desarrollando se tiene:

$$\hat{\Theta}(k+1) = \hat{\Theta}(k) - \frac{P(k)x^T(k+1)x(k+1)}{1+x(k+1)P(k)x^T(k+1)}\hat{\Theta}(k) + P(k)x^T(k+1)Iy(k+1) -$$
$$- \frac{P(k)x^T(k+1)x(k+1)P(k)x^T(k+1)y(k+1)}{1+x(k+1)P(k)x^T(k+1)}$$

se introduce la matriz identidad en el tercer término definida como:

$$I = \left(1+x(k+1)P(k)x^T(k+1)\right)^{-1}\left(1+x(k+1)P(k)x^T(k+1)\right)$$

agrupando términos se tiene:

$$\hat{\Theta}(k+1) = \hat{\Theta}(k) - \frac{P(k)x^T(k+1)x(k+1)}{1+x(k+1)P(k)x^T(k+1)}\hat{\Theta}(k) + \frac{P(k)x^T(k+1)y(k+1)}{1+x(k+1)P(k)x^T(k+1)}$$
$$+ \frac{P(k)x^T(k+1)x(k+1)P(k)x^T(k+1)y(k+1)}{1+x(k+1)P(k)x^T(k+1)}$$
$$- \frac{P(k)x^T(k+1)x(k+1)P(k)x^T(k+1)y(k+1)}{1+x(k+1)P(k)x^T(k+1)}$$

Al simplificar términos semejantes y agrupar se obtiene la ecuación:

$$\hat{\Theta}(k+1) = \hat{\Theta}(k) + \frac{P(k)x^T(k+1)}{1+x(k+1)P(k)x^T(k+1)}\left(y(k+1) - x(k+1)\hat{\Theta}(k)\right) \qquad (2.9)$$

La ecuación (2.9) puede ser simplificada al definir las siguientes relaciones:

$$\varepsilon(k+1) = y(k+1) - x(k+1)\hat{\Theta}(k) \qquad (2.10)$$

$$L(k+1) = \frac{P(k)x^T(k+1)}{1+x(k+1)P(k)x^T(k+1)} \qquad (2.11)$$

$$P(k+1) = \left(I - \frac{P(k)x^T(k+1)x(k+1)}{1+x(k+1)P(k)x^T(k+1)}\right)P(k) \qquad (2.12)$$

reemplazando las ecuaciones (2.10) y (2.11) en la ecuación (2.9) se tiene:

$$\hat{\Theta}(k+1) = \hat{\Theta}(k) + L(k+1)\,\varepsilon(k+1) \qquad (2.13)$$

La ecuación (2.13) muestra la naturaleza recursiva del método, ya que los nuevos valores de $\hat{\Theta}(k+1)$ se hallan en función del anterior $\hat{\Theta}(k)$ más una corrección en función del error.

2.1.2 CARACTERÍSTICAS Y CONSIDERACIONES COMPUTACIONALES DEL MÉTODO DE MCR.

Al algoritmo desarrollado hasta aquí se deben realizar algunas consideraciones adicionales que son útiles para la implementación computacional del método.

2.1.2.1 Ventajas de los Mínimos Cuadrados Recursivos.

Las principales ventajas de este algoritmo en relación con el método de mínimos cuadrados ordinarios son:

- Se evita la inversión de la matriz (X^TX), pues se reduce a la inversión de un escalar.

- Es un algoritmo adecuado para implementarlo en línea debido a que utiliza la información anterior y lo que se hace es corregir el valor de los parámetros estimados al ir incrementando información.

- La cantidad de memoria que se necesita para hallar los parámetros no depende del número de mediciones, y se mantiene fija a lo largo de todo el proceso de identificación.

2.1.2.2 Identificabilidad [13]

El modelo ARMA describe a un sistema en base a sus mediciones tanto de entrada como de salida, este modelo puede ser representado como:

$$y(k) + \sum_{i=1}^{n} a_i y(k-i) = \sum_{i=r}^{m} b_i u(k-i) \qquad (2.14)$$

donde r es el retardo entre la salida y la entrada.

Se aplica la transformada Z a la ecuación (2.14) y se expresa como:

$$(1+A(z))y(k) = z^{-r}B(z)u(k) \qquad (2.15)$$

Un diagrama de bloques del modelo ARMA se muestra en la figura 2.1.

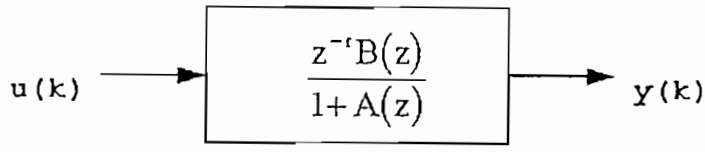

Fig. 2.1 Modelo ARMA

Una característica importante del modelo ARMA, es que el vector de estado esta formado por los valores de las entradas y salidas del sistema, lo que lo hace muy factible para utilizarlo en la identificación paramétrica discreta.

El modelo ARMA es una representación canónica de tal manera que tiene el menor número de parámetros y en consecuencia la identificación de los parámetros $\hat{\Theta}$ es única, a esta característica se la conoce con el nombre de identificabilidad y por definición se dice que: Los parámetros $\hat{\Theta}$ que tienen la propiedad de que un valor y solamente uno de $\hat{\Theta}$ que hace que $J(\Theta)$ sea mínimo es identificable.

2.1.2.3 Excitación Persistente [3],[31]

Los algoritmos de identificación paramétrica discreta basan su acción en la posibilidad de que la entrada varíe lo suficiente (tenga un amplio rango de frecuencia) como para que la dinámica de la planta pueda ser reflejada a través de las mediciones de entrada y salida. A esta propiedad de la señal de entrada se conoce con el nombre de excitación

persistente y es una condición necesaria para que se pueda estimar parámetros.

Para analizar la propiedad de excitación persistente en la entrada, se toma el vector de información:

$$x(k) = \begin{pmatrix} y(k-1) & y(k-2) & \cdots & y(k-n) & | & u(k-1) & u(k-2) & \cdots & u(k-n) \end{pmatrix} \tag{2.16}$$

si en la ecuación (2.16) se define para facilidad de cálculo:

$$x(k) = \begin{pmatrix} y(k) & | & u(k) \end{pmatrix} \tag{2.17}$$

en la matriz de información al reemplazar la ecuación (2.17) se tiene:

$$X(k) = \begin{pmatrix} x(n) \\ \vdots \\ x(k) \end{pmatrix} = \begin{pmatrix} y(n) & u(n) \\ \vdots & \vdots \\ y(k) & u(k) \end{pmatrix} \tag{2.18}$$

al obtener el producto $X^T(k)X(k)$ se tiene:

$$X^T(k)X(k) = \begin{pmatrix} Y^T(k)Y(k) & | & Y^T(k)U(k) \\ \cdots & | & \cdots \\ U^T(k)Y(k) & | & U^T(k)U(k) \end{pmatrix} \tag{2.19}$$

La entrada es de excitación persistente si el término inferior derecho de la matriz $X^T(k)X(k)$ es no singular, esta es una condición de excitación.

En general para un conjunto de datos suficientemente grande la matriz de covarianzas C_n debe estar definida positiva. Esto es:

$$C_n = \begin{pmatrix} c(0) & c(1) & \cdots & c(n-1) \\ c(1) & c(0) & \cdots & c(n-2) \\ \vdots & \vdots & \vdots & \vdots \\ c(n-1) & c(n-2) & \cdots & c(0) \end{pmatrix} \qquad (2.20)$$

donde:

$$c(i) = \lim_{k\to\infty} \frac{1}{k} \sum_{j=1}^{k} u(j)u(j-i) \qquad (2.21)$$

Entonces $u(k)$ es de excitación persistente de orden n si la matriz C_n dada por (2.20) es definida positiva.

Esto implica o garantiza la estimación mediante MCR de un modelo de respuesta impulso finita (FIR) con n parámetros será consistente y la varianza de los estimados tiende a cero como $\frac{1}{k}$ si la señal de entrada es de excitación persistente de orden n. Más adelante se analiza la característica de consistencia y del gradiente.

Finalmente cabe señalar que una señal de excitación persistente es de orden n si:

$$\lim_{k\to\infty} \frac{1}{k} \left(\sum_{i=1}^{k} A(z)u(i) \right)^2 > 0 \qquad (2.22)$$

En efecto se tiene que:

$$A(z) = a_0 z^{n-1} + a_1 z^{n-2} + \ldots + a_{n-1}$$

$$\lim_{k\to\infty} \frac{1}{k} \left(\sum_{i=1}^{k} A(z)u(i) \right)^2 =$$

$$= \lim_{k\to\infty} \frac{1}{k} \left[\sum_{i=1}^{k} a_0 u(i+n+1) + \ldots\ldots + a_{n-1} u(i) \right]^2$$

$$= a^T C_n a$$

donde C_n es la matriz dada por (2.20).

Aplicando las definiciones anteriores es posible demostrar que el ruido es una señal aleatoria no predecible de cualquier orden, puesto que en un proceso de ruido el error de predicción no es cero (la señal no puede predecirse), entonces siempre existirán términos diferentes de cero en la ecuación (2.22) y la matriz (2.20) será definida positiva, un análisis detallado se encuentra en la referencia [3].

2.1.2.4 Condiciones Iniciales [13]

Existen algunas opciones que son las más recomendadas para establecer las condiciones iniciales del algoritmo de identificación y son:

- Se establece $\hat{\Theta}(k) = 0$, $P(k+1) = \alpha I$, donde α es un escalar que esta comprendido entre 1000 y 10000 para que la convergencia de los parámetros sea rápida. Una matriz $P(k)$ grande indica mayor incertidumbre en el valor inicial pues

como se muestra más adelante es proporcional a la matriz de covarianzas.

- Se toma un lote de k mediciones tal que $k>2n$ y se utiliza el algoritmo de mínimos cuadrados ordinarios para hallar $\hat{\Theta}(k)$ mediante la ecuación (1.17), luego se encuentra también $P(k)$ y $L(k+1)$ y se utilizan estos valores como punto de partida para empezar a identificar parámetros por el método de mínimos cuadrados recursivos.

- Una forma final de establecer condiciones iniciales para el algoritmo es tener cierta experiencia con el sistema para de cierta manera establecer el valor de los parámetros que se esta buscando y esperar que el algoritmo converja a los verdaderos valores en pocas iteraciones.

2.1.2.5 Factor de Olvido [13] [3] [28] [29].

El dar igual importancia a todas las mediciones que se adquieren para estimar los parámetros en un principio puede ser aceptable, sin embargo a medida que se van adquiriendo más valores de entrada - salida se debe tener en cuenta algún criterio para dar mayor peso a las últimas mediciones con respecto a las primeras, por cuanto las últimas reflejan de mejor forma la dinámica de la planta, sobre todo si existen cambios lentos de la misma. Se debe entonces implementar un mecanismo para ponderar las mediciones.

La forma más sencilla y adecuada de ponderar las mediciones es mediante un factor que se introduce en la función de costo, llamado factor de olvido.

Para analizar la influencia del factor de olvido en la identificación paramétrica se define el criterio de minimización de la siguiente forma:

$$J(\Theta) = \sum_{k=n}^{k} w(k)e^2(k;\Theta) = E^T(k)W(k)E(k)$$

$$J(\Theta) = \begin{pmatrix} e_n & e_{n+1} & \cdots & e_k \end{pmatrix} \begin{pmatrix} w_n & 0 & \cdots & 0 \\ 0 & w_{n+1} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & w_k \end{pmatrix} \begin{pmatrix} e_n \\ e_{n+1} \\ \vdots \\ e_k \end{pmatrix} \quad (2.23)$$

Este criterio da lugar a la variante de los mínimos cuadrados ponderados, en donde la matriz de ponderación $W(k)$ es positiva definida, y $w(k)$ es el factor de olvido que debe ser positivo. Se define el factor de olvido como $w(i)=a\gamma^{k-i}$ más adelante en esta misma sección se detalla aspectos sobre el factor de olvido, por ahora se pone énfasis en el desarrollo del algoritmo con factor de olvido o mínimos cuadrados ponderados.

Minimizando el índice de funcionamiento de la ecuación (2.23) se llega a tener en forma similar que en el caso de mínimos cuadrados ordinarios una expresión para hallar el valor de los parámetros estimados:

$$\hat{\Theta}(k) = \left(X^T(k)W(k)X(k)\right)^{-1} X^T(k)W(k)Y(k) \quad (2.24)$$

La ecuación (2.24) permite la evaluación de un lote de datos teniendo en cuenta el factor de olvido, sin embargo como se busca un algoritmo recursivo, se desarrolla la ecuación

(2.23) aumentando una medición más y de manera similar que en el caso de los mínimos cuadrados recursivos se llega a la misma expresión para el valor de los parámetros estimados.

$$\hat{\Theta}(k+1) = \hat{\Theta}(k) + L(k+1)\,\varepsilon(k+1) \qquad (2.25)$$

donde se redefine los siguientes términos de la ecuación (2.25) como:

$$\varepsilon(k+1) = y(k+1) - x(k+1)\hat{\Theta}(k) \qquad (2.26)$$

$$L(k+1) = \frac{\frac{1}{\gamma}P(k)x^{T}(k+1)}{\frac{1}{a} + \frac{x(k+1)P(k)x^{T}(k+1)}{\gamma}} \qquad (2.27)$$

$$P(k+1) = \frac{1}{\gamma}\left(I - \frac{\frac{1}{\gamma}P(k)x^{T}(k+1)x(k+1))}{\frac{1}{a} + \frac{x(k+1)P(k)x^{T}(k+1)}{\gamma}}\right)P(k) \qquad (2.28)$$

El hecho de incluir el factor de olvido en los algoritmos de identificación paramétrica discreta evita la posibilidad de inestabilidad numérica, la cual se presenta por la tendencia de que los valores de $\hat{\Theta}(k)$ tiendan a un valor constante debido a que la matriz de covarianza $P(k)$ tiende a cero, además el factor de olvido al evitar que la matriz de covarianza $P(k)$ tienda a cero permite rastrear o actualizar cualquier cambio o variación lenta de los parámetros, si la variación es fuerte, entonces habrá que esperar que el

algoritmo trabaje con las nuevas mediciones que reflejen la nueva dinámica de la planta.

Básicamente existen dos tipos de factores de olvido:

- Constantes.
- Variables.

Se analiza primero el algoritmo básico con el factor de olvido constante, donde $w(i)=\gamma^{k-i}$, $a=1$ para este caso se escoge γ dentro del intervalo $0.95\leq\gamma\leq0.99$ típicamente $\gamma=0.985$ por lo que las ecuaciones (2.27) y (2.28) se expresan como:

$$L(k+1)=\frac{\frac{1}{\gamma}P(k)x^{T}(k+1)}{1+\frac{x(k+1)P(k)x^{T}(k+1)}{\gamma}} \qquad (2.29)$$

$$P(k+1)=\frac{1}{\gamma}\left(I-\frac{\frac{1}{\gamma}P(k)x^{T}(k+1)x(k+1))}{1+\frac{x(k+1)P(k)x^{T}(k+1)}{\gamma}}\right)P(k) \qquad 2.30)$$

al ser γ un valor cercano a 1 lo que se hace al dividir la matriz de covarianza para γ es impedir que esta se haga cero, por lo que siempre se incrementa ligeramente a $P(k)$, este es un objetivo que debe cumplir el factor de olvido.

El otro objetivo es dar más peso a las últimas mediciones esto se puede visualizar de la ecuación (2.23) para este caso se tiene:

$$J(\Theta) = \begin{pmatrix} e_n & e_{n+1} & \cdots & e_{k-1} & e_k \end{pmatrix} \begin{pmatrix} \gamma^{k-n} & 0 & \cdots & 0 & 0 \\ 0 & \gamma^{k-n-1} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \gamma^{1} & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} e_n \\ e_{n+1} \\ \vdots \\ e_{k-1} \\ e_k \end{pmatrix}$$

$$= e_n^2\gamma^{k-n} + e_{n+1}^2\gamma^{k-n-1} + e_{n+2}^2\gamma^{k-n-2} + \ldots + e_{k-2}^2\gamma^2 + e_{k-1}^2\gamma^1 + e_k^2 \qquad (2.31)$$

En la ecuación (2.31) puesto que $\gamma<1$, las potencias de γ son cada vez más pequeñas, se da menor importancia a las primeras mediciones y más a las últimas.

Se analiza a continuación algunas variantes importantes del factor de olvido.

Una variante del factor de olvido esta dada por $w(i) = (1-\gamma)\gamma^{k-i}$, donde $a = (1-\gamma)$, que corresponde a un filtro de primer orden. El factor $a = (1-\gamma)$ produce una ganancia unitaria para cuando se tiene errores constantes.

En la función de costo:

$$J(\Theta) = E^T(k)W(k)E(k) \qquad (2.32)$$

si se adquiere k mediciones cuando el error es constante se tiene:

$$e_1 = e_2 ..... = e_k = e$$

al reemplazar la igualdad anterior en la ecuación (2.23) se tiene:

$$J(\Theta) = \begin{pmatrix} e & e & \cdots & e & e \end{pmatrix} \begin{pmatrix} a\gamma^k & 0 & \cdots & 0 & 0 \\ 0 & a\gamma^{k-1} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & a\gamma^1 & 0 \\ 0 & 0 & \cdots & 0 & a \end{pmatrix} \begin{pmatrix} e \\ e \\ \vdots \\ e \\ e \end{pmatrix} \qquad (2.33)$$

multiplicando y agrupando (2.33) se llega a:

$$\begin{aligned} &= e^2 a\gamma^k + e^2 a\gamma^{k-1} + ... + e^2 a\gamma + e^2 a \\ &= e^2 a\left(\gamma^k + \gamma^{k-1} + ...+\gamma + 1\right) \end{aligned} \qquad (2.34)$$

en la ecuación (2.34) si el término $\left(\gamma^k + \gamma^{k-1} + ...+\gamma + 1\right)$ se reemplaza por su serie geométrica se llega a:

$$= e^2(1-\gamma)\frac{\left(1-\gamma^{k+1}\right)}{(1-\gamma)} = e^2\left(1-\gamma^{k+1}\right) \qquad (2.35)$$

de donde se observa que para cuando k se hace grande (al estabilizarse el error) $\gamma^{k+1} \cong 0$, entonces la ganancia es aproximadamente 1.

Se tiene también factores de olvido variables con el tiempo al instante k tales como:

- $\gamma(k) = \frac{k}{k+1}$ que tiene el mismo efecto que el anterior pues para cuando k se hace muy grande la ganancia tiende a 1, pero al inicio el factor de olvido es menor que 1. Aquí $\gamma(k)$ siempre es menor que 1 para que la matriz de covarianza $P(k)$ no tienda a cero.

- Otro factor de olvido variable es el factor exponencial definido como $\gamma(k) = \gamma_i + (1-\gamma_i)(1-e^{-k/\tau})$. Si $k=0$ se tiene el factor de olvido inicial γ_i y a medida que el valor de k crece, y se hace muy grande $\gamma(k) \to 1$. Mediante pruebas en linea se determina un τ adecuado.

2.1.2.6 Algoritmo Final

Luego de haber establecido el algoritmo de mínimos cuadrados recursivos así como las distintas características del mismo se procede a presentar los pasos del algoritmo final de identificación.

Se indica que excepto las rutinas de generación de datos en el proceso de simulación y la rutina de adquisición y salida de datos en tiempo real las restantes funciones para los dos casos son las mismas.

1.- Establecer modo de trabajo simulación o tiempo real.
2.- Inicializar Θ y $P = \alpha I$.
3.- Empezar proceso de identificación para cuando el número de iteraciones es mayor que el orden de $y(k)$ por identificar.

4.- Leer valores de $y(k)$; $u(k)$.

5.- Actualizar el valor del vector $x(k)$.

6.- Encontrar el error $e(k+1)$ según la ecuación:

$$\varepsilon(k+1) = y(k+1) - x(k+1)\hat{\Theta}(k)$$

7.- Calcular la matriz $L(k+1)$ de acuerdo a:

$$L(k+1) = \frac{\frac{1}{\gamma}P(k)x^T(k+1)}{\frac{1}{a}+\frac{x(k+1)P(k)x^T(k+1)}{\gamma}}$$

8.- Hallar la matriz $P(k+1)$ de acuerdo a:

$$P(k+1) = \frac{1}{\gamma}\left(I-\frac{\frac{1}{\gamma}P(k)x^T(k+1)x(k+1))}{\frac{1}{a}+\frac{x(k+1)P(k)x^T(k+1)}{\gamma}}\right)P(k)$$

9.- Calcular el valor de los estimados $\hat{\Theta}(k+1)$ según la ecuación:

$$\hat{\Theta}(k+1) = \hat{\Theta}(k) + L(k+1)\,\varepsilon(k+1)$$

10.- Incrementar k.

11.- Ir al paso 4.

2.2 MÍNIMOS CUADRADOS ESTOCÁSTICOS (MCS) [1], [13], [16]

Al añadir ruido a la salida el proceso de identificación se ve afectado por lo que se desarrolla un estudio teórico para determinar como el algoritmo de mínimos cuadrados recursivos se ve afectado con la presencia de ruido en los datos de salida y la forma en que cambia el modelo para identificar el nuevo sistema.

El ruido que se suma a la salida puede ser generado a través de distintos algoritmos. Se utilizan tres tipos de ruido:

- Señal randómica.
- Señal binaria pseudoaleatoria (PRBS).
- Ruido estadístico.

Cuando se identifica los parámetros de un sistema en presencia de ruido se tiene incertidumbre en los valores de dichos parámetros debido al carácter aleatorio del proceso. Se debe garantizar bajo que condiciones se puede converger a los verdaderos valores por lo que es necesario definir ciertas características que debe tener el algoritmo de identificación. Estas características son:

- Consistencia.
- Desviación.
- Mejor estimador lineal.

Las tres características anteriores se analizan aplicadas al método de mínimos cuadrados recursivos al ser sometido a la presencia de ruido.

2.2.1 ASPECTOS ESTOCÁSTICOS DEL MÉTODO DE MCR.

Al estudiar el método de mínimos cuadrados recursivo sometido a ruido es necesario analizar las características de este, así como el modelo que se utilizará para identificar parámetros y algunas características de los parámetros encontrados.

2.2.1.1 Ruido Blanco [1], [9], [12], [14], [30]

En general el término ruido blanco se utiliza para indicar que el espectro de densidad de potencia de una señal dada contiene todas las frecuencias posibles.

De la definición se observa que para que una señal sea considerada ruido blanco la transformada de Fourier de la función de correlación Rx, esto es la función de densidad de espectro de potencia $G_x(jw)$ debe ser un valor constante a lo largo de todo el espectro de frecuencias como se muestra en la figura 2.2. Es decir:

$$G_x(jw) = \int_{-\infty}^{+\infty} Rx(\tau)e^{-jw\tau}d\tau = c \qquad (2.36)$$

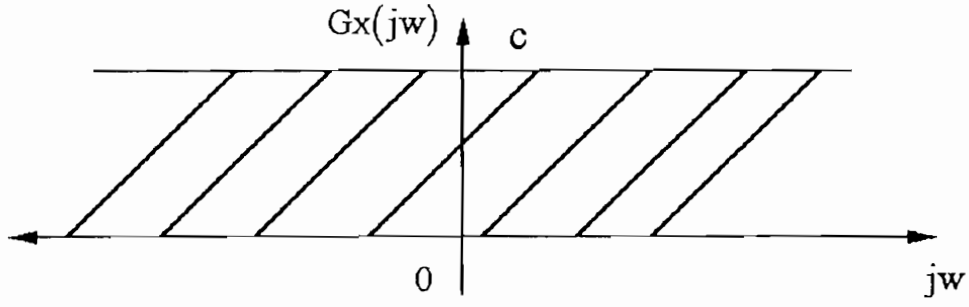

Fig. 2.2 Densidad espectro de potencia del ruido blanco

Para hallar un proceso continuo que genere ruido blanco se toma la transformada inversa de Fourier del espectro antes indicado y se tiene:

$$Rx(\tau) = \int_{-\infty}^{+\infty} ce^{jw\tau} dw = c\delta(\tau) \qquad (2.37)$$

donde la función $Rx(\tau)$ tiene las siguientes características:

$$c\,\delta(\tau) = \begin{cases} \infty & ;\tau = 0 \\ 0 & ;\tau \neq 0 \end{cases} \qquad (2.38)$$

De las características de la ecuación (2.38) se observa que no se puede generar una señal continua con dicha función $Rx(\tau)$ desde ningún sistema real. Por lo que se tiene que el ruido blanco continuo no es nada más que una idealización y que en la práctica no se lo puede conseguir.

Sin embargo para ciertos procesos en que se necesita la presencia de ruido "blanco" se utiliza señales de un amplio rango de frecuencia o banda ancha, las mismas que pueden ser generadas con sistemas discretos.

El ruido blanco discreto se define como una secuencia de valores discretos no correlacionados que cumplen con la misma definición de ruido blanco continuo, esto es su densidad de espectro de potencia es constante $G_x(jw) = c$, pero para sistemas discretos se tiene la función $Rx(k)$ como:

$$Rx(k) = \int_{-\pi}^{+\pi} e^{jkw} c\,dw$$

$$= \frac{2c}{k}\sin(k\pi)$$

donde:

$$Rx(k) = \begin{cases} 2\pi c & k=0 \\ 0 & k=\pm1,\pm2,.... \end{cases}$$

(2.39)

Esto implica que los valores discretos del proceso en diferentes instantes son no correlacionados y esta secuencia de ruido blanco si es factible de realizar.

En este trabajo se utiliza ruido blanco gaussiano pues aún en el caso de no tenerlo, por el teorema de Límite Central, la suma de muchos proceso aleatorios tiende a ser un proceso gaussiano. En la práctica se puede aproximar un proceso real a un proceso gaussiano.

Un proceso gaussiano esta completamente descrito mediante su valor medio y la desviación estándar. Para el presente trabajo se utilizará ruido blanco de media cero $\mu(x)=0$ y desviación estándar σ. El ruido debe ser de media cero para que no exista un offset en el valor de los parámetros estimados.

En este trabajo se utiliza tres tipos de algoritmos para generar ruido blanco discreto entre los que se tiene:

- Señal aleatoria a través de la función **random** del computador. En este caso lo que se hace es tener un

semillero de valores entre [0 y 1] a los cuales se los desplaza entre [-1 y 1] para tener media cero.

- Señal binaria pseudoaleatoria (PRBS, Pseudo-Random Binary Signal) de dos niveles +V y -V, que se la genera a través de un registro de desplazamiento de 10 estados como se indica en la figura 2.3.

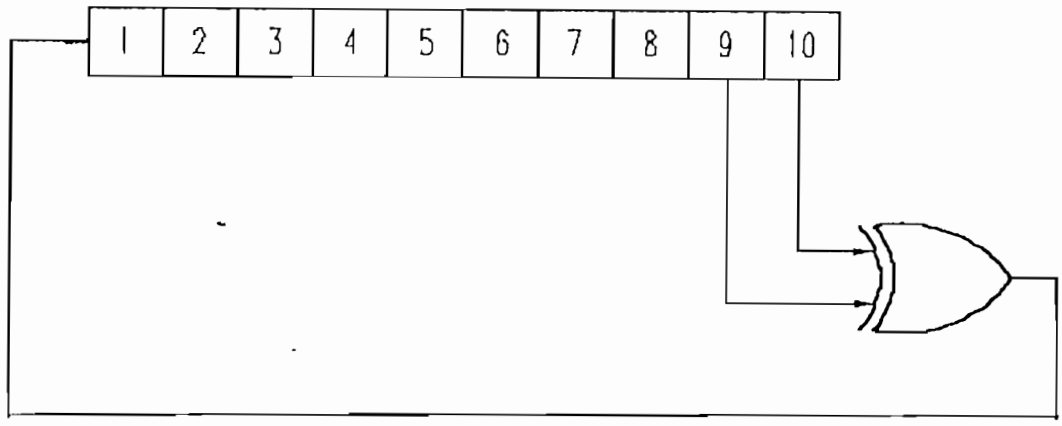

Fig. 2.3 Registro de desplazamiento para generar PRBS

Cada estado tiene el valor de 1 ó 0 y la salida se toma de la operación lógica OR-Exclusivo y se retroalimenta.

- Ruido Estadístico, es un algoritmo para generar ruido blanco, sin embargo este tiene un tratamiento adicional partiendo de un semillero de datos comprendido entre 0 y 1.

El algoritmo consiste en:

1. Generar dos variables independientes R_0 y R_1 uniformemente distribuidas en el intervalo de (0,1) y calcular:

$$V_1 = 2R_0 - 1$$
$$V_2 = 2R_1 - 1 \qquad (2.40)$$

2. Evaluar $S = V_1^2 + V_2^2$

3. Si $S \geq 1$ regresar al paso 1, en caso contrario continuar.

4. Calcular las variables distribuidas normalmente mediante las fórmulas:

$$\xi_1 = V_1 \sqrt{\frac{-2\ln S}{S}}$$
$$\xi_2 = V_2 \sqrt{\frac{-2\ln S}{S}} \qquad (2.41)$$

Para probar la validez de este método, se observa que:

Si $S<1$ en el paso 3, el punto en el plano cartesiano de coordenadas V_1, V_2 es un punto distribuido uniformemente que cae dentro del círculo unitario, con lo que se logra que las variables aleatorias ξ_1 , ξ_2 sean independientes.

Para la generación de los números aleatorios R_0 y R_1 se utiliza el siguiente algoritmo.

1. Definir un R (R semilla de números aleatorios tal que $0<R\leq 1$).

2. Asignar $R = 0.92821 \cdot R + 0.211327$

3. Asignar $R = R - \mathrm{INT}(R)$

4. Asignar a $R_i = R$ (i=0,1)

5. Ir al paso 2

2.2.1.2 Ruido Correlacionado

Se conoce como ruido coloreado a la señal que produce un espectro de frecuencia de banda limitada o angosta.

Para obtener ruido coloreado lo que se hace es tomar ruido blanco y pasarlo a través de un filtro, tal como se muestra en la figura 2.4.

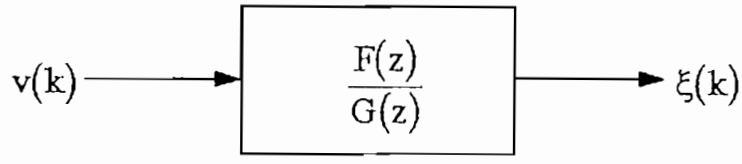

Fig. 2.4 Generación de Ruido Correlacionado

Debido a la consecuencia teorema de Factorización Espectral que establece que:

"Cualquier proceso estocástico puede modelarse como la salida de ruido blanco a través de un filtro".

En el caso de que:

$$\xi(k) = (1+C(z))v(k) \qquad (2.42)$$

se tiene:

$$\xi(k) = v(k) + c_1 v(k-1) + c_2 v(k-2) + \ldots\ldots + c_p v(k-p) \qquad (2.43)$$

que es el modelo de proceso de ruido coloreado que se utiliza en esta tesis.

2.2.1.3 Modelo ARMAX

El modelo ARMAX tiene una estructura similar al modelo ARMA, pero además se añade ruido a la salida de ahí la letra adicional X (exogenorus variable), que significa que no se puede predecir la magnitud de ruido.

La representación matemática del método ARMAX es:

$$y(k) + \sum_{i=1}^{n} a_i y(k-i) = \sum_{i=r}^{m} b_r u(k-i) + v(k) + \sum_{i=1}^{p} c_i v(k-i) \qquad (2.44)$$

aplicando la transformada Z a la ecuación (2.44) se tiene:

$$(1+A(z))y(k) = z^{-r}B(z)u(k) + (1+C(z))v(k) \qquad (2.45)$$

El modelo puede ser expresado en el diagrama de bloques de la figura 2.5:

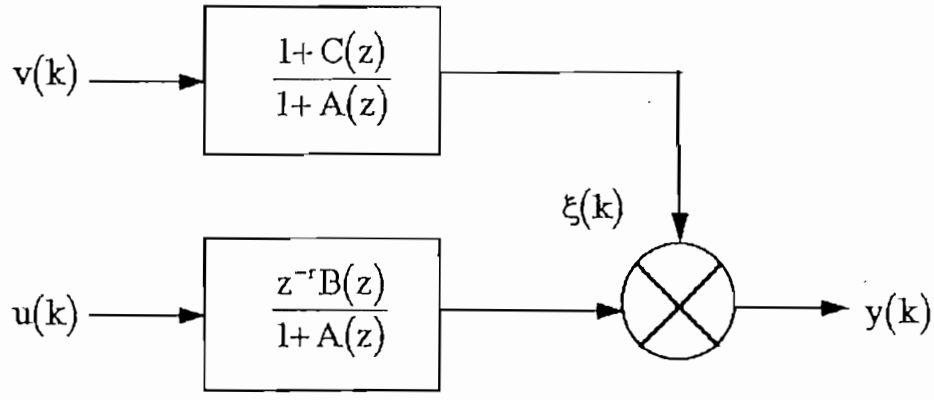

Fig. 2.5 Diagrama de bloques del modelo ARMAX

Al tratarse de ruido blanco la función de transferencia $(1+C(z))$ vale 1, mientras que si la señal que se suma a la

salida es ruido coloreado la función de transferencia $(1+C(z))$ es distinta de 1, pues aquí se considera al polinomio $(1+C(z))$ que representa la dinámica del proceso de ruido como se detalló en la ecuación (2.43).

2.2.2 CARACTERÍSTICAS DEL ALGORITMO DE MCS [13]

Al analizar el algoritmo de identificación paramétrica discreta mediante el método de los mínimos cuadrados recursivos, no se comentó nada acerca de la posibilidad de que los datos tomados fueran contaminados por ruido, y de como esto afectaría al algoritmo.

A continuación se considera el algoritmo de mínimos cuadrados estocásticos y se tiene que:

$$y(k) = a^T\Theta^\circ(k) + v(k) \qquad (2.46)$$

en términos de matriciales se tiene:

$$Y(k) = A(k)\,\Theta^\circ(k) + V(k) \qquad (2.47)$$

Se denota como $A(k)$ a la matriz de información pues se considera que se trabaja con los verdaderos valores de los parámetros ($A(k)$ es determinística), se define a $v(k)$ como una variable aleatoria de tipo ruido blanco con media cero y covarianza conocida, por lo que se tiene:

$$
\begin{aligned}
E\left[v(k)v(j)\right] &= \sigma^2 \quad (k=j) \\
&= 0 \quad (k\neq j) \\
E\left[V(k)V^T(k)\right] &= \sigma^2 I
\end{aligned}
\tag{2.48}
$$

En este caso se puede considerar el modelo determinístico ($A(k)$ es determinística) y los errores se modelan como una variable aleatoria $v(k)$.

De manera similar que para el caso de los mínimos cuadrados recursivos, en los mínimos cuadrados estocásticos el problema es encontrar $\Theta(k)$ tal que minimice la ecuación de error, se trabaja con el criterio del error cuadrático.

$$
\begin{aligned}
J(\Theta) &= \left(Y(k) - A\Theta(k)\right)^T \left(Y(k) - A\Theta(k)\right) \\
J(\Theta) &= \sum_{k=n}^{k} e^2(k;\Theta)
\end{aligned}
\tag{2.49}
$$

Aquí se debe indicar que los errores que pueden darse dependen no solamente de la aproximación de $\Theta(k)$ a sus valores verdaderos, sino también del ruido aleatorio. La solución será aleatoria porque los datos ó información en los que se basa son en sí mismo es aleatorios.

La solución de la ecuación (2.49) ya se la obtuvo en el algoritmo de mínimos cuadrados ordinarios y se tiene que:

$$
\hat{\Theta}(k) = \left(A^T(k)A(k)\right)^{-1} A^T(k)Y(k) \tag{2.50}
$$

Se debe indicar que aunque se utilice los verdaderos valores de Θ° no se debe esperar obtener un error igual a cero debido a los efectos del ruido por lo que se tiene que ampliar el concepto de lo que constituye una buena estimación.

Un buen estimador debe tener las siguientes características:

- Consistencia.
- No desviación.
- Mejor estimador lineal.

A continuación se analiza cada propiedad.

2.2.2.1 Consistencia

Un estimador de $\hat{\Theta}$ de parámetros Θ° es consistente si luego de un gran número de iteraciones la diferencia entre $\hat{\Theta}$ y Θ° se vuelve despreciable. Se hace explícita la dependencia de la longitud de datos escribiendo $\hat{\Theta}(k)$. Se utiliza el criterio medio cuadrático para definir cuando se desprecia la diferencia $\hat{\Theta}(k) - \Theta^{\circ}$. Entonces:

$$\lim_{k\to\infty} E\left(\left(\hat{\Theta}(k) - \Theta^{\circ}\right)^{T}\left(\hat{\Theta}(k) - \Theta^{\circ}\right)\right) = 0$$
$$\lim_{k\to\infty} \mathrm{tr}\, E\left(\left(\hat{\Theta}(k) - \Theta^{\circ}\right)\left(\hat{\Theta}(k) - \Theta^{\circ}\right)^{T}\right) = 0 \qquad (2.51)$$

Esto pues solamente interesa las varianzas.

Si la ecuación (2.51) se cumple se dice que $\hat{\Theta}(k)$ converge a Θ° en el sentido medio cuadrático cuando k se hace muy grande.

Entonces para el estimador de mínimos cuadrados recursivo se tiene según (2.50) y (2.47):

$$\begin{aligned}\hat{\Theta}(k)-\Theta^\circ &= \left(A^TA\right)^{-1}A^TY-\Theta^\circ \\ &= \left(A^TA\right)^{-1}A^T\left(A\Theta^\circ+V\right)-\Theta^\circ \\ &= \left(A^TA\right)^{-1}\left(A^TA\right)\Theta^\circ+\left(A^TA\right)^{-1}A^TV-\Theta^\circ \\ &= \Theta^\circ+\left(A^TA\right)^{-1}A^TV-\Theta^\circ \\ &= \left(A^TA\right)^{-1}A^TV\end{aligned} \qquad (2.52)$$

Luego :

$$\begin{aligned}E\left\{\left(\hat{\Theta}(k)-\Theta^\circ\right)\left(\hat{\Theta}(k)-\Theta^\circ\right)^T\right\} &= E\left\{\left(A^TA\right)^{-1}A^TVV^TA\left(A^TA\right)^{-1}\right\} \\ &= \left(A^TA\right)^{-1}A^T\,E\left[VV^T\right]A\left(A^TA\right)^{-1} \\ &= \sigma^2\left(A^TA\right)^{-1}\end{aligned} \qquad (2.53)$$

En el desarrollo anterior se considera que A es una matriz conocida y no aleatoria y que $E\left(VV^T\right)=\sigma^2 I$. Por lo tanto se dice que el estimador de mínimos cuadrados es consistente o que $\hat{\Theta}$ converge Θ° si:

$$\lim_{k\to\infty} \operatorname{tr}\sigma^2\left(A^TA\right)^{-1}=0 \qquad (2.54)$$

Cuando se desarrollo el algoritmo de MCR se utilizó la definición $P(k) = (X^T X)^{-1}$ ecuación (2.4), que en este caso equivale a $P(k) = (A^T A)^{-1}$. La interpretación que tiene $P(k)$ es la de que es proporcional a la covarianza pues:

$$E\left[\left(\hat{\Theta}(k)-\Theta^\circ\right)\left(\hat{\Theta}(k)-\Theta^\circ\right)^T\right] = C$$
$$= \sigma^2 (A^T A)^{-1}$$

entonces $P(k) = \frac{C}{\sigma^2}$ si $\sigma=1$ para un proceso gaussiano normal (0,1), la matriz $(A^T A)^{-1}$ es igual a la covarianza.

2.2.2.2 No Desviación

Si la diferencia entre el valor promedio o esperado de los parámetros $\hat{\Theta}(k)$ y el parámetro verdadero Θ° es igual a un cierto valor b (bias), entonces la estimación es desviada, en caso contrario $(b=0)$, la estimación es no desviada y no presenta offset.

Para analizar esta característica se parte de la ecuación (2.52) en donde:

$$\hat{\Theta}(k)-\Theta^\circ = (A^T A)^{-1} A^T V$$

se encuentra el valor esperado:

$$E\left[\hat{\Theta}(k)-\Theta^{\circ}\right]=E\left[\left(A^{T}A\right)^{-1}A^{T}V\right]$$
$$=\left(A^{T}A\right)^{-1}A^{T}E[V] \qquad (2.55)$$
$$=0$$

La ecuación (2.55) es cero debido a que se considera ruido blanco de media igual a cero es decir $E[V]=0$.

Esta característica es muy importante por que establece que el algoritmo de identificación paramétrica discreta basado en el método de los mínimos cuadrados recursivos es no desviado y consistente cuando se tenga perturbaciones de ruido blanco en las señales de entrada o salida.

De esta forma cuando se considera ruido blanco, los MCS tienen el mismo resultado y aplicación que los MCR determinísticos. En otras palabras el MCR es aplicable hasta en procesos estocásticos de ruido blanco.

A continuación se desarrolla un ejemplo para demostrar las características de consistencia y no desviación analizadas en el método de mínimos cuadrados estocásticos.

Sea un modelo de la siguiente forma:

$$y(k) = a^{\circ} + b^{\circ}k + v(k)$$

donde este modelo corresponde a las observaciones o mediciones de una masa que se mueve con movimiento rectiliñeo uniformemente variado, y; con condiciones iniciales de posición y velocidad desconocida.

Entonces:

$$y(t) = a^{\circ} + b^{\circ}t = \begin{bmatrix} 1 & t \end{bmatrix} \begin{bmatrix} a^{\circ} \\ b^{\circ} \end{bmatrix}$$

Para a°= posición inicial,

b°= velocidad inicial y

$t = kT, \quad k = 0,1,2...$

Para observaciones en instantes $t_0, t_1, ..., t_k$ se tiene que la matriz de información $A(k)$ esta dada por:

$$A(k) = \begin{bmatrix} 1 & t_0 \\ 1 & t_1 \\ \vdots & \vdots \\ 1 & t_k \end{bmatrix}$$

Si se considera velocidad cero por simplicidad para reducir a un problema escalar la identificación se tiene $b^{\circ} = 0$. Las matriz de información y el vector de parámetros estimados será:

$$A = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \quad ; \quad \Theta^{\circ} = a^{\circ}$$

evaluando las expresiones para hallar $\hat{\Theta}$ se tiene:

$$A^T A = \sum_{k=0}^{k} 1 = k+1 \quad ; \quad A^T Y = \sum_{k=0}^{k} y(k)$$

entonces sustituyendo en la ecuación (2.50) se tiene:

$$\hat{\Theta}(k) = (k+1)^{-1} \sum_{k=0}^{k} y(k) = \frac{\sum_{k=0}^{k} y(k)}{k+1}$$

Ahora si se aplica la ecuación (2.54) referente a la propiedad de consistencia se tiene:

$$\lim_{k\to\infty} \operatorname{tr} \sigma^2 \frac{1}{k+1} = \lim_{k\to\infty} \frac{\sigma^2}{k+1} = 0$$

de donde se observa que la estimación es consistente y además que el valor estimado es igual al valor medio de las mediciones.

2.2.2.3.- Mejor estimador lineal sin desviación (BLUE)

Así como se ha demostrado que el método de Mínimos Cuadrados Recursivos es consistente y no desviado se puede demostrar que este, es el mejor estimador lineal no desviado porque es el que minimiza la función de costo. Esto es $J(\hat{\Theta})$ es menor que en cualquier otro estimador $J(\Theta)$ es decir:

$$J(\hat{\Theta}) \leq J(\Theta)$$

La demostración se basa en considerar que el estimador es una función lineal de los datos de la forma $\hat{\Theta} = LY$ y en aplicar multiplicadores de Lagrange para la minimización temática que corresponde a Control Óptimo que escapa al alcance de esta tesis, para mayor detalle puede considerarse las referencias [13],[32].

2.2.2.4 Conclusión MCS

Luego de que se desarrollo en el algoritmo de MCS se concluye que el método de mínimos cuadrados recursivos es valido o sea, consistente, no desviado y el mejor estimador lineal aún en presencia de ruido blanco al cual se le puede aproximar el ruido presente en procesos industriales.

Cuando el ruido no es blanco (correlacionado o coloreado) el algoritmo de MCR presenta un offset o desviación. En el caso en que el ruido es correlacionado se tiene un comportamiento distinto, esto se analiza con el siguiente ejemplo:

Se supone un modelo de primer orden, tal que:

$$y(k) = a^{\circ}y(k-1) + v(k) + c_1 v(k-1) \qquad (2.56)$$

para el ruido definido en la ecuación (2.56) se tiene las siguientes características:

$$\begin{aligned} &Ev(k)=0 \\ &Ev(k)v(j) = \sigma^2 \quad k=j \\ &\qquad\quad = 0 \quad k\neq j \end{aligned} \qquad (2.57)$$

En general no se conoce el valor de la constante c_1 así como el valor de la desviación estándar σ y lo que se desea es calcular el valor de los parámetros en la ecuación (2.56).

Formando la matriz de información, vector de salida y vector de parámetros para k datos se tiene:

$$X = \begin{pmatrix} x(0) \\ x(2) \\ \vdots \\ x(k-1) \end{pmatrix} = \begin{pmatrix} y(0) \\ y(1) \\ \vdots \\ y(k-1) \end{pmatrix}$$

$$Y = \begin{pmatrix} y(1) \\ y(2) \\ \vdots \\ y(k) \end{pmatrix} \quad ; \quad \Theta = a \tag{2.58}$$

la expresión para encontrar el vector de parámetros estimados mediante el método de MCR es:

$$(X^T X)\Theta = XY \tag{2.59}$$

desarrollando los términos de la ecuación (2.59) se tiene:

$$X^T X = \sum_{k=1}^{k} y(k-1)y(k-1) = \sum_{k=1}^{k} y^2(k-1)$$

$$X^T Y = \sum_{k=1}^{k} y(k-1)y(k) \tag{2.60}$$

reemplazando las igualdades de (2.60) en la ecuación (2.59) se tiene:

$$\left[\sum_{k=1}^{k} y^2(k-1)\right]\hat{\Theta}(k) = \sum_{k=1}^{k} y(k-1)y(k) \qquad (2.61)$$

Para encontrar los parámetros $\hat{\Theta}$ es necesario resolver la ecuación (2.61), si se toma en cuenta el ruido en el sistema y la función de autocorrelación para $y(k)$ se tiene:

$$R_y(j) = Ey(k)y(k+j) \qquad (2.62)$$

donde:

$$R_y(j) = \lim_{k\to\infty} \frac{1}{k} \sum_{k=1}^{k} y(k)y(k+j) \qquad (2.63)$$

Entonces para un conjunto de datos suficientemente grande en la ecuación (2.61) se tiene:

$$\lim_{k\to\infty} \left(\frac{1}{k} \sum_{k=1}^{k} y^2(k-1)\right) \Theta = \lim_{k\to\infty} \frac{1}{k} \sum_{k=1}^{k} y(k-1)y(k) \qquad (2.64)$$

en la ecuación (2.64) al aplicar los criterios de la ecuación (2.63) se tiene:

$$R_y(0)\Theta = R_y(1)$$

$$\Theta = \frac{R_y(1)}{R_y(0)} \qquad (2.65)$$

retornando a la ecuación (2.56) de la cual se supone se genera $y(k)$ para obtener $R_y(1)$ y además $R_y(0)$ se tiene que multiplicando (2.56) por $y(k-1)$ y si se toma el valor esperado:

$$Ey(k-1)y(k) = Ea^{\circ}y(k-1)y(k-1) + Ev(k-1)y(k-1) + c_1Ev(k-1)y(k-1) \tag{2.66}$$

aplicando las ecuación (2.62) y el hecho de que $Ev(k)y(k-1) = 0$ debido a que $y(k-1)$ es independiente del ruido blanco $v(k)$ se tiene:

$$R_y(1) = a^{\circ}R_y(0) + Ev(k-1)y(k-1) \tag{2.67}$$

donde para tener en función de la autocorrelación es necesario calcular el término $Ev(k-1)y(k-1)$ para esto en la ecuación (2.56) se reemplaza k por $k-1$ se tiene:

$$y(k-1) = a^{\circ}y(k-2) + v(k-1) + c_1v(k-2) \tag{2.68}$$

multiplicando la ecuación (2.68) por $v(k-1)$ y tomando el valor esperado:

$$Ey(k-1)v(k-1) = Ea^{\circ}y(k-2)v(k-1) + Ev^2(k-1) + cEv(k-2)v(k-1)$$
$$= \sigma^2 \tag{2.69}$$

entonces al reemplazar la ecuación (2.69) en (2.67) se tiene:

$$R_y(1) = a^{\circ}R_y(0) + c_1\sigma^2 \tag{2.70}$$

reemplazando (2.70) en (2.65) se tiene:

$$\hat{\Theta}(\infty) = \frac{R_y(1)}{R_y(0)}$$

$$= a^\circ + c_1 \frac{\sigma^2}{R_y(0)}$$

(2.71)

De la ecuación (2.71) se observa que si $c_1 = 0$ los valores de $\hat{\Theta}(\infty) = a^\circ$, es decir el método de mínimos cuadrados recursivo es asintoticamente no desviado, sin embargo si $c_1 \neq 0$ se ve que el método de MCR es desviado. Por lo tanto cuando existe ruido correlacionado el método de mínimos cuadrados recursivos da un estimador lineal desviado. En consecuencia es necesario buscar variantes del MCR.

El algoritmo de MCR y sus variantes se conocen como métodos de predicción porque utilizan el error de predicción:

$$\varepsilon(k) = y(k) - x(k)\Theta(k-1)$$

o el residual:

$$\eta(k) = y(k) - x(k)\Theta(k)$$

y se conocen también como métodos del gradiente por la razón que se analiza más adelante.

2.3 VARIANTES DEL ALGORITMO DE MÍNIMOS CUADRADOS [17], [28]

Si se tiene ruido correlacionado, entonces MCR ó MCS produce un offset en la identificación de parámetros. Se tienen varios métodos para resolver esta situación:

- Variable Instrumental (IV).
- Mínimos cuadrados Generalizados (MCG).
- Mínimos cuadrados Extendidos (MCE).
- Máximo de Likelihood (Máximo de verosimilitud MLH).

Todos estos métodos utilizan el error de predicción para obtener el modelo de regresión, por ello se llaman métodos de predicción. Existen dos clases de errores:

- $\varepsilon(k) = y(k) - x(k)\Theta(k-1)$ error de predicción.
- $\eta(k) = y(k) - x(k)\Theta(k)$ residual.

El método de mínimos cuadrados ordinarios, recursivos, ponderados y estocásticos han sido analizados anteriormente a continuación se hace un análisis breve de los métodos restantes, se añade a este análisis el método del gradiente por la utilización del gradiente en el algoritmo de máximo de Likelihood.

2.3.1 MÉTODO DEL GRADIENTE

Existe una variante muy sencilla para el cálculo del modelo ARMA, en base al gradiente; que se conoce como MÉTODO DEL

GRADIENTE. En efecto, se tiene que; expresando en series de Taylor la función de costo:

$$J(\hat{\Theta}+h) = J(\hat{\Theta}) + hJ'(\hat{\Theta}) + \frac{h^2}{2}J'(\hat{\Theta}) + \ldots\ldots \qquad (2.72)$$

donde $\hat{\Theta}$ da la mínima función de costo.

Si en la ecuación (2.72) se tiene pequeñas variaciones de h $J(\hat{\Theta}+h)$ puede aproximarse a:

$$J(\hat{\Theta}+h) = J(\hat{\Theta}) + hJ'(\hat{\Theta}) \qquad (2.73)$$

donde $J'(\hat{\Theta})$ es el gradiente con respecto a $\hat{\Theta}$ es decir se tiene $J'(\hat{\Theta}) = \frac{dJ}{d\Theta}$.

Por lo que el nuevo valor es igual al anterior más una corrección, en donde h es la ganancia y $J'(\hat{\Theta})$ es el gradiente.

Se tiene que:

$$\hat{\Theta}(k) = \hat{\Theta}(k-1) + L(k)\varepsilon(k) \qquad (2.74)$$

Por analogía con el método del gradiente para el cálculo del mínimo se tiene que $e(k)$ proviene de un gradiente ε^2. Para corregir el factor cambiante se tiene.

$$\hat{\Theta}(k) = \hat{\Theta}(k-1) + \frac{L(k)}{2} \operatorname{grad} \varepsilon^2(k) \qquad (2.75)$$

Si el error esta definido como:

$$\varepsilon(k) = y(k) - \hat{y}(k)$$

reemplazando el valor de $\hat{y}(k) = x(k)\Theta(k-1)$ en la ecuación (2.75) se tiene:

$$\varepsilon(k) = y(k) - x(k)\Theta(k-1) \qquad (2.76)$$

derivando el cuadrado del error se tiene:

$$\frac{d}{d\Theta}\varepsilon^2(k) = \frac{d}{d\Theta}\left(y(k) - x(k)\Theta(k-1)\right)^2$$

$$= -2x^T(k)\left(y(k) - x(k)\Theta(k-1)\right) \qquad (2.77)$$

Al reemplazar (2.77) en la ecuación (2.75) se tiene:

$$\hat{\Theta}(k) = \hat{\Theta}(k-1) - L(k)x^T(k)\varepsilon(k) \qquad (2.78)$$

A medida que $\hat{\Theta}(k) \rightarrow \Theta^{\circ}(k)$, la ganancia va disminuyendo pues la corrección es menor. Al analizar $L(k)$ esta es proporcional a $P(k)$ y al disminuir $P(k)$ también disminuye $L(k)$.

Una variación muy sencilla de este método del gradiente es la siguiente, si se tiene:

$$\hat{\Theta}(k) = \hat{\Theta}(k-1) - L(k)x^{T}(k)\varepsilon(k) \qquad (2.79)$$

se toma $L(k) = \frac{1}{k}$ con lo cual se hace un algoritmo muy sencillo y rápido.

2.3.2 MÉTODO DE VARIABLE EXPERIMENTAL (INSTRUMENTAL VARIABLE IV) [10], [23], [24], [26]

El método de variable instrumental es un mecanismo de identificación de parámetros de un proceso en el cual se tiene presente ruido correlacionado.

La variable instrumental es una señal que esta relacionada con variables útiles del proceso y no esta correlacionada con el ruido, si se considera un matriz Z cuyos elementos son funciones de los datos (variable instrumental). Para un diagrama de un sistema con ruido correlacionado que se muestra en la figura 2.6.

Se tiene que $y_s(k)$ no esta contaminada por ruido por lo que la matriz $Z(k)$ esta definida como:

$$Z(k) = \begin{pmatrix} z(1) \\ z(2) \\ \vdots \\ z(k) \end{pmatrix}$$

donde:

$$z(k) = \left(y_e(k-1) \quad \cdots \quad y_e(k-n) \quad u(k-r) \quad \cdots \quad u(k-m)\right)$$

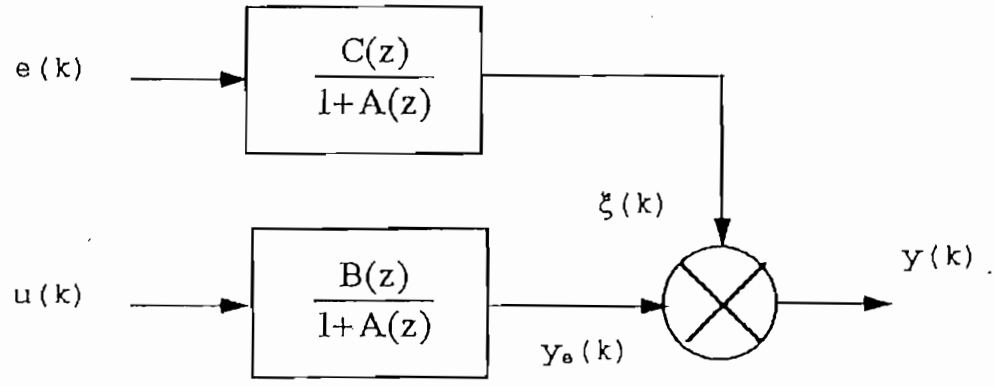

Fig. 2.6 Modelo ARMAX

Si se tiene la ecuación de diferencias de un sistema:

$$Y(k) = X(k)\Theta(k) + E(k) \tag{2.80}$$

y se considera un matriz Z cuyos elementos son funciones de los datos (variable instrumental), se multiplica su transpuesta Z^T por la ecuación (2.80) se tiene:

$$Z^T(k)Y(k) = Z^T(k)X(k)\Theta(k) + Z^T(k)E(k) \tag{2.81}$$

Para identificar los parámetros $\Theta(k)$ sin desviación se debe minimizar la función de costo que para este caso es:

$$J(k)=\left(Z^T(k)E(k)\right)^T Z^T(k)E(k) \tag{2.82}$$

reemplazando $Z^T(k)E(k) = Z^T(k)Y(k)-Z^T(k)X(k)\Theta(k)$ en la ecuación (2.82) se tiene:

$$J(k)=\left(Z^T(k)Y(k)-Z^T(k)X(k)\Theta(k)\right)^T\left(Z^T(k)Y(k)-Z^T(k)X(k)\Theta(k)\right) \quad (2.83)$$

Minimizando la ecuación (2.83) para identificar los parámetros $\Theta(k)$, se tiene un desarrollo similar al método de mínimos cuadrados recursivos en donde se llega luego de derivar con respecto a $\Theta(k)$ a:

$$X^T(k)Z(k)Z^T(k)Y(k) - X^T(k)Z(k)Z^T(k)X(k)\hat{\Theta}(k) = 0 \quad (2.84)$$

en la ecuación (2.84) se tiene como factor común $X^T(k)Z(k)$ entonces:

$$X^T(k)Z(k)\left(Z^T(k)Y(k) - Z^T(k)X(k)\hat{\Theta}(k)\right) = 0 \quad (2.85)$$

de donde el término $X^T(k)Z(k)$ es distinto de cero por lo que:

$$Z^T(k)Y(k) - Z^T(k)X(k)\hat{\Theta}(k) = 0 \quad (2.86)$$

en la ecuación (2.86) se puede despejar $\hat{\Theta}(k)$ si existe la inversa de $Z^T(k)X(k)$ con lo que se tiene que el vector de parámetros identificados es:

$$\hat{\Theta}(k) = \left(Z^T(k)X(k)\right)^{-1} Z^T(k)Y(k) \quad (2.87)$$

la ecuación (2.87) da lugar a un estimador no desviado.

La matriz de variable instrumental debe satisfacer las condiciones:

$$E\left(Z^T(k)E(k)\right)^{-1} = 0$$

$$\det E\left(Z^T(k)E(k)\right) \neq 0$$

por cuanto, para cuando $\hat{\Theta} \rightarrow \Theta^\circ$ el error solo depende del ruido y Z no esta correlacionada con él. Además Z^TX debe ser invertible.

Bajo estas condiciones se conoce como matriz instrumental y puede ser aplicada por ejemplo en sistemas retroalimentados en donde la entrada de comando no esta correlacionada con ruido pueden ser tomados como variable instrumental. Se puede satisfacer la condición de variable instrumental por ejemplo removiendo o filtrando el ruido a la señal de salida y.

Este método no se utiliza en el presente trabajo por no disponer de variables instrumentales, a partir de sistemas retroalimentados o filtrados.

2.3.3 MÍNIMOS CUADRADOS GENERALIZADOS (MCG) [5], [8], [9], [15, [16], [24].

Este método basa la identificación de parámetros en el modelo ARMAX:

$$(1+A(z))y(k) = z^{-r}B(z)u(k) + \xi(k) \qquad (2.88)$$

Una diagrama que muestra el modelo de identificación se tiene en la figura 2.7.

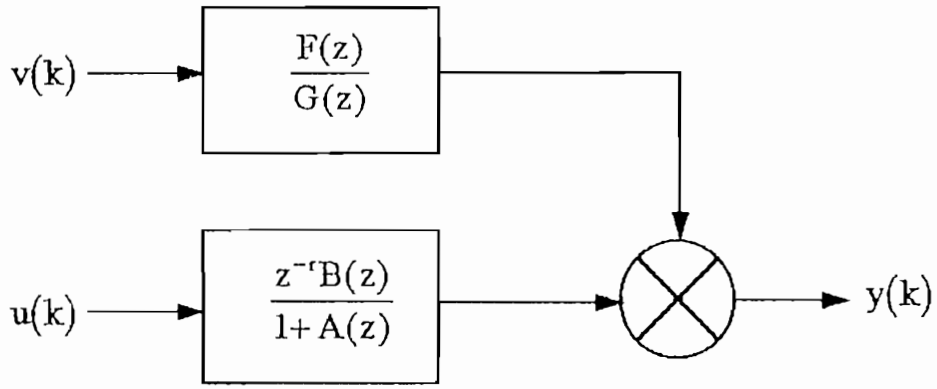

Fig. 2.7 Modelo ARMAX

Cualquier proceso estocástico (ruido coloreado o correlacionado) $\xi(k)$ puede obtenerse haciendo pasar ruido blanco $v(k)$ por el filtro $\frac{F(z)}{G(z)}$. Para el caso de mínimos cuadrados generalizados se tiene que $F(z) = 1$ y $G(z) = 1+C(z)$.

Por lo que el diagrama del modelo ARMAX con el filtro para obtener ruido coloreado se representa en la figura 2.8.

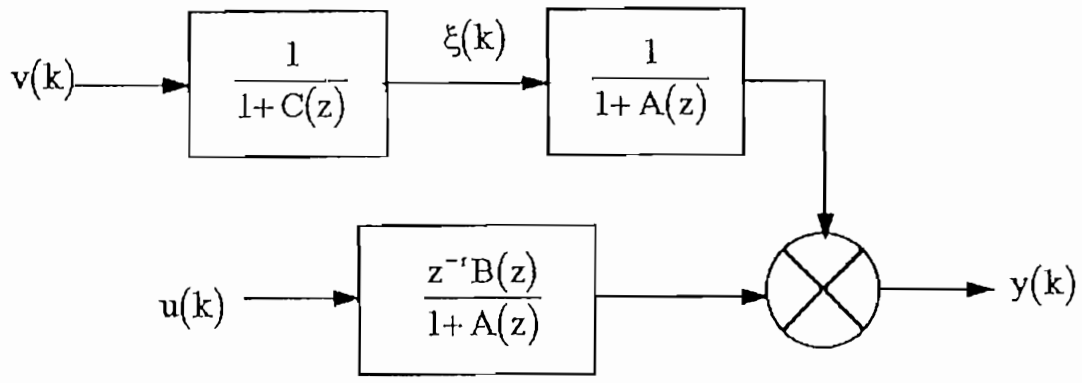

Fig. 2.8 Modelo de ruido autorregresivo

entonces del modelo se tiene:

$$y(k) = \frac{z^{-r}B(z)}{1+A(z)}u(k) + \frac{1}{(1+C(z))(1+A(z))}v(k)$$

$$(1+A(z))y(k) = z^{-r}B(z)u(k) + \frac{v(k)}{1+C(z)} \qquad (2.89)$$

De donde $\xi(k) = \frac{v(k)}{1+C(z)}$ y $(c_0=1)$, por lo que $v(k) = (1+C(z))\xi(k)$, es decir se utiliza un modelo autoregresivo. Multiplicando la ecuación (2.89) por $(1+C(z))$ se obtiene:

$$(1+A(z))(1+C(z))y(k) = z^{-r}B(z)(1+C(z))u(k) + v(k) \qquad (2.90)$$

la ecuación (2.90) se puede agrupar como:

$$(1+A(z))y_e(k) = z^{-r}B(z)u_e(k) + v(k) \qquad (2.91)$$

donde:

$$y_c(k) = (1+C(z))y(k)$$

$$u_c(k) = (1+C(z))u(k)$$

Por lo tanto la ecuación (2.91) es un problema de Mínimos Cuadrados Recursivos pero en las variables $y_e(k)$, $u_e(k)$ en donde se puede obtener los parámetros de A y B, sin embargo se mide y, u con lo que es necesario identificar $C(z)$ para obtener $y_e(k)$, $u_e(k)$, que es otro caso de identificación de MCR, es decir se filtra la información actual a través de $C(z)$. continuación se examina una variante de este método que es más sencillo de implementar computacionalmente, llamado mínimos cuadrados extendidos.

2.3.4 MÍNIMOS CUADRADOS EXTENDIDOS (MCE) [5], [8], [9], [15], [16], [24].

Una variante del método MCG es asumir el modelo de la figura 2.9, pero con un proceso de ruido de media móvil.

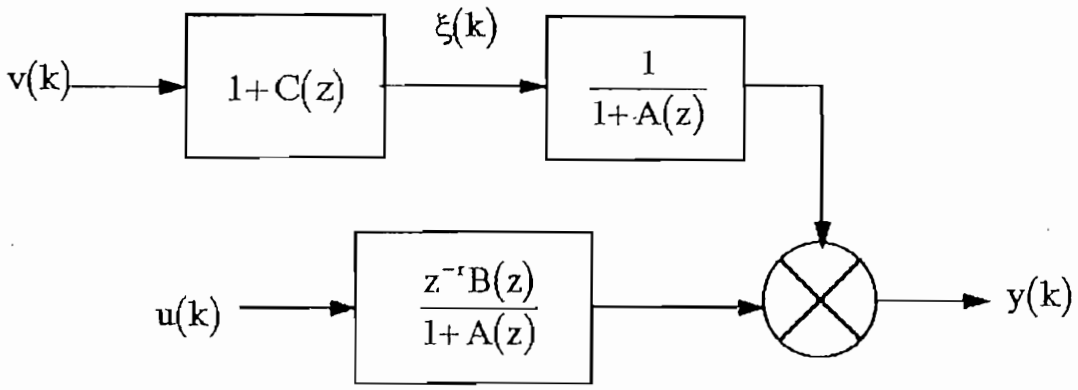

Fig. 2.9 Modelo ARMAX

El ruido de media móvil esta dado por:

$$\xi(k) = (1+C(z))v(k) \tag{2.92}$$

por lo que si se reemplaza (2.92) en el modelo general de identificación se tiene:

$$(1+A(z))y(k) = z^{-r}B(z)u(k) + (1+C(z))v(k)$$

$$y(k) = \frac{z^{-r}B(z)}{1+A(z)}u(k) + \frac{1+C(z)}{1+A(z)}e(k) \tag{2.93}$$

entonces lo que se hace es extender el vector de parámetros a los coeficientes $C(z)$ y extender el vector de información al ruido blanco $v(k)$, y se tiene:

$$y(k) = x(k)\Theta(k-1)$$

donde se define el nuevo vector de información como:

$$x(k) = \left(-y(k-1) \quad \cdots \quad -y(k-n) \quad u(k-r) \quad \cdots \quad u(k-m) \quad v(k-1) \quad \cdots \quad v(k-p)\right) \tag{2.94}$$

$$\Theta(k) = \left(a_1 \quad \cdots \quad a_n \quad b_r \quad \cdots \quad b_m \quad c_1 \quad \cdots \quad c_p\right) \tag{2.95}$$

sin embargo como no se conoce el ruido blanco $v(k)$ en el proceso se aproxima por el error de predicción que se lo puede calcular como:

$$\varepsilon(k) = y(k) - x(k)\Theta(k-1) \tag{2.96}$$

Al desarrollar la ecuación (2.96) se tiene que:

$$\varepsilon(k) = y(k) + A(z)y(k) - z^{-r}B(z)u(k) - C(z)\varepsilon(k) \qquad (2.97)$$

de la ecuación (2.97) el error de predicción puede ser expresado como:

$$\varepsilon(k) = \frac{1}{1+C(z)}\left((1+A(z))y(k) - z^{-r}B(z)u(k)\right) \qquad (2.98)$$

en la ecuación (2.98) se observa que la mediciones de entrada y salida son filtradas a través del filtro $1+C(z)$. Entonces a medida que $A(z)\rightarrow A^{\circ}(z)$, $B(z)\rightarrow B^{\circ}(z)$, $C(z)\rightarrow C^{\circ}(z)$, se tiene que $\varepsilon(k)\rightarrow v(k)$.

2.3.5 MÁXIMO DE LIKELIHOOD (MÁXIMO DE VEROSIMILITUD MLH) [2], [3], [10], [11], [13], [14], [16], [20], [24].

2.3.5.1 Introducción

El Máximo de Likelihood es un método estadístico (probabilístico) pues se manejan variables aleatorias en forma de proceso estocástico.

Para analizar procesos estadísticos es necesario tener modelos estocásticos, sin embargo el modelo ARMA planteado anteriormente es determinístico, es decir este método es útil para identificar procesos determinísticos o inclusive procesos que tengan ruido blanco, como se demostró en el desarrollo del método de MCR. Cuando se tiene la presencia de ruido correlacionado es necesario la utilización del modelo ARMAX, para la correcta identificación de los parámetros del modelo.

Cuando se tiene procesos estocásticos una buena alternativa es la utilización de un método estadístico, para encontrar parámetros y de esta manera garantizar resultados satisfactorios a costo de aumentar la complejidad y el tiempo de computación en los algoritmos.

2.3.5.2 Función de Likelihood

Si se considera un proceso estocástico gaussiano x, que tiene una distribución de Gauss, en donde la función densidad de probabilidad esta dada por la ecuación:

$$f_x(\xi) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2}\frac{(\xi-\mu)^2}{\sigma^2}} \qquad (2.99)$$

en donde σ^2 (varianza), μ (valor medio), describen completamente a f_x para un proceso escalar.

Si se tiene n variables aleatorias $x_1, x_2 \ldots\ldots x_n$ la función de probabilidad es vectorial, donde μ es un vector de valores medios.

$$E(x) = \mu$$

$$E\left((x-\mu)(x-\mu)^T\right) = R \qquad (2.100)$$

Se define a R como matriz de Covarianzas.

En la función densidad de probabilidad reemplazando las definiciones de (2.99) se tiene la función densidad de probabilidad vectorial:

$$f_x(\xi) = \frac{1}{\sqrt{(2\pi)^n \det R}} e^{-\frac{1}{2}(\xi-\mu)R^{-1}(\xi-\mu)^T} \qquad (2.101)$$

si las variables $x_1, x_2 \ldots\ldots x_n$ son mutuamente excluyentes no correlacionadas y tienen idénticos valores medios y de varianza que en la realidad se cumple (pues corresponde al mismo proceso de ruido).

Al tomar n observaciones o mediciones contaminadas por ruido, independientes la una de la otra, se tiene que la función de probabilidad total es:

$$f = f_1 * f_2 \ldots\ldots f_n$$

reemplazando el valor de cada función densidad de probabilidad multiplicando y agrupando se tiene:

$$f_x(\xi) = \frac{1}{(2\pi\sigma)^{n/2}} e^{-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(\xi_i-\mu)^2} \qquad (2.102)$$

donde: $R = \sigma^2 I$.

Como el modelo ARMAX es estocástico se debe utilizar la función densidad de probabilidad que se ha descrito anteriormente, para la identificación de parámetros.

Para la identificación de procesos por el método de verosimilitud se toma la función de densidad de probabilidad porque es un proceso estocástico, pero se hace que esta función de densidad de probabilidad dependa de los parámetros. Entonces:

$$L(y,\Theta) = f(y,\Theta) \qquad (2.103)$$

donde L es la función densidad de probabilidad o función de Likelihood.

Se utiliza el cambio de notación para expresar el hecho de que lo que se mide es y a la salida; Θ son los parámetros desconocidos del modelo y L la función densidad de probabilidad se hace depender Θ para que en el algoritmo de identificación se maximize la función densidad de probabilidad y lo mejor que se puede hacer es obtener los

parámetros con el máximo de probabilidad para el proceso estadístico.

Para construir la función de Likelihood se utilizan los errores de las mediciones, al utilizar los errores de las mediciones se pone la función de Likelihood en función de Θ.

Sea el modelo que se esta identificando

$$Y = X\Theta^{\circ} + \xi \tag{2.104}$$

en donde el error esta dado por el ruido y la exactitud de los parámetros Θ. Para los verdaderos valores de Θ° se tiene que el ruido ξ se puede escribir como:

$$\xi = Y - X\Theta^{\circ} \tag{2.105}$$

por lo que la función de probabilidad se puede escribir como $f(y,\Theta^{\circ})$ que significa densidad de probabilidad del proceso estocástico dado Θ° así:

$$f(y,\Theta^{\circ}) = \frac{1}{(2\pi\sigma)^{m/2}}\, e^{-\frac{1}{2\sigma^2}\left\{(Y - X\Theta^{\circ})^T (Y - X\Theta^{\circ})\right\}} \tag{2.106}$$

donde: Y es el vector de observaciones.
X es la matriz de información.
Θ° vector de parámetros.
m es el número de mediciones.

El sumatorio de la exponencial en la ecuación (2.102) se ha sustituido por el producto de $\xi^T \cdot \xi$, y se considera además ruido presente en el sistema con un valor de media cero.

Se genera la función de Likelihood sustituyendo el valor de Θ° por el de Θ y entonces se tiene la función L como:

$$L(y|\Theta) = \frac{1}{(2\pi\sigma)^{m/2}} e^{-\frac{1}{2\sigma^2}\left\{(Y - X\Theta)^T(Y - X\Theta)\right\}} \qquad (2.107)$$

se define la función de costo J como menos el logaritmo natural de L para obtener una función no negativa.

$$J = -\ln(L) \qquad (2.108)$$

se toma J de esta forma para tener una función más simple de optimizar entonces:

$$-\ln(L(y|\Theta)) = \ln(2\pi\sigma^2)^{m/2} + \frac{1}{2\sigma^2}\left((Y - X\Theta)^T(Y - X\Theta)\right) \qquad (2.109)$$

Para maximizar la función de Likelihood es necesario minimizar la ecuación (2.109), es decir se debe encontrar los valores de Θ y σ^2 que hagan que se tenga el menor valor de $-\ln(L(y|\Theta))$. Sin embargo para el tema de la presente tesis solamente se toma en cuenta los valores de Θ que hacen que la función de Likelihood sea máxima ya que se esta buscando los parámetros que describan de mejor manera a la planta.

Entonces derivando la función de costo J con respecto a Θ, para minimizar la ecuación (2.109) se tiene:

$$\frac{\partial J}{\partial \Theta} = \frac{\partial}{\partial \Theta}(-\ln L)$$

$$\frac{\partial}{\partial \Theta}\ln\left(2\pi\sigma^2\right)^{m/2} + \frac{\partial}{\partial \Theta}\frac{1}{2\sigma^2}\left((Y-X\Theta)^T(Y-X\Theta)\right) = 0$$

$$\frac{1}{2\sigma^2}\frac{\partial}{\partial \Theta}\left((Y-X\Theta)^T(Y-X\Theta)\right) = 0 \qquad (2.110)$$

resolver la ecuación (2.110) es igual que cuando se deriva la función de costo cuadrática y el resultado es similar al encontrado al desarrollar el método de mínimos cuadrados determinísticos, y se tiene:

$$\frac{1}{2\sigma^2}\left(X^TX\Theta - X^TY\right) = 0 \qquad (2.111)$$

Con esto se concluye que el Método de Likelihood o de Probabilidad Máxima es en esencia Mínimos Cuadrados pero desde un punto de vista probabilístico, en el que se utiliza la función densidad de probabilidad.

2.3.5.3 Algoritmo de Probabilidad Máxima

Se tiene el modelo ARMAX, que se utiliza para desarrollar el algoritmo de Probabilidad Máxima.

$$y(k) + \sum_{i=1}^{n} a_i y(k-i) = \sum_{i=1}^{m} b_1 u(k-i) + v(k) + \sum_{i=1}^{p} c_i v(k-i) \qquad (2.112)$$

lo que se trata es de encontrar un algoritmo que permita encontrar los parámetros del modelo en función de los valores anteriores de $y(k)$ y $u(k)$.

Al instante k, $y(k-i)$, $u(k-i)$ son conocidas pues corresponden a los valores (mediciones) anteriores de u, y entonces una buena aproximación de $v(k-i)$ puede calcularse a partir de u, y.

$$v_e(k) = y(k) - \sum_{i=1}^{n} a_i y(k-i) + \sum_{i=1}^{m} b_1 u(k-i) + \sum_{i=1}^{p} c_i v_e(k-i) \qquad (2.113)$$

Para valores de $k = 1,\ldots,k-1$.

El valor de $v(k)$ no se conoce y no se puede predecir de los valores anteriores pues es independiente de lo sucedido hasta el instante $k-1$, entonces la predicción natural de $\hat{y}(k|\Theta(k))$ al instante k, en base a los valores anteriores es:

$$\hat{y}(k|\Theta(k)) = -\sum_{i=1}^{n} a_i y(k-i) + \sum_{i=1}^{m} b_1 u(k-i) + \sum_{i=1}^{p} c_i v_e(k-i) \qquad (2.114)$$

La expresión de la ecuación (2.114) requiere de condiciones iniciales sin embargo el efecto de estas decae con el factor

$|c|^{k-1}$, por lo que si $|c|<1$ se pueden escoger arbitrariamente, sin que estas afecten mucho a la predicción.

A continuación se procede a tomar en cuenta el efecto de tener coeficientes c, en la ecuación (2.104) por lo que se define:

$$D = \sum_{i=1}^{p} c_i \tag{2.115}$$

multiplicando la ecuación (2.115) por la ecuación (2.114) en instantes anteriores de muestreo se tiene:

$$\sum_{i=1}^{p} c_i \hat{y}\left(k-i|\Theta(k-i)\right) = -D\sum_{i=2}^{n} a_{i-1} y(k-i) + D\sum_{i=2}^{m} b_{i-1} u(k-i) + D\sum_{i=2}^{p} c_{i-1} v_e(k-i) \tag{2.116}$$

Sumando miembro a miembro la ecuación (2.116) con (2.114) se llega a :

$$\begin{aligned}\hat{y}\left(k|\Theta(k)\right) + \sum_{i=1}^{p} c_i \hat{y}\left(k-i|\Theta(k-i)\right) = &-\sum_{i=1}^{n} a_i y(k-i) - D\sum_{i=2}^{n} a_{i-1} y(k-i) + \\ &+ \sum_{i=1}^{m} b_i u(k-i) + D\sum_{i=2}^{m} b_{i-1} u(k-i) + \\ &+ \sum_{i=1}^{p} c_i v_e(k-i) + D\sum_{i=2}^{p} c_{i-1} v_e(k-i)\end{aligned} \tag{2.117}$$

Pero se debe notar que:

$$v_e(k-i) = y(k-j) - \sum_{i=1}^{n} a_i y(k-j-i) + \sum_{i=1}^{m} b_1 u(k-j-i) + \sum_{i=1}^{p} c_i v_e(k-j-i)$$

$$\text{Para } j = 1,\ldots,k-1 \tag{2.118}$$

Reemplazando (2.118) en (2.117).

$$\hat{y}(k|\Theta(k)) + \sum_{i=1}^{p} c_i \hat{y}(k-i|\Theta(k-i)) = -\sum_{i=1}^{n} a_i y(k-i) - D\sum_{i=2}^{n} a_{i-1} y(k-i) +$$
$$+ \sum_{i=1}^{m} b_i u(k-i) + D\sum_{i=2}^{m} b_{i-1} u(k-i) +$$
$$+ Dy(k-j) + D\sum_{i=1}^{n} a_i y(k-j-i) +$$
$$- D\sum_{i=1}^{m} b_i u(k-j-i) - D^2 v_e(k-j-i) +$$
$$+ D\sum_{i=1}^{p} c_{i-1} v_e(k-i)$$

Para $j = 1,....,k-1$

$$\hat{y}(k|\Theta(k)) + \sum_{i=1}^{p} c_i \hat{y}(k-i|\Theta(k-i)) = \sum_{i=1}^{l} (c_i - a_i) y(k-i) + \sum_{i=1}^{m} b_i u(k-i)$$

definiendo el valor de YEST como:

$$YEST = \hat{y}(k|\Theta(k)) + \sum_{i=1}^{p} c_i \hat{y}(k-1|\Theta(k-1))$$

entonces:

$$YEST = \hat{y}(k|\Theta(k)) + \sum_{i=1}^{p} c_i \hat{y}(k-1|\Theta(k-1)) = \sum_{i=1}^{n} (c_i - a_i) y(k-i) + \sum_{i=1}^{m} b_i u(k-i) \qquad (2.119)$$

Para minimizar la función de Likelihood se debe obtener el error de predicción $\varepsilon(k) = y(k) - \hat{y}(k,\Theta)$ entonces:

$$\varepsilon(k) = y(k) - \sum_{i=1}^{n} a_i y(k-i) - \sum_{i=1}^{m} b_i u(k-i) + \sum_{i=1}^{n} (c_i - a_i) y(k-i)$$

Esta ecuación no es lineal para los parámetros Θ, por cuanto existen productos de los parámetros en el último término, por consiguiente no se puede realizar el algoritmo de minimización en forma analítica, y en consecuencia se debe plantear un método numérico alternativo aproximado.

2.3.5.4 Método Numérico para el cálculo de Máximo de Likelihood [2],[3],[14],[13],[10],[11],[24].

Se puede apreciar de la función de costo J que:

$$J = -\ln\left(L(y|\Theta)\right) = \ln\left(2\pi\sigma^2\right)^{m/2} + \frac{1}{2\sigma^2}\left((Y-X\Theta)^T(Y-X\Theta)\right) \quad (2.120)$$

existen 2 partes: una que depende de σ y la otra que depende de Θ, en este caso el interés esta centrado en la dependencia con respecto al vector de parámetros Θ, entonces al minimizar J respecto a Θ se tiene $\frac{\partial J}{\partial \Theta} = 0$, el primer término no se toma en cuenta y lo que se debe considerar son solamente los errores debido a Θ que están dados por $(Y-X\Theta)^T(Y-X\Theta)$ que en definitiva es la suma de los errores de predicción al cuadrado, esto es; se define una nueva función de costo dada por:

$$\begin{aligned} V_k(\Theta) &= \frac{1}{2\sigma^2}\left((Y-X\Theta)^T(Y-X\Theta)\right) \\ &= \frac{1}{2\sigma^2}\sum_{k=1}^{K}\varepsilon^2(k,\Theta) \end{aligned} \quad (2.121)$$

donde $\varepsilon(k,\Theta(k)) = y(k) - \hat{y}(k,\Theta(k-1))$ es el error de predicción.

La ecuación (2.121) es en última instancia Mínimos Cuadrados, de ahí que este método sea considerado como una variante en el cual habrá que introducir los coeficientes c, de manera que se remueva el offset que se produce al tener ruido correlacionado. Para desarrollar un método numérico se expande $V_k(\Theta)$ en series de Taylor alrededor del punto $\Theta(k-1)$ se tiene entonces:

$$V_k[\Theta(k)] = V_k[\Theta(k-1)] + V_k'[\Theta(k-1)][\Theta(k)-\Theta(k-1)] + \\ + \frac{1}{2}[\Theta(k)-\Theta(k-1)]^T V_k''[\Theta(k-1)][\Theta(k)-\Theta(k-1)] + \ldots\ldots \quad (2.122)$$

Al llevar a cabo la expansión cerca del mínimo, para pequeñas variaciones, cuando $\hat{\Theta} \rightarrow \Theta^*$ se pueden despreciar los términos superiores a la segunda potencia. Minimizando la función de costo $V_k[\Theta(k)]$ con respecto a $\Theta(k)$ se tiene:

$$\frac{dV_k[\Theta(k)]}{d\Theta(k)} = 0$$

$$\frac{dV_k[\Theta(k)]}{d\Theta(k)} = V_k'[\Theta(k-1)]^T + \frac{1}{2}V_k''[\Theta(k-1)][\Theta(k)-\Theta(k-1)] + \\ + \frac{1}{2}V_k'[\Theta(k-1)]^T[\Theta(k)-\Theta(k-1)] = 0 \quad (2.123)$$

La matriz de segundas derivadas es simétrica, debido a que es el gradiente de la primera derivada, y se tiene derivadas cruzadas que son iguales simplificando entonces:

$$V_k'\left[\Theta(k-1)\right]^T = -V_k''\left[\Theta(k-1)\right]\left[\Theta(k)-\Theta(k-1)\right] \qquad (2.124)$$

de la ecuación (2.124) se puede obtener el vector de parámetros estimados como:

$$\hat{\Theta}(k) = \hat{\Theta}(k-1) - V_k''\left[\Theta(k-1)\right]^{-1} V_k'\left[\Theta(k-1)\right]^T \qquad (2.125)$$

Se define la función $\Phi(k)$ como el gradiente de los errores entonces:

$$\Phi(k,\Theta) = \left[-\frac{d\varepsilon(k,\Theta)}{d\Theta}\right]^T \qquad (2.126)$$

entonces:

$$V_k\left[\Theta(k)\right] = \frac{1}{2}\sum_{k=1}^{k} e^2(k,\Theta) = \frac{1}{2}E^T E$$

$$V_k'\left[\Theta(k)\right]^T = \frac{dE^T}{d\Theta}E \qquad (2.127)$$

Utilizando la ecuación (2.126) para las derivadas del error se tiene:

$$\begin{aligned} V_k'\left[\Theta(k)\right]^T &= -\sum_{k=1}^{K}\Phi(k,\Theta(k))\varepsilon(k,\Theta(k)) \\ &= -V_{k-1}'\left[\Theta(k-1)\right]^T - \Phi(k,\Theta(k))\varepsilon(k,\Theta(k)) \end{aligned} \qquad (2.128)$$

si se toma la derivada de la ecuación (2.127) entonces:

$$V_k''\left[\Theta(k)\right]^T = \frac{d^2E^T(k)}{d^2\Theta(k)}E(k) + \frac{dE^T(k)}{d\Theta(k)}\cdot\frac{dE(k)}{d\Theta(k)}$$

$$= V_{k-1}''\left[\Theta(k-1)\right]^T + \Phi\left(k,\Theta(k)\right)\Phi^T\left(k,\Theta(k)\right) + \varepsilon''\left(k,\Theta(k)\right)\varepsilon\left(k,\Theta(k)\right) \quad (2.129)$$

Para continuar con el desarrollo del método numérico para la obtención de los parámetros es necesario realizar algunas aproximaciones.

- Puesto que se trabaja en la vecindad de Θ° se tiene que la segunda derivada del error cambia menos que la primera derivada entonces se puede despreciar la segunda derivada del error que es:

$$\varepsilon''\left[k,\Theta(k)\right]$$

y en consecuencia se tiene que el término

$$\varepsilon''\left[k,\Theta(k)\right]\varepsilon\left[k,\Theta(k)\right] \cong 0 \quad (2.130)$$

con lo cual se aproxima la ecuación (2.120) (segunda derivada) a:

$$V_k''\left[\Theta(k)\right]^T = V_{k-1}''\left[\Theta(k-1)\right]^T + \Phi\left(k,\Theta(k)\right)\Phi^T\left(k,\Theta(k)\right) \quad (2.131)$$

- En la vecindad de Θ° la primera derivada de la función de costo se expresa mediante la siguiente ecuación:

$$V_k'\left[\Theta(k)\right]^T = V_{k-1}'\left[\Theta(k-1)\right]^T - \Phi\left(k,\Theta(k)\right)\varepsilon\left(k,\Theta(k)\right) \qquad (2.132)$$

sin embargo como en la vecindad de Θ° se puede considerar que se ha minimizado la función de costo, entonces se cumple que:

$$\frac{dV_{k-1}\left[\Theta(k-1)\right]}{d\Theta} = V_{k-1}'\left[\Theta(k-1)\right] \cong 0 \qquad (2.133)$$

y la variación se debe solamente al gradiente del error que se da porque $\hat{\Theta} \rightarrow \Theta^\circ$.

Para actualizar el valor de $\hat{\Theta}(k)$ se debe calcular $V_k'\left[\hat{\Theta}(k-1)\right]^{-1}$ y $V_k'\left[\hat{\Theta}(k-1)\right]^T$, los mismos que se han simplificado a:

$$V_k''\left[\Theta(k)\right]^T = V_{k-1}''\left[\Theta(k-1)\right]^T + \Phi\left(k,\Theta(k)\right)\Phi^T\left(k,\Theta(k)\right) \qquad (2.134)$$

$$V_k'\left[\Theta(k)\right]^T = -\Phi\left(k,\Theta(k)\right)\varepsilon\left(k,\Theta(k)\right) \qquad (2.135)$$

Sin embargo se tiene que :

$$V_k''\left[\Theta(k)\right]^T \cong V_k''\left[\Theta(k-1)\right] \qquad (2.136)$$

porque la variación en $\hat{\Theta}$ es muy pequeña cuando se esta en la vecindad de Θ° y además según se indico anteriormente la segunda derivada cambia muy poco en la vecindad, por lo que se tiene que la ecuación (2.125) cambia a:

$$\hat{\Theta}(k) = \hat{\Theta}(k-1) - V_k''\left[\hat{\Theta}(k)\right]^{-1} V_k'\left[\hat{\Theta}(k-1)\right]^T \qquad (2.137)$$

por lo que si se sustituye lo que vale la primera derivada ecuación (2.135) en la ecuación (2.137) se tiene:

$$\hat{\Theta}(k) = \hat{\Theta}(k-1) + V_k''\left[\hat{\Theta}(k)\right]^{-1} \Phi(k,\Theta(k))\varepsilon(k,\Theta(k)) \qquad (2.138)$$

$$V_k''\left[\Theta(k)\right] = V_{k-1}''\left[\Theta(k-1)\right] + \Phi(k,\Theta(k))\Phi^T(k,\Theta(k)) \qquad (2.139)$$

Estas 2 ecuaciones (2.138) y (2.139) se dan en forma recursiva. Sin embargo se puede realizar más simplificaciones.

En primer lugar se debe obtener $\varepsilon(k,\Theta(k))$ y su gradiente, de la ecuación (2.126) se tiene:

$$\Phi(k,\Theta) = \left[-\frac{d\varepsilon(k,\Theta)}{d\Theta}\right]^T$$

En efecto se tiene que el error de predicción esta dado por:

$$\varepsilon(k,\Theta(k)) = y(k) - \hat{y}(k,\Theta(k-1))$$

entonces al formar el gradiente se tiene que:

$$\frac{d\varepsilon(k,\Theta(k))}{d\Theta} = -\frac{d\hat{y}(k,\Theta(k))}{d\Theta} \qquad (2.140)$$

Ahora se necesita calcular (2.140), si :

$$\Phi^{T}\left(k,\Theta(k)\right) = \frac{d\hat{y}\left(k,\Theta(k)\right)}{d\Theta} \qquad (2.141)$$

De la ecuación (2.120) se tiene el valor de $\hat{y}\left(k|\Theta(k)\right)$ que es:

$$\hat{y}\left(k|\Theta(k)\right) + \sum_{i=1}^{p} c_i\, \hat{y}\left(k-1|\Theta(k-1)\right) = \sum_{i=1}^{n}\left(c_i - a_i\right) y(k-i) + \sum_{i=1}^{m} b_i\, u(k-i) \qquad (2.142)$$

entonces aplicando la ecuación (2.141) en la ecuación (2.142) se tiene:

$$\frac{\partial\, \hat{y}\left(k,\Theta(k)\right)}{\partial\, a_1} + \sum_{i=1}^{p} c_i \frac{\partial\, \hat{y}\left(k-1,\Theta(k-1)\right)}{\partial\, a_1} = -y(k-1)$$

.

.

.

$$\frac{\partial\, \hat{y}\left(k,\Theta(k)\right)}{\partial\, a_n} + \sum_{i=1}^{p} c_i \frac{\partial\, \hat{y}\left(k-1,\Theta(k-1)\right)}{\partial\, a_n} = -y(k-n)$$

$$\frac{\partial\, \hat{y}\left(k,\Theta(k)\right)}{\partial\, b_1} + \sum_{i=1}^{p} c_i \frac{\partial\, \hat{y}\left(k-1,\Theta(k-1)\right)}{\partial\, b_1} = u(k-1)$$

.

.

.

.

$$\frac{\partial \hat{y}(k,\Theta(k))}{\partial b_m} + \sum_{i=1}^{p} c_i \frac{\partial \hat{y}(k-1,\Theta(k-1))}{\partial b_m} = u(k-m)$$

$$\frac{\partial \hat{y}(k,\Theta(k))}{\partial c_1} + \sum_{i=1}^{p} c_i \frac{\partial \hat{y}(k-1,\Theta(k-1))}{\partial c_1} = \varepsilon(k-1)$$

$$\vdots$$

$$\frac{\partial \hat{y}(k,\Theta(k))}{\partial c_p} + \sum_{i=1}^{p} c_i \frac{\partial \hat{y}(k-1,\Theta(k-1))}{\partial c_p} = \varepsilon(k-p) \qquad (2.143)$$

El anterior grupo de ecuaciones (2.143) puede ser agrupado como:

$$\Phi(k,\Theta(k)) + \sum_{i=1}^{p} c_i \Phi(k-1,\Theta(k-1)) = \begin{bmatrix} -y(k-1) \\ \vdots \\ -y(k-n) \\ u(k-1) \\ \vdots \\ u(k-m) \\ \varepsilon(k-1) \\ \vdots \\ \varepsilon(k-p) \end{bmatrix}^T \qquad (2.144)$$

De la ecuación (2.144) se observa que $\Phi(k,\Theta(k))$ se puede expresar en forma recursiva en base a las mediciones que es lo único que se conoce.

Si se utiliza el error de predicción:

$$\varepsilon(k,\Theta(k)) = y(k) - \hat{y}(k,\Theta(k-1))$$

cuando se actualiza el error de predicción se utiliza el residual que esta dado por

$$\eta(k,\Theta(k)) = y(k) - \hat{y}(k,\Theta(k))$$
$$\eta(k,\Theta(k)) = y(k) - x(k)\Theta(k)$$

Por lo que volviendo a definir el vector de información en base al residual se tiene que

$$x^T(k) = \begin{bmatrix} -y(k-1) \\ \vdots \\ -y(k-n) \\ u(k-1) \\ \vdots \\ u(k-m) \\ \eta(k-1) \\ \vdots \\ \eta(k-p) \end{bmatrix} \qquad (2.145)$$

reemplazando la ecuación (2.145) en la ecuación (2.144) se tiene:

$$\Phi(k,\Theta(k)) + \sum_{i=1}^{p} c_i \Phi(k-1,\Theta(k-1)) = x(k) \qquad (2.146)$$

entonces despejando de la ecuación (2.146) $\Phi(k,\Theta(k))$ se tiene:

aplicando el Lema de inversión de matrices bajo el mismo concepto que en el método de mínimos cuadrados recursivos haciendo:

$$P(k+1) = V''[k+1]^{-1}$$

fila copiarla en el vector $x(k+1)$, para utilizarlo en las funciones de cálculo del vector $L(k+1)$ y matriz $P(k+1)$.

$$x(k+1) = XL[0][i] = -\sum_{i=1}^{p} c_i XL[0][k-i] + XMLH[k+1]$$

8.- Calcular la matriz $L(k+1)$ de acuerdo a:

$$L(k+1) = \frac{P(k)x^T(k+1)}{1 + x(k+1)P(k)x^T(k+1)}$$

9.- Hallar la matriz $P(k+1)$ de acuerdo a:

$$P(k+1) = \left(I - \frac{P(k)x^T(k+1)x(k+1))}{1 + x(k+1)P(k)x^T(k+1)}\right)P(k)$$

10.- Calcular el valor de los estimados $\hat{\Theta}(k+1)$ según la ecuación:

$$\hat{\Theta}(k+1) = \hat{\Theta}(k) + P(k+1)\,x(k)\,\varepsilon(k+1)$$

11.- Encontrar el valor del residual de acuerdo a la ecuación:

$$\eta(k+1) = y(k+1) - x(k+1)\Theta(k+1)$$

12.- Incrementar k.

13.- Ir al paso 4.

CAPITULO III: PROGRAMA PARA IDENTIFICACIÓN PARAMÉTRICA DISCRETA

3.1 LENGUAJE DE PROGRAMACIÓN Y COMPILADORES

3.2 UTILIZACIÓN DEL PAQUETE ProtoGen+.

3.3 PROGRAMA PRINCIPAL

3.4 RUTINA DE INGRESO DE DATOS

3.5 RUTINA DE SIMULACION DE PLANTA Y RUIDO

3.6 RUTINA DE ADQUISICIÓN DE DATOS

3.7 RUTINA DE MÍNIMOS CUADRADOS RECURSIVOS

3.8 RUTINA DE MÍNIMOS CUADRADOS GENERALIZADOS

3.9 RUTINA DE MÁXIMO DE VEROSIMILITUD

3.10 RUTINA DE GRÁFICOS DE SEÑALES Y CONVERGENCIA DE PARÁMETROS

A continuación se detallan los paquetes de software (compiladores) que se utilizan para el desarrollo del programa de ***"Identificación Paramétrica Discreta"***, la estructura del mismo, sus módulos más importantes y la forma como estos interactuan para tener finalmente el programa para identificación paramétrica discreta ***"IPD.EXE"***. Se presenta además el diagrama de flujo de las rutinas más importantes en el programa. También se analizan las características mínimas necesarias del equipo para utilizar el programa de identificación.

3.1 LENGUAJE DE PROGRAMACIÓN Y COMPILADORES

Al desarrollar software para una cierta aplicación es necesario conocer las necesidades básicas, y posibles usos que se le va a dar al mismo para de esta manera seleccionar adecuadamente un lenguaje de programación, que satisfaga las condiciones anteriores.

El software que se desarrolló en esta tesis esta enfocado a trabajar a nivel de simulación así como también en tiempo real, por lo que fue necesario utilizar un lenguaje de programación que sea versátil en la creación de un ambiente agradable de trabajo (bajo la plataforma Windows 3.1) y potente en el manejo de periféricos (tarjeta de adquisición de datos); por lo cual se escogió el lenguaje "C", debido a que tiene una gran cantidad de recursos para crear software agradable al usuario y una gran capacidad de tener un control total de los periféricos, que lo hacen útil para trabajar en aplicaciones en línea.

Entre los compiladores más importantes para el desarrollo de programas en C, se tiene el *BORLAND C++*, sin embargo para el desarrollo de aplicaciones en ambiente Windows se tiene software adicional de programación que convierte algunas tareas comunes de programación en un desarrollo gráfico del mismo, para estas tareas se utiliza el programa *ProtoGen+*.

Por lo tanto para desarrollar el programa de identificación se utilizó los 2 paquetes anteriores que son:

- *ProtoGen+* ; ver 4.2 1994-1995.
- *BORLAND C++ FOR WINDOWS* ; ver 3.1 1993.

Para el correcto funcionamiento del software desarrollado es necesario tener los requerimientos mínimos para que la plataforma Windows versión 3.1 funcione adecuadamente, los mismo que son suficientes para que el software de identificación paramétrica discreta trabaje correctamente.

Estos requerimientos son:

- Computador 386 o superior.
- Sistema Operativo MS-DOS ver 5.0 ó posterior y la plataforma Windows 3.1.
- Memoria mínima 4 MB.

Para un trabajo a nivel de tiempo real es necesario además que se posea la tarjeta de adquisición de datos, que en este caso es la DAS-128.

Es necesario indicar que a medida que el equipo de computación sea más potente el trabajo en el programa "IPD.EXE" es mucho más rápido y agradable.

3.2 UTILIZACIÓN DEL PAQUETE ProtoGen+.

El lenguaje C como se indicó en el numeral anterior es el lenguaje más idóneo para crear aplicaciones para el ambiente Windows, en este lenguaje se puede tener básicamente 2 filosofías de desarrollo de software, la primera (que se utiliza en esta tesis) es crear aplicaciones utilizando el sistema de mensajes y la segunda utilizando una programación orientada a objetos.

El programa *"IPD.EXE"* se desarrolló utilizando el sistema de mensajes, un mensaje desde el punto de vista de una aplicación se puede considerar como una notificación de que se ha producido algún suceso de interés que puede o no requerir de una acción específica. Los mensajes en Windows pueden provenir de 4 fuentes que son: el mismo Windows, el propio programa, otra aplicación o el usuario.

En general cada programa tiene un bucle de procesamiento de mensajes que es el encargado de procesar los distintos mensajes que le lleguen al mismo a través de Windows y realizar alguna acción en caso de que el mensaje sea de interés. A través del sistema de mensajes se logra tener un sistema multitarea que es el que se maneja por medio de Windows. Además de esta característica, es necesario indicar que una de las mayores ventajas que se tiene al desarrollar software para Windows es que el programador se despreocupa del manejo de los periféricos ya que este manejo lo hace la misma plataforma.

Luego de haber explicado brevemente la forma como se comunica el programa con la plataforma Windows se procede a

indicar algunas características del programa *ProtoGen+* que ayuda al desarrollo de aplicaciones para Windows.

El programa *ProtoGen+* es un conjunto de herramientas que ayudan a desarrollar de una manera gráfica: menús, cuadros de diálogos, ventanas especiales, íconos, etc, para luego dicha información guardada en archivos de recurso (archivos gráficos) sea transformada en archivos de código C, el cual puede ser editado y modificado en el compilador *BORLAND C++*. Además de realizar la interface gráfica el programa *ProtoGen+* tiene algunas tareas básicas como la captura de información desde los cuadros de dialogo de una manera transparente al programador. El programa *ProtoGen+* da la capacidad de a la vez que se va desarrollando la interface gráfica de la aplicación ir paulatinamente transformando esta información a lenguaje C e ir escribiendo las funciones o rutinas propias del programa que se esta desarrollando.

Luego de que la interface gráfica ha sido desarrollada y transformada en archivos en C, estos archivos están en capacidad de manejar los distintos mensajes asignados al programa y a cada cuadro de diálogo, para realizar una tarea específica que haya sido desarrollada.

Para completar el desarrollo se utiliza BORLAND C++, en donde se adiciona en cada lugar las llamadas a las rutinas o funciones correspondientes y dichas rutinas se las guarda en archivos de cabecera. Esta forma de programar da la capacidad de poder cambiar la forma de presentación las veces que sea necesario, mientras paralelamente se va desarrollando las rutinas para realizar un trabajo específico, dichas funciones se pueden escribir en archivos

de cabecera llamados *"nombre.h"* y que se incluyen al inicio del archivo en C que va a ser uso de las funciones escritas en dicho archivo.

3.2 PROGRAMA PRINCIPAL

El módulo denominado programa principal es el responsable de manejar la comunicación entre el ambiente Windows y el software desarrollado, el código de este programa se encuentra bajo el nombre de *ipd.c.*

En general este módulo provee la capacidad de manejar los distintos mensajes que le envía el sistema o el usuario al programa *"IPD.EXE"*, y de esta manera encaminarlos a los restantes módulos, estos mensajes en el programa son entre otros los siguientes: abrir/cerrar un cuadro de dialogo, abrir/cerrar archivos, etc.

El programa *"IPD.EXE"* tiene un menú principal que abarca las siguientes opciones:

Archivos, Planta, Entrada, Modelos, TiempoReal, Gráficos, Acerca.

Las cuales a su vez tienen sub-menus para ingresar o presentar resultados de acuerdo al tipo de cuadro de diálogo que se trate.

En la figura 3.2.1 se muestra la pantalla inicial del programa de Identificación Paramétrica Discreta.

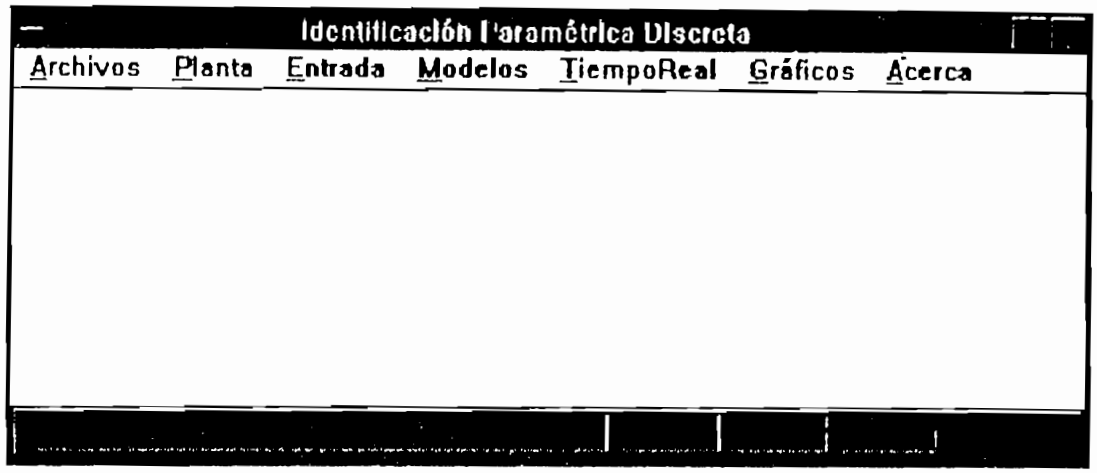

Fig. 3.2.1 Menú Principal del programa IPD.EXE

A continuación se realiza una breve descripción de las opciones del programa principal, en tanto que en el apéndice A (Manual de Usuario) se encuentra una explicación detallada de como opera el programa de identificación y lo que realiza cada submenu.

- ***OPCIÓN ARCHIVOS***

La opción ***Archivos\Abrir*** es la encargada de recuperar información de un archivo que tiene extensión *"dat"*, la información es referida al vector de ruido en la entrada y en la salida del sistema. Esta opción funciona a nivel de simulación solamente.

La opción ***Archivos\Guardar*** esta ligada a la capacidad del programa de guardar en un archivo todos los datos sobre la identificación de un sistema ya sea en simulación o en tiempo real en extensión *"ipd"*, mientras que si el archivo es de extensión *"dat"* se guarda los vectores de ruido.

En la opción ***Archivos\Modo de Trabajo*** se escoge la forma como va a trabajar el programa ya sea a nivel de simulación o en tiempo real.

La opción ***Archivos\Salir*** *ALT-F4* característica en WINDOWS sirve para terminar la ejecución del programa.

- ***OPCIÓN PLANTA***

Este sub-menu esta disponible solamente cuando se trabaja a nivel de simulación.

En la opción ***Planta\Orden*** se ingresa el orden de los polinomios y retardo de la planta que se va a simular.

En la opción ***Planta\Coeficientes*** se deben ingresar los coeficientes de los polinomios de la planta que se va a simular.

En la opción ***Planta\Ruido*** se tiene la capacidad de sumar ruido a la salida ya sea ruido blanco o correlacionado.

- ***OPCIÓN ENTRADA***

En La opción ***Entrada\Excitación,*** se tiene que escoger el tipo de entrada que se va a utilizar en el proceso de identificación.

En la opción ***Entrada\Condiciones iniciales,*** se ingresa las condiciones iniciales para el funcionamiento del algoritmo de identificación.

En la opción ***Entrada\Factor de olvido***, se escoge el tipo de factor de olvido a utilizar.

- ***OPCIÓN MODELOS***

En esta opción se presenta un cuadro de diálogo en el cual se tiene la posibilidad de escoger el tipo de modelo a utilizar en la identificación, luego se puede ingresar el número de coeficientes que debe tener cada modelo en el botón *COEF*, finalmente para presentar los resultados (solamente nivel de simulación) se tiene el botón de *Ident* (identificar el sistema).

- ***OPCIÓN TIEMPO REAL***

Este submenú solamente se encuentra disponible cuando se trabaja a nivel de tiempo real.

La opción ***TiempoReal\Adquisición entrada salida***, es donde se puede ver en forma gráfica los valores de la entrada y salida del sistema.

La opción ***TiempoReal\Identificación numérica***, permite identificar en línea al sistema y todos los parámetros identificados se los presenta en forma numérica.

La opción ***TiempoReal\Identificación gráfica***, da la capacidad de visualizar como el programa va identificando uno de los parámetros del modelo.

La opción ***TiempoReal\Reporte final***, muestra los resultados de la última identificación en tiempo real.

- ***OPCIÓN GRÁFICOS***

En la opción ***Gráficos\Parámetros*** se puede ver el gráfico del parámetro seleccionado a lo largo del proceso de identificación.

En la opción ***Gráficos\Entrada Salida*** se presenta los gráficos de la entrada y salida del sistema.

La opción ***Gráficos\Error Residual***, se encuentra habilitada solamente a nivel de simulación y a través de esta se tiene la capacidad de ver como el error de predicción para el método de mínimos cuadrados recursivos y mínimos cuadrados extendidos o el residual para el máximo de Likelihood, se aproxima al ruido blanco en la salida del sistema.

- ***OPCIÓN ACERCA***

En esta opción se despliega un cuadro de dialogo referente a datos del autor así como la fecha del desarrollo del software.

3.4 RUTINA DE INGRESO DE DATOS

Antes de proceder con cualquier algoritmo de identificación es necesario ingresar datos al programa para que este opere adecuadamente, los principales datos que se tienen que ingresar al programa se encuentran en el submenu **Entrada** en la barra del menú principal.

En la opción ***Excitación*** del submenu **Entrada** se tiene que escoger el tipo de entrada la cual se va a aplicar al sistema para identificarlo. La entrada que se puede escoger esta formada por una señal escalón a la cual se le puede sumar ruido blanco el cual puede ser randómico, PRBS o estadístico, generado a través de los 3 algoritmos indicados anteriormente. Se puede escoger también solamente ruido como señal de entrada.

En la opción ***Condiciones Iniciales***, del submenú **Entrada** se establece el valor inicial de los parámetros más importantes para el desarrollo de los algoritmos de identificación. Los parámetros que se inicializan son: el valor alfa de la matriz de covarianza $P(k)$, el número de iteraciones que va a identificar el sistema, además del porcentaje de valores que se utilizarán para obtener el promedio de los parámetros identificados. También se puede escoger valores iniciales para los parámetros a identificar.

En la opción ***Factor de olvido*** del submenú **Entrada** se escoge el tipo de factor de olvido que se utilizará durante el proceso de identificación y que puede ser de tres tipos:

- Factor de olvido constante.

- Factor de olvido tipo filtro.
- Factor de olvido exponencial.

Las opciones anteriores se encuentran disponibles tanto a nivel de simulación así como también en tiempo real.

Al trabajar a nivel de simulación es necesario ingresar información sobre la planta que se va a simular, esta información se debe ingresar en el submenú ***Planta*** de la barra del menú principal.

En la opción ***Orden*** del submenu ***Planta*** se ingresa el orden de los polinomios $A(z)$ y $B(z)$ así como retardo existente.

En la opción ***Coeficientes*** del submenu ***Planta*** se debe ingresar los coeficientes de los polinomios $A(z)$ y $B(z)$.

La opción ***Ruido*** del submenu ***Planta*** esta relacionada a la opción de poder ingresar ruido en la salida ya sea ruido blanco o ruido correlacionado en los distintos tipos de ruido disponibles (randómico, PRBS y estadístico)

En este cuadro de diálogo se puede escoger el porcentaje de ruido que se va asumar a la salida, y en caso se escoger ruido correlacionado se debe ingresar los coeficientes c del ruido.

Todas las variables globales que se utilizan se pueden encontrar en el archivo "ipd.h" en el cual además se tiene la definición de las llamadas a los cuadros de diálogos utilizados en el programa.

3.5 RUTINA DE SIMULACIÓN DE PLANTA Y RUIDO

En los archivos "simular.h" y "esruido.h" se tienen las funciones para simular la planta y generar la señal de entrada (excitación persistente) respectivamente, así como el ruido necesario que se puede añadir a la salida en caso de estar trabajando a nivel de simulación.

La rutina para simular la planta esta formada por un conjunto de funciones que interactuan entre sí, y son:

- Inicialización del proceso de simulación. Esta función esta definida en el programa como *IniciarSimulación()* y lo que se hace es inicializar los vectores asociados al sistema a simular. Estos vectores son: vector de salida denominado yPS(), vector de entrada llamado uPS(), y el vector de ruido denominado ruidoS(). Se utiliza un arreglo vectorial para estas variables debido a que es necesario conocer el valor actual y los estados anteriores de dichas variables los mismos que se almacenan en los vectores de acuerdo al orden que tengan. Esta función se la utiliza cada vez que se requiere iniciar o reiniciar el proceso de simulación.

- Simulación de la planta. Esta función es la más importante en el archivo "simular.h" y en el programa se la encuentra definida como *Simulaciondey()*, esta función es la encargada de traer el valor actual de la entrada uPS[0] y actualizar la salida de la planta yPS[0], y en caso de existir ruido a la salida sumarle dicho valor.

Se trae el valor de entrada uPS[0] a través de la función Entradamcr() y el valor de ruido a la salida por medio de la función Ruidomcr() funciones que se encuentran definidas en el archivo "esruido.h".

- Actualización de los vectores utilizados. Esta función en el programa se conoce con el nombre de *ActualizarPS()*, y esta encargada de desplazar un instante la información en los vectores de yPS(), uPS() y ruidoS(), que es necesario para un nuevo proceso de cálculo.

- Valor final de la planta. Esta función en el programa se conoce con el nombre de *ValorFinal()*, y es la encargada de encontrar el valor final para la planta de acuerdo al valor de la entrada que se tenga, sin tomar en cuenta el ruido. Para el caso de tener una entrada formada solamente por ruido se toma el valor máximo de dicho ruido y con este valor se calcula el valor final del sistema.

La rutina para generar los datos de entrada así como también el porcentaje de ruido que se suma a la salida esta dada por las siguientes funciones:

- Valor de ruido a la entrada y salida. En el programa esta función se conoce con el nombre de *VRuidoES()*, y es la encargada de generar 2 vectores de ruido de acuerdo a la selección para el ruido ingresado en el la opción *Entrada\Excitación* y *Planta\Ruido*, la dimensión de estos 2 vectores es el número de iteraciones que se vayan a realizar. Los vectores de ruido se almacenan en dos arreglos, para la entrada como vruido1[], y para la salida como vruido2[], los mismos que son generados con media cero y una amplitud alrededor de 1.

- Valor de la entrada. Esta función en el programa se conoce con el nombre de *Entradamcr()*, y es la encargada de tomar un valor del vector1[] de ruido dependiendo de la iteración en la cual se encuentre el proceso, es decir a través de esta función se actualiza el valor de la entrada que se encuentra reflejada en uPS[0].

- Valor de ruido a la salida. En el programa esta función se conoce con el nombre de *Ruidomcr()*, y es la encargada, dependiendo de la iteración en que se encuentre el proceso, de sumarle ruido a la salida, pero tomando en cuenta el porcentaje del mismo, así como también pasandola por un polinomio en caso de que el ruido a la salida que se haya escogido sea ruido correlacionado. Esta opción de sumar ruido a la salida solamente se encuentra habilitada a nivel de simulación.

Las rutinas de simulación y generación de datos son usadas en los 3 algoritmos de identificación de sistemas (MCR, MCG, MLH), por lo que se la llama desde cualquier parte de dichos algortimos cuando se necesita los valores de uPS[0] y yPS[0] valores que se asignan a las vectores de identificación uPI[0] y yPI[0] que se utilizan en las funciones de identificación.

Un diagrama de flujo de esta rutina se realiza en la rutina de mínimos cuadrados recursivos en donde se observa la manera como las funciones interactuan para tener el valor de entrada y salida de la planta que se simula y que va luego a ser identificada.

3.6 RUTINA DE ADQUISICIÓN DE DATOS

Esta rutina de adquisición y salida de datos es importante cuando se trabaja a nivel de tiempo real en la identificación de sistemas. Para que un sistema sea identificable es necesario que la entrada tenga ciertas características, por lo que es necesario generar esta señal y enviarla al sistema real para luego medir la respuesta del mismo y con este par de datos a través del tiempo poder realizar la identificación.

La interface que se utiliza para enviar y recibir datos es la tarjeta de adquisición de datos DAS-128 [7], la cual maneja los puertos de entrada y salida de datos como determinadas posiciones de memoria, por lo cual es necesario tener instrucciones adecuadas para la escritura y lectura de los puertos, en el lenguaje C se tiene:

- *void outportb(int salida,unsigned char dato)*, esta función pone el valor del dato (0-255) en el puerto especificado por la variable salida [22].

- *unsigned char inportb(int entrada)*, esta función devuelve el valor del byte que se encuentra en el puerto especificado en la variable entrada [22].

Estas 2 instrucciones son las utilizadas para leer y escribir en los puertos utilizando la tarjeta DAS-128.

Al realizar un proceso en tiempo real es necesario definir un tiempo de muestreo entre cada lectura y escritura de los datos, por lo cual es necesario medir el tiempo de ejecución

del algoritmo y de acuerdo a esto poder establecer el mínimo tiempo de muestreo que se puede tener.

Un diagrama basado en el período de muestreo se muestra en la figura 3.7.

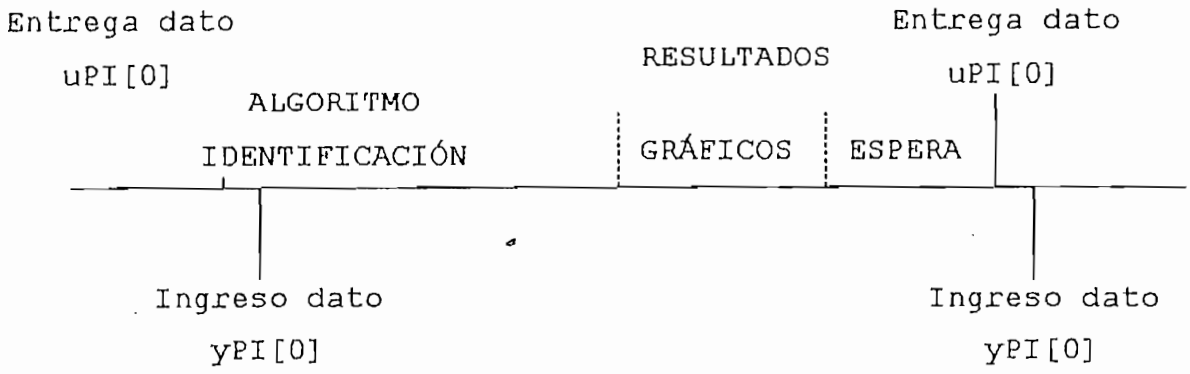

Fig. 3.6.1 Período de Muestreo

Cuando se tiene *ESPERA* en el diagrama de tiempo del período de muestreo, esto significa que el programa en este momento deja de utilizar el microprocesador y vuelve a tener el control del mismo cuando el temporizador utilizado ha completado el tiempo de muestreo (generando un mensaje WM_TIMER); es decir, este tiempo el usuario a través de la plataforma Windows puede utilizarlo para realizar otra tarea que puede ser totalmente ajena al programa de identificación en tiempo real y solamente realizará la identificación cuando se haya generado el mensaje de interrupción.

Para tener el tiempo de muestreo el programa hace uso de uno de los 16 temporizadores disponibles en la plataforma Windows, por lo que es necesario asegurar que exista al menos 1 temporizador disponible, ya que en caso contrario no se podrá realizar identificación en tiempo real.

Las instrucciones utilizadas en lenguaje C para habilitar y desabilitar un temporizador se indican a continuación [18].

- *UINT SetTimer(hwnd,idTimer,uTimeout,tmprc)* , esta función permite habilitar el temporizador donde los parámetros de la misma están dados por las siguientes variables:

hwnd, manipulador de la ventana.
idTimer, identificador del temporizador.
UTimeout, duración del temporizador.
tmprc, dirección de un procedimiento.

Esta función luego de ser ejecutada devuelve un valor que sirve para analizar y determinar si se ha producido una correcta activación del temporizador.

- *BOOL KillTimer(hwnd,idTimer)*, esta función desabilita el temporizador de una ventana especificada de acuerdo a su identificación, las variables que utiliza son:

hwnd, manipulador de la ventana.
IdTimer, identificador del temporizador.

Esta función devuelve un valor distinto de cero si se ha producido la desactivación del temporizador, en caso contrario devuelve el valor cero.

Los resultados de la identificación al trabajar en tiempo real se pueden presentar en forma gráfica o numérica, sin embargo en cualquiera de los 2 casos se tiene una pantalla de resultados, la misma que se va actualizando con cada nuevo valor de identificación. En esta pantalla existen dos

botones de control adicionales que son *INICIAR* , *PARAR* tal como se indica en la figura 3.6.2.

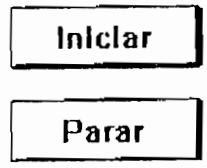

Fig. 3.6.2 Diagrama de controles adicionales en tiempo real

El botón *INICIAR* es el encargado de generar un mensaje para que se active el temporizador, y por lo tanto empiece el proceso de identificación, mientras que el botón de *PARAR* es el encargado de desabilitar el temporizador y por consiguiente suspender la identificación.

La identificación puede continuar volviendo a presionar el botón de *INICIAR*.

A continuación, en la figura 3.6.3 se muestra un diagrama de flujo de la rutina de adquisición y salida de datos, luego de que se ha ingresado en la pantalla de presentación de resultados (numéricos o gráficos), y en donde se observa que es lo que se realiza en el momento de la interupción.

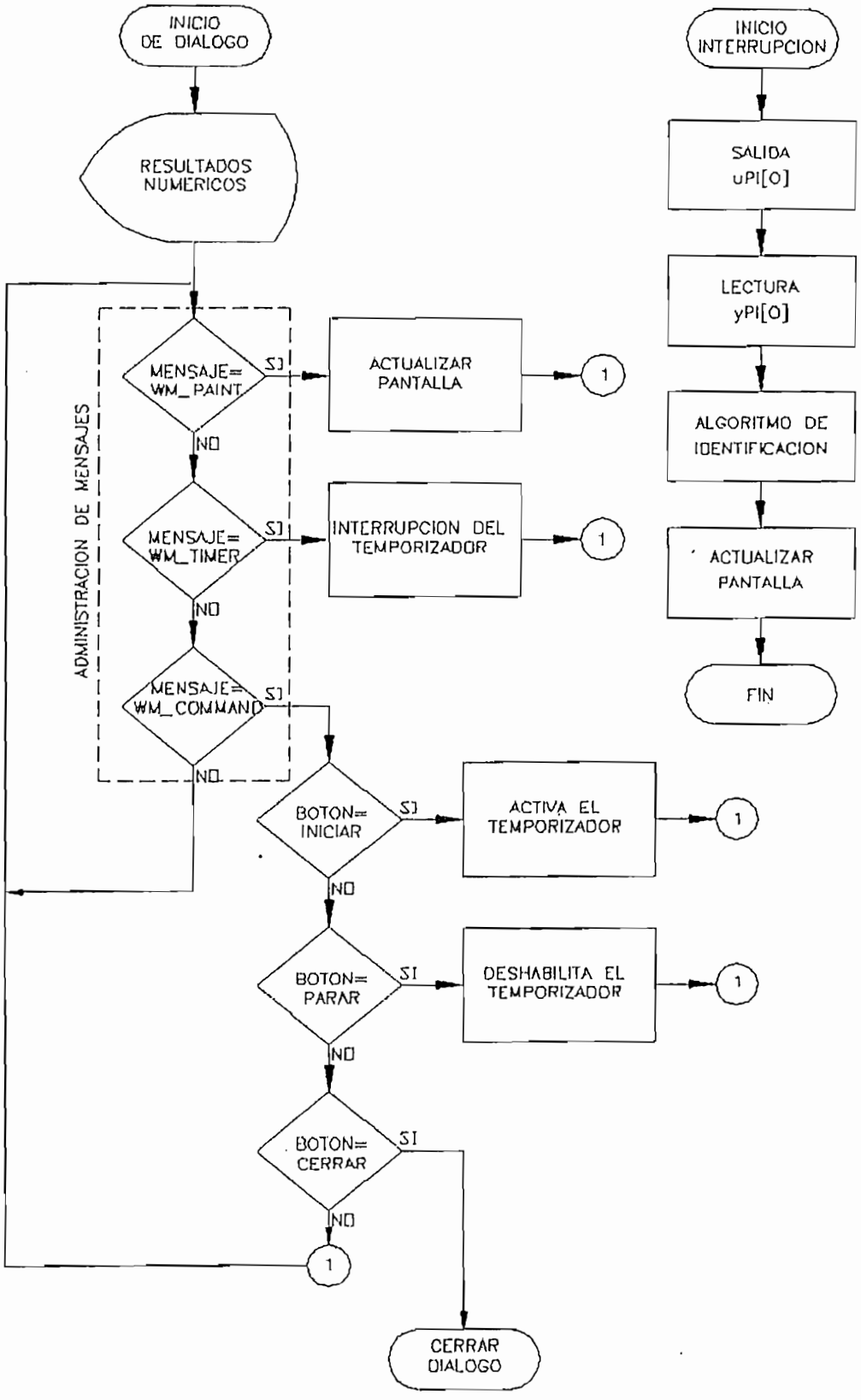

Fig. 3.6.3 Diagrama de la identificación, adquisición y salida de datos

3.7 RUTINA DE MÍNIMOS CUADRADOS RECURSIVOS (MCR)

Esta rutina de identificación esta implementada a través de funciones las cuales realizan una tarea específica y que se encuentran en el archivo "calmcr.h", estas funciones se agrupan para realizar el proceso de identificación en el archivo "mcr.h", el cual es un archivo de cabecera, para el archivo de presentación de resultados del método de MCR.

En el archivo mcr.h se forma el bucle de identificación llamando tanto a las funciones para simular al sistema (trabajando a nivel de simulación), como las funciones para identificar al sistema. Mientras que trabajando a nivel de tiempo real se utilizan las funciones del archivo "calmcr.h" para identificar al sistema y las rutinas de adquisición de datos en la parte de codigo correspondiente a la interrupción del timer.

Las principales funciones para identificar a un sistema mediante el método de mínimos cuadrados recursivos, que se encuentran en el archivo "calmcr.h" son:

- *Transferencia()*, esta función es la encargada de tomar los valores anteriores del vector de salida definido como YPI() y de la entrada definida como uPI(), para formar el vector de información llamado X(). Estos valores se los toma de los datos simulados cuando se trabaja a nivel de simulación o medidos cuando se trabaja en tiempo real.

- *Error()*, esta función permite calcular el error de predicción utilizando el vector de información y el vector

de parámetros estimados $\hat{y}$, definido como theta(). El valor del error se almacena en la variable ErrorXYth.

- *VectorL()*, esta función se encarga de calcular el valor del arreglo L(), que se conoce como ganancia.

- *Matriz P()*, esta función es la encargada de calcular el valor de la matriz de covarianza, en función del vector L() definido anteriormente, esta matriz se almacena en el arreglo P().

- *theta()*, esta función es la encargada de actualizar el valor de los parámetros que se esta estimando luego de haber calculado el error de predicción así como también el vector L() de corrección. Los parámetros se almacenan en el vector theta().

Además de las funciones descritas anteriormente existen otras funciones necesarias para el proceso de identificación entre las que se tiene:

- *InicializarAdd()*. Esta función es la encargada de inicializar los vectores asociados al modelo que se va a identificar uPI() vector de entrada, yPI() vector de salida, matriz de covarianza $P(k)$ y el valor del vector de parámetros theta().

- *ActualizarPI()*. Esta función se encarga de desplazar un instante la información contenida en los vectores uPI(), yPI() para un nuevo instante de cálculo.

Dependiendo si la presentación de los resultados es en forma numérica o por medio de gráficos de los valores estimados, se tiene rutinas adicionales de presentación de resultados.

Los datos para el correcto funcionamiento del algoritmo son ingresados en la opción *INGRESO* del menú principal que se explicó anteriormente, por lo que aquí solamente se hace uso de las respectivas rutinas de identificación.

Al trabajar a nivel de simulación, la rutina de identificación, es llamada cuando se crea el cuadro de diálogo para la presentación de los resultados ya se a nivel numérico o en forma gráfica, mientras que al trabajar a nivel de tiempo real, la identificación se realiza paso a paso (en línea) de acuerdo al tiempo de muestreo seleccionado.

A continuación se presenta un diagrama de flujo en la figura 3.7.1 tanto de la rutina de identificación de parámetros, así como de la rutina de generación de ruido y simulación de la planta debido a que esta última solo existe cuando se va a realizar identificación de sistemas por cualquiera de los 3 algoritmos descritos en el capítulo II.

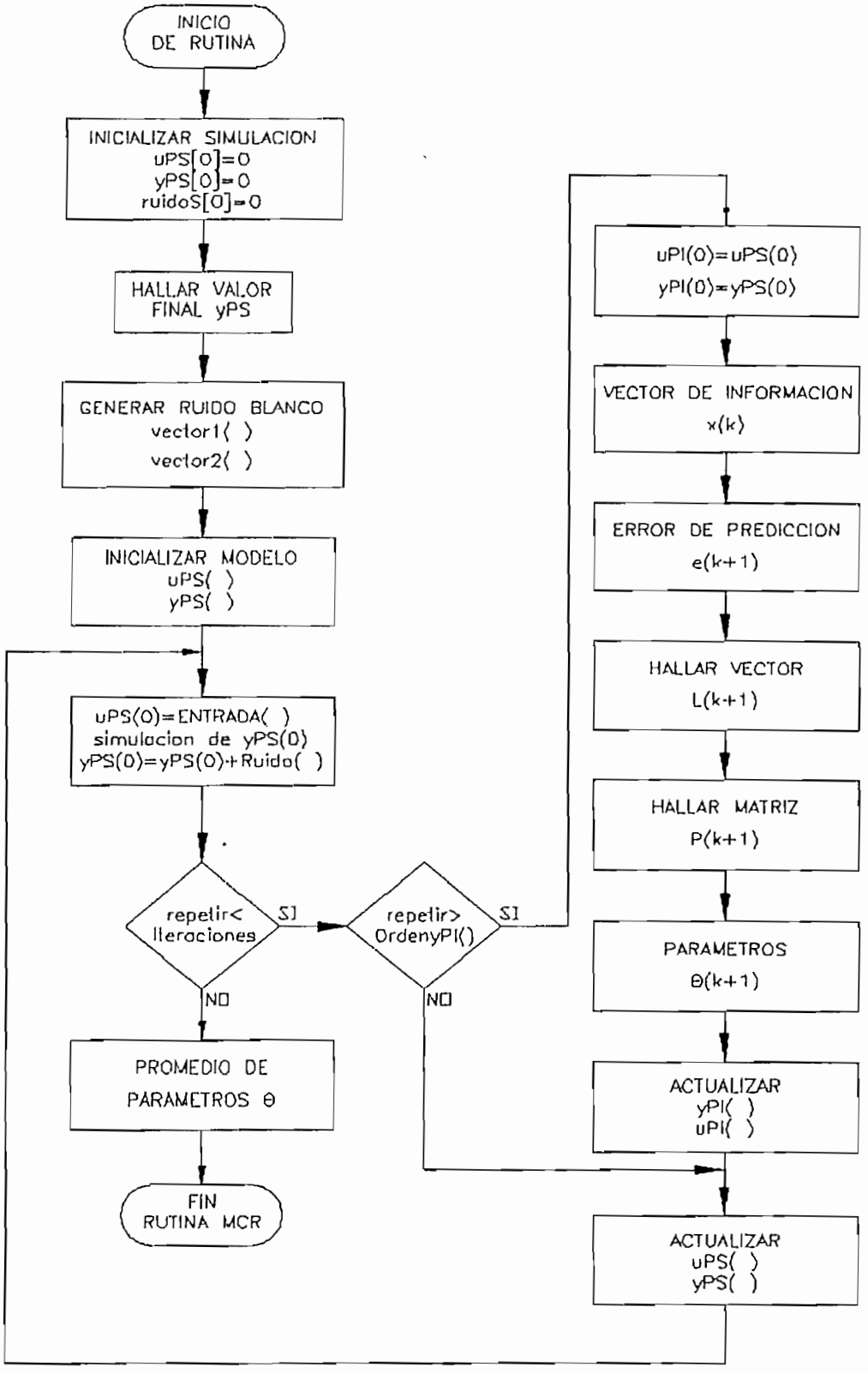

Fig. 3.7.1 Diagrama de flujo del algortimo de MCR y de la simulación de planta

3.8 RUTINA DE MÍNIMOS CUADRADOS GENERALIZADOS (MCG)

En el capítulo II, se indicó la forma como el método de MCG se iba a implementar mediante la extensión del vector de información X(), hacia los errores de predicción por lo cual la rutina de MCG, es una extensión de la rutina de MCR y algunas funciones de identificación definidas en el archivo "calmcr.h" se utilizan en este método de identificación.

Al igual que en el caso de los mínimos cuadrados recursivos para el caso de la rutina de mínimos cuadrados generalizados se tiene 2 archivos de cabecera, así; el archivo "mcg.h" en el cual se forma el bucle de identificación de parámetros (trabajando a nivel de simulación) llamando a las funciones que son necesarias, y el archivo "calmcg.h" en donde se encuentran funciones para la identificación.

Al igual que en el caso de mínimos cuadrados recursivos, cuando se identifica un sistema en tiempo real solamente se utilizan las funciones de identificación definidas en los archivos "calmcg.h" y "calmcr.h".

A continuación se explicará las funciones adicionales creadas para identificar un sistema mediante MCG.

- *InicializarPlanta()*. Esta función se encargada de inicializar el valor de los vectores de entrada uPI(), salida yPI() el errorPMCG() de acuerdo al orden ingresado, además de estas variables se inicializa otros parámetros importantes en el programa (dichos parámetros se explicaron en la Rutina de Ingreso de datos anteriormente explicada).

- *transferenciaXMG()*. Esta función es la encargada de formar el vector de información sin embargo no solo se toma la información de los arreglos de entrada uPI() y salida yPI(), sino también del vector de errores de predicción errorPMCG(), para tener el vector de información extendido X().

- *ErrorYMCG()*. Esta función permite actualizar el valor del error de predicción, utilizando el nuevo vector de información y el vector de parámetros estimados theta().

- ActualizaruyPIMCG(). Esta función es la encargada de desplazar la información en los vectores de entrada uPI(), salida yPI() y el vector de ruido errorYMCG() para un nuevo instante de cálculo.

En forma similar que en el caso de los MCR, el archivo "mcg.h" se encuentra ubicado en la cabecera del archivo de presentación de resultados para el método de mínimos cuadrados generalizados.

Al tratarse de un algoritmo similar al de mínimos cuadrados recursivos por las explicaciones dadas anteriormente, en este caso no se realiza el diagrama de flujo para esta rutina.

3.9 RUTINA DE MAXIMO DE VEROSIMILITUD (MAXIMO DE LIKELIHOOD MLH)

La rutina de máximo de verosimilitud (MLH), nos da la posibilidad de identificar parámetros en un sistema, cuando en este existe la presencia de ruido blanco ó ruido correlacionado.

Esta rutina de identificación esta escrita en 2 archivos de cabecera que se encuentran en el archivo de presentación de resultados y son: "mlh.h" en el cual se maneja el bucle de simulación e identificación y el archivo "calmlh.h" en donde se encuentran solo las funciones para identificar a un sistema.

Entre las principales funciones que se encuentran en el archivo "calmlh.h" para identificar a un sistema mediante el método de máximo de verosimilitud se tiene:

- *InicializarAddMLH()*. Esta función es la encargada de determinar el número de parámetros que se va a identificar y además de inicializar los vectores de entrada uPI(), de salida yPI(), así como el vector de ruido (residual) RuidoMLH(), e inicializar parámetros necesarios para el buen funcionamiento del programa.

- *TransferenciaXLMLH()*. Esta función se encarga de recoger la información de los estados anteriores de la entrada, salida así como del residual, estos valores se los almacena en una fila de la matriz XL().

- *ErrorYMLH()*. Esta función calcula el error de predicción utilizando el valor actual de la salida yPI[0], así como la información definida en XL() y los parámetros theta(), el resultado se almacena en la variable ErrorXYMLth.

- *FiltradoMLH()*. Esta función es la encargada de filtrar la matriz de información Xl() a través de los valores anteriores de esta, de acuerdo al número de parámetros c, del modelo planteado.

- *thetaMLH()*. En esta función se actualiza el valor de los parámetros estimados.

- *RuidoMLH()*. Esta función es la encargada de calcular el residual, y este almacenarlo en RuidoMLH[0], para utilizar los valores anteriores del residual cuando se forme la fila de información en la matriz Xl().

- *ActualizaruyPIMLH()*. En esta función se desplaza la información en los vectores uPI(), yPI(), RuidoMLH() y la matriz de información XL(), para un nuevo instante de muestreo.

Un diagrama que explica el flujo de la información al identificar un sistema mediante este método se muestra en la figura 3.9.1.

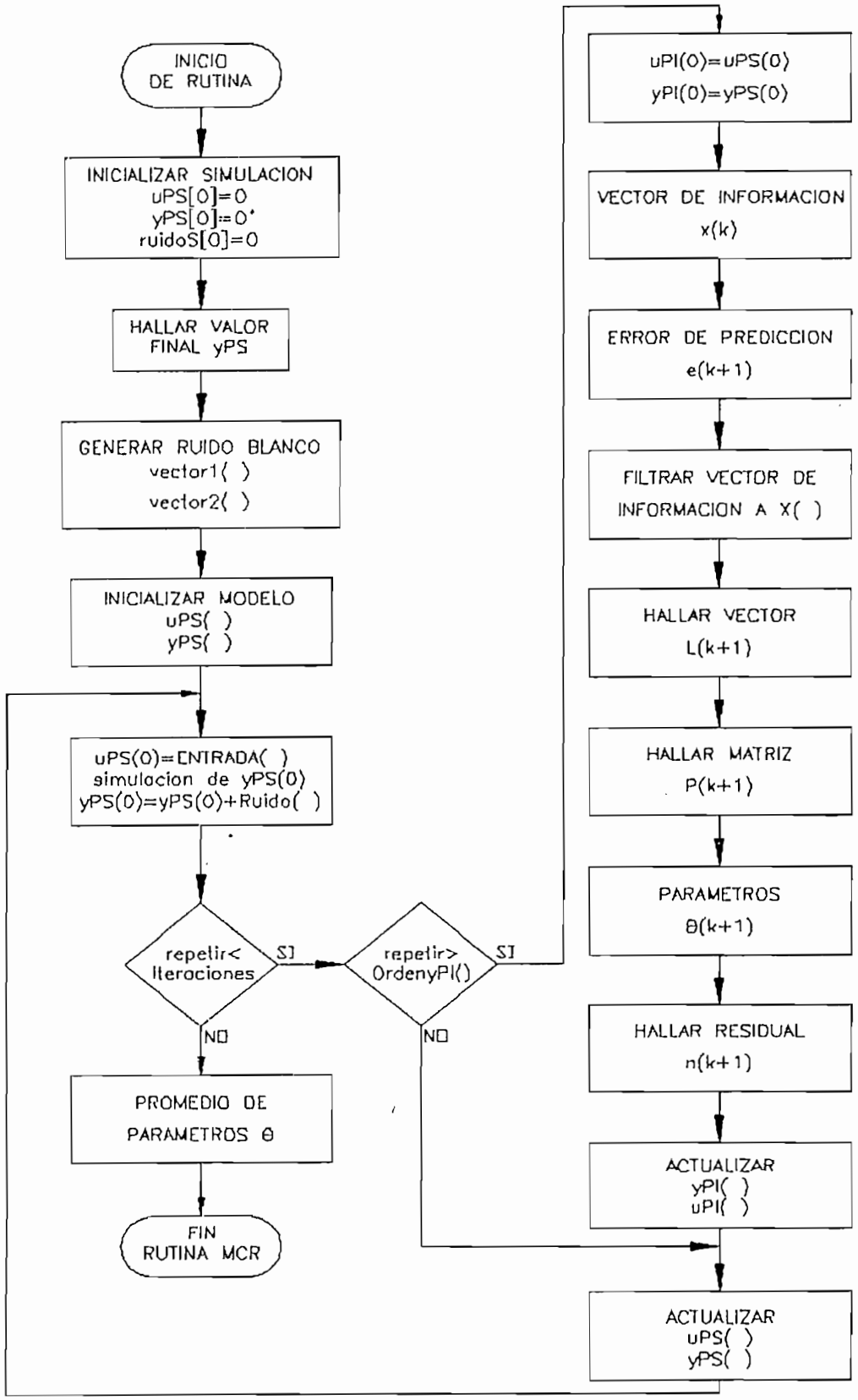

Fig. 3.9.1 Diagrama de flujo. Método de Máximo de Likelihood

3.8 RUTINA DE GRÁFICOS DE SEÑALES Y CONVERGENCIA DE PARÁMETROS

La rutina de gráficos es una de las más importantes debido a que a través de esta se puede concluir o no si un proceso de identificación esta convergiendo.

En general las líneas de código para graficar, las señales son similares por lo que estas se repiten en los cuadros de resultados en la parte correspondiente al mensaje WM_PAINT.

Para graficar las señales se definió una matriz de datos en donde se carga a lo largo del proceso de identificación los valores de los parámetros, la entrada y salida al sistema así como también los errores de predicción y el residual.

Entonces al estar en el cuadro de diálogo correspondiente se escoge el gráfico que se desea ver, esta información es analizada y transformada a la columna en la matriz de datos y esta información es la que se va a graficar.

Al graficar un parámetro a nivel de simulación se presenta una línea que indica el valor del parámetro estimado de acuerdo al valor que este tenga y que parte desde el final hasta donde se tomo el número de datos para extraer el promedio.

Es necesario indicar que a nivel de tiempo real el tener que graficar los parámetros hace que el tiempo de muestro pueda ir variando del valor seleccionado esto debido a que cada vez que se adquiere un nuevo punto es necesario volver a graficar todos, por lo que se recomienda escoger 200

iteraciones en el vector de entrada al sistema, con lo cual se recogerá solamente los 200 últimos datos para graficar el sistema y el tiempo de muestreo no variará radicalmente.

CAPITULO IV: RESULTADOS Y CONCLUSIONES

4.1 RESULTADOS DE SIMULACIÓN

4.2 RESULTADOS EN TIEMPO REAL

4.3 CONCLUSIONES

Para comprobar la validez del programa de identificación paramétrica discreta *"IPD.EXE"* se realizaron numerosas pruebas a nivel de simulación y en tiempo real en las cuales se trató de cubrir la mayor parte de posibilidades que brinda el programa para identificar un sistema. Estas pruebas permitieron determinar la validez de los algoritmos desarrollados así como también a través de dichas pruebas se pudo realizar ciertos refinamientos en la presentación de resultados.

A continuación se presenta algunas de las pruebas más representativas de identificación de sistemas a nivel de simulación y en tiempo real, para cada uno de los métodos de identificación desarrollados que son: mínimos cuadrados recursivos, mínimos cuadrados extendidos y máximo de Likelihood. A través de estas pruebas se pretende indicar las múltiples capacidades del programa para cambiar condiciones en la identificación de sistemas.

4.1 RESULTADOS DE SIMULACION

A nivel de simulación se presenta pruebas de identificación con modelos de primer, segundo y tercer orden, utilizando distintos tipos de entrada entre las que se tiene:

- Entrada escalón sumada ruido de tipo randómico, PRBS, y estadístico.
- Entrada solamente de ruido tal como randómico, PRBS y estadístico.

En simulación se encuentra disponible la opción de sumar ruido a la salida en un porcentaje de su valor final, este

ruido puede ser blanco o correlacionado y de cualquiera de las tres clase de ruido (randómico, PRBS, estadístico).

El programa desarrollado genera un reporte numérico de la identificación realizada, así como también permite una clara visualización de la convergencia de los parámetros, sin embargo si se desea también se puede grabar en un archivo *<nombre>.ipd* toda la información paso a paso de como van evolucionando los parámetros a lo largo de la identificación, o como un archivo *<nombre>.dat* en el caso de que se desee almacenar la entrada y salida de ruido al sistema para en un momento posterior poder recuperar dicha información y continuar con la sesión de trabajo. Es necesario indicar que al abrir un archivo *<nombre.dat>* en el programa de identificación paramétrica discreta se debe tener las mismas condiciones en las que fue grabado este archivo y entre las que se tiene: tipo de entrada, tipo de salida, y número de iteraciones.

Estos archivos se los puede abrir en una hoja electrónica para realizar algún análisis extra o imprimir un reporte completo de la convergencia de los parámetros. Esta opción se utiliza para imprimir los resultados de las pruebas de simulación que se muestra a continuación.

4.1.1 SISTEMA DE PRIMER ORDEN (RUIDO BLANCO)

El sistema de primer orden que se simula y que se va a identificar esta dado por la ecuación de diferencias:

$$y(k)=0.4y(k-1)+1.2u(k-1)$$

en donde los parámetros verdaderos a identificar son:

$$\Theta(k)=\begin{bmatrix}0.4 & 1.2\end{bmatrix}^T$$

Este sistema tiene las siguientes características:
Entrada: escalón (1) + ruido randómico al 100%.
Salida: ruido blanco randómico 100%.

Debido a que en la salida tiene ruido blanco se puede identificar al sistema mediante el método de mínimos cuadrados recursivo, en la tabla 4.1.1 se presenta un reporte similar al generado por el programa "*IPD.EXE*" al identificar el sistema mediante este método.

MÍNIMOS CUADRADOS RECURSIVOS (MCR)
Identificación en simulación

Planta simulada		Modelo planteado	
Orden de A(z) = 1		Orden de A(z) = 1	
Orden de B(z) = 1		Orden de B(z) = 1	
Retardo = 1		Retardo = 1	
A(z)	B(z)	A(z)	B(z)
a1 = 0.4	b1 = 1.2	a1 = 0.39409	b1 = 1.2007

Entrada: Escalón (1) más ruido randómico (50%)
Ruido a la salida: Blanco randómico 100%
Número de iteraciones = 200

Tabla de resultados 4.1.1 IDENTIFICACIÓN POR MCR

A continuación se presenta los gráficos más relevantes en la identificación paramétrica del sistema.

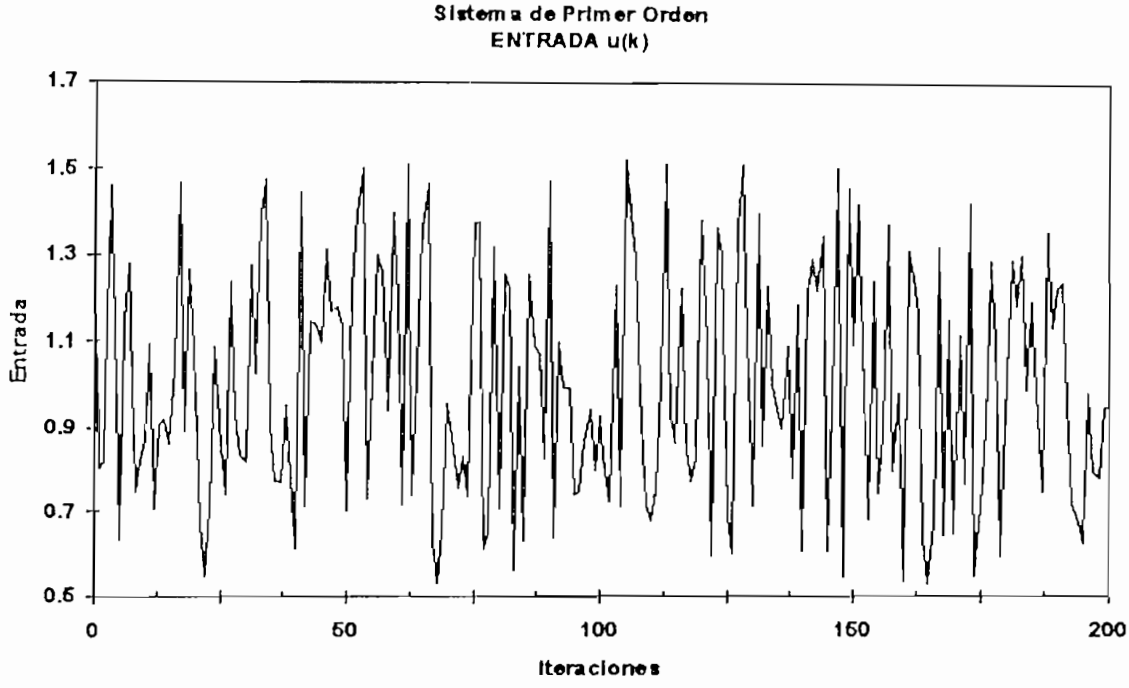

Fig. 4.1.1.1 Entrada al sistema

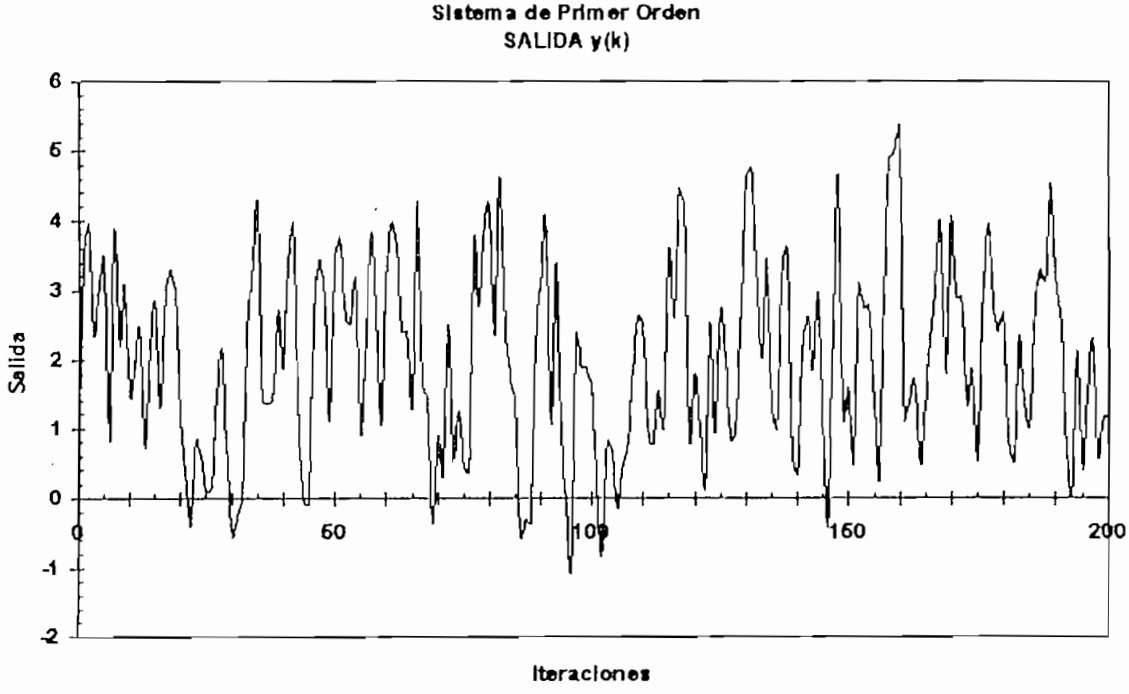

Fig. 4.1.1.2 Salida del sistema

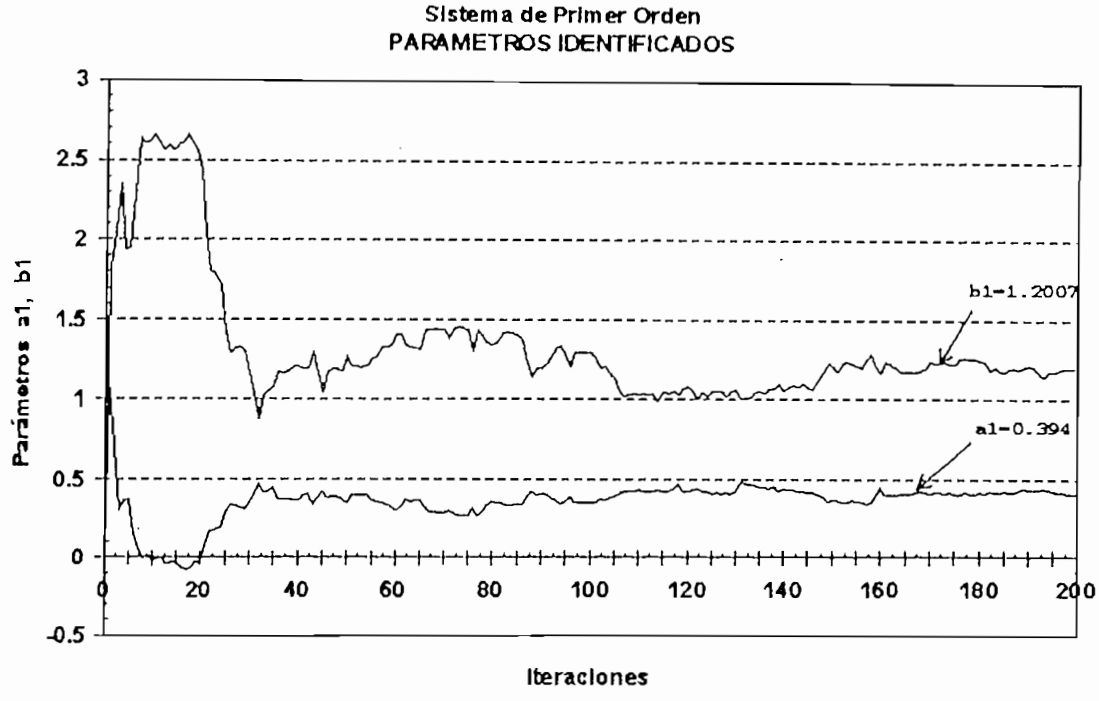

Fig. 4.1.1.3 Parámetros identificados

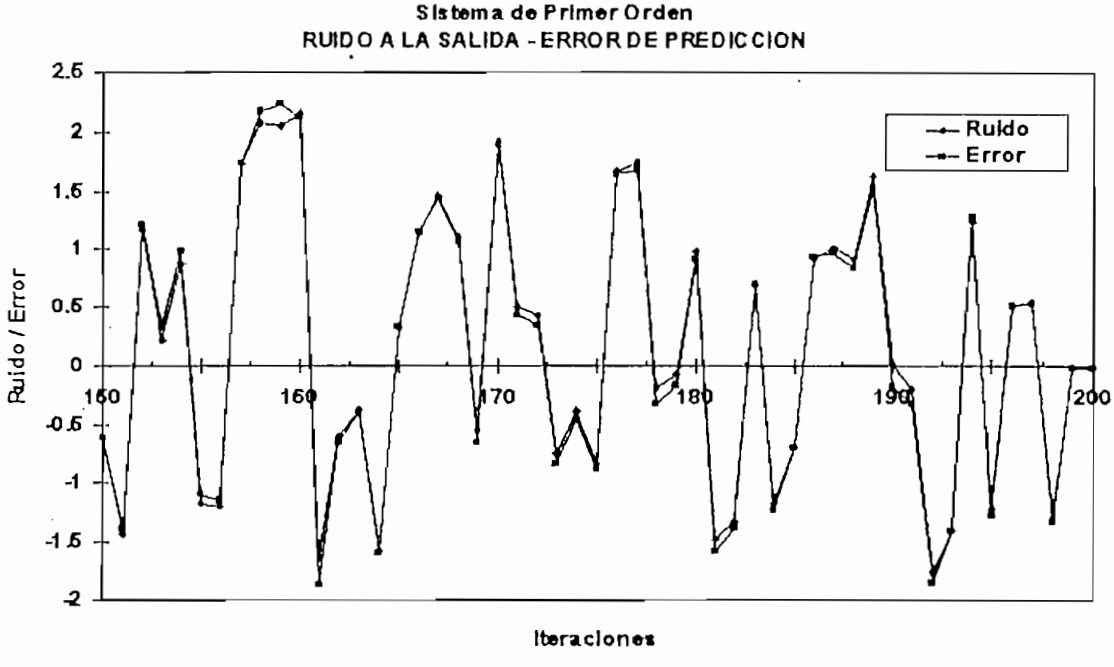

Fig. 4.1.1.4 Ruido a la salida y Error de predicción

4.1.2 SISTEMA DE SEGUNDO ORDEN (RUIDO BLANCO)

El sistema que se simula y que se va a identificar esta dado por la ecuación de diferencias:

$$y(k) = 1.2y(k-1) - 0.35y(k-2) + 0.2u(k-1) + 0.14u(k-2)$$

en donde los parámetros verdaderos a identificar son:

$$\Theta(k) = \begin{bmatrix} 1.2 & -0.35 & 0.2 & 0.14 \end{bmatrix}^T$$

Este sistema tiene las siguientes características:
Entrada: Ruido PRBS (2).
Ruido a la salida: Blanco estadístico 50%.

Al tener en la salida ruido blanco se utiliza el método de mínimos cuadrados recursivos. En la tabla de resultados 4.1.2 se presenta un reporte similar al generado por el programa de identificación.

MÍNIMOS CUADRADOS RECURSIVOS (MCR)
Identificación en simulación

Planta simulada		Modelo planteado	
Orden de A(z) = 2		Orden de A(z) = 2	
Orden de B(z) = 2		Orden de B(z) = 2	
Retardo = 1		Retardo = 1	
A(z)	B(z)	A(z)	B(z)
a1 = 1.2	b1 = 0.2	a1 = 1.202	b1 = 0.192
a2 = -0.35	b2 = 0.14	a2 = -0.356	b2 = 0.135

Entrada: Ruido PRBS (2)
Ruido a la salida: Blanco estadístico 50%
Número de iteraciones = 300

Tabla de resultados 4.1.2 IDENTIFICACIÓN POR MCR

A continuación se presenta los gráficos más importantes en la identificación del sistema.

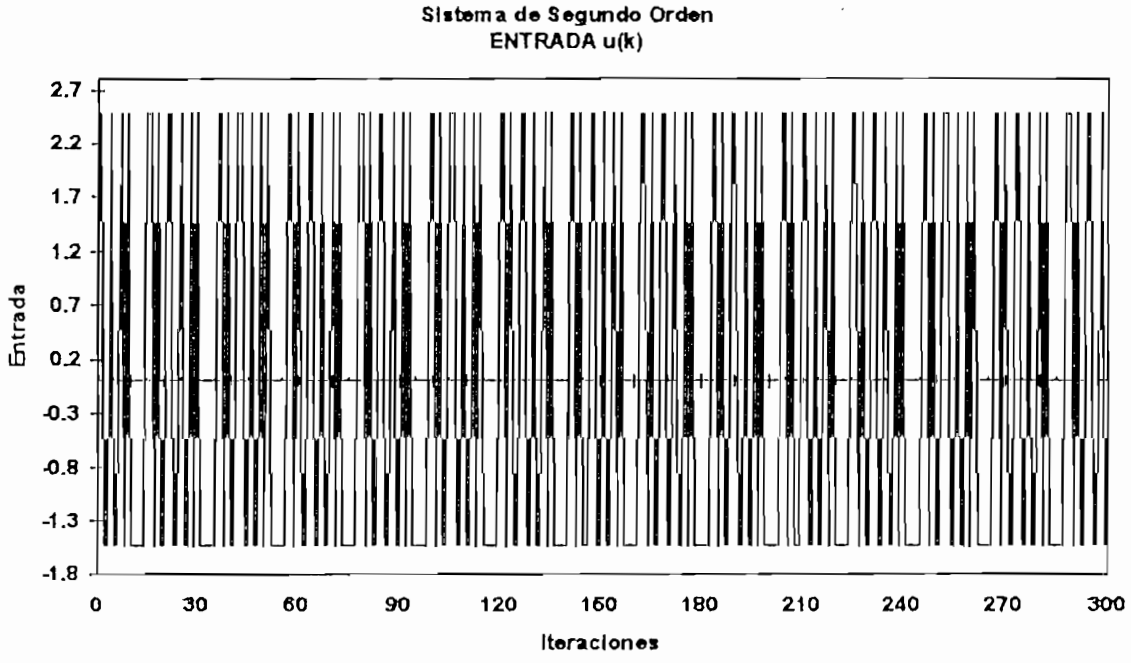

Fig. 4.1.2.1 Entrada al sistema

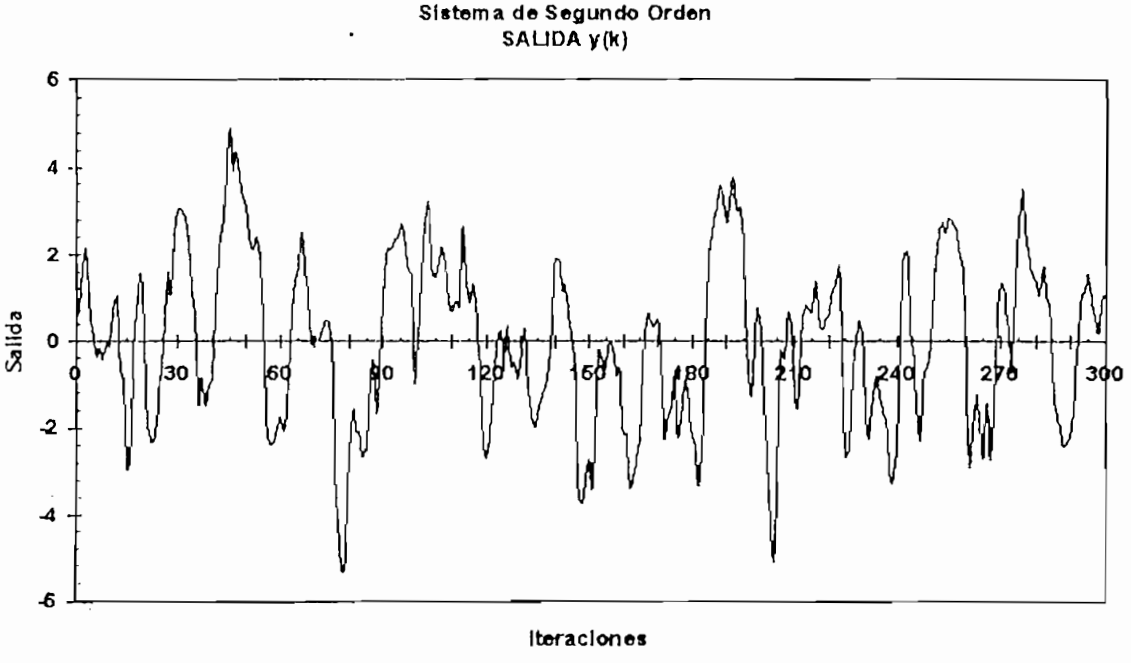

Fig. 4.1.2.2 Salida del sistema

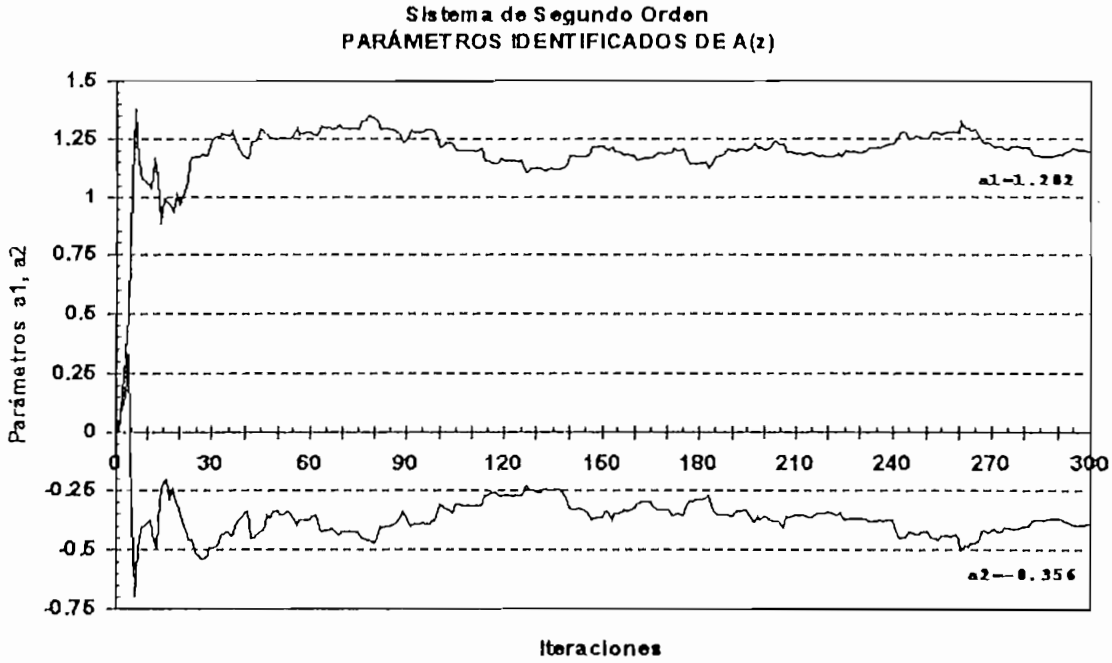

Fig. 4.1.2.3 Parámetros Identificados del polinomio A(z)

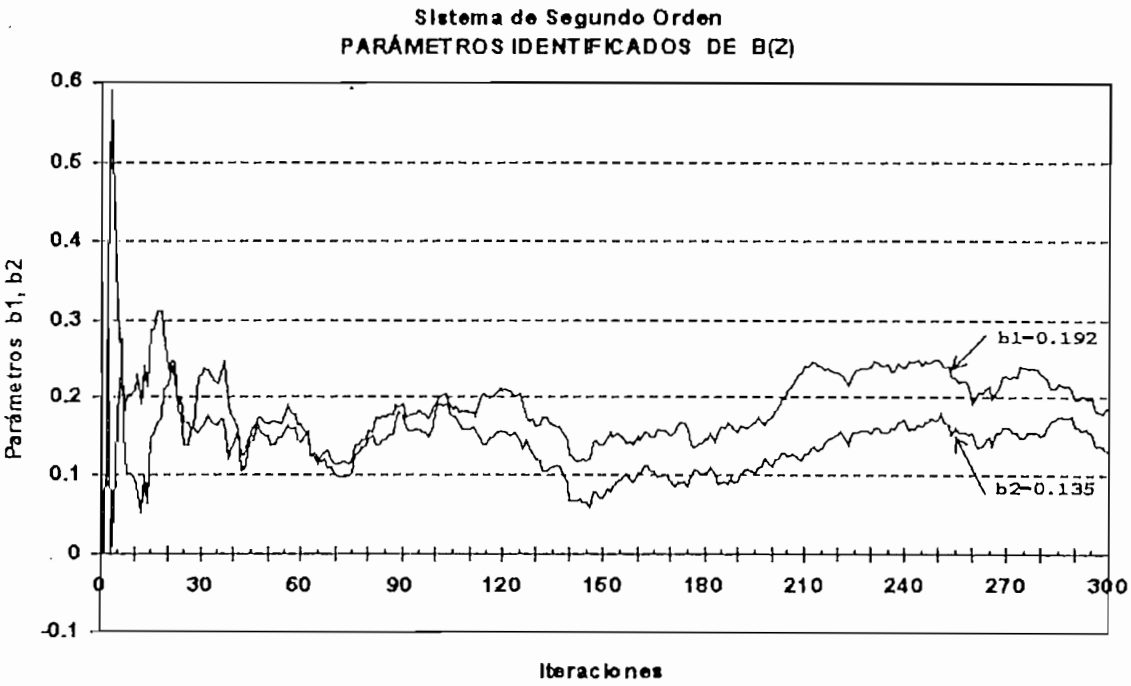

Fig. 4.1.2.4 Parámetros identificados del polinomio B(z)

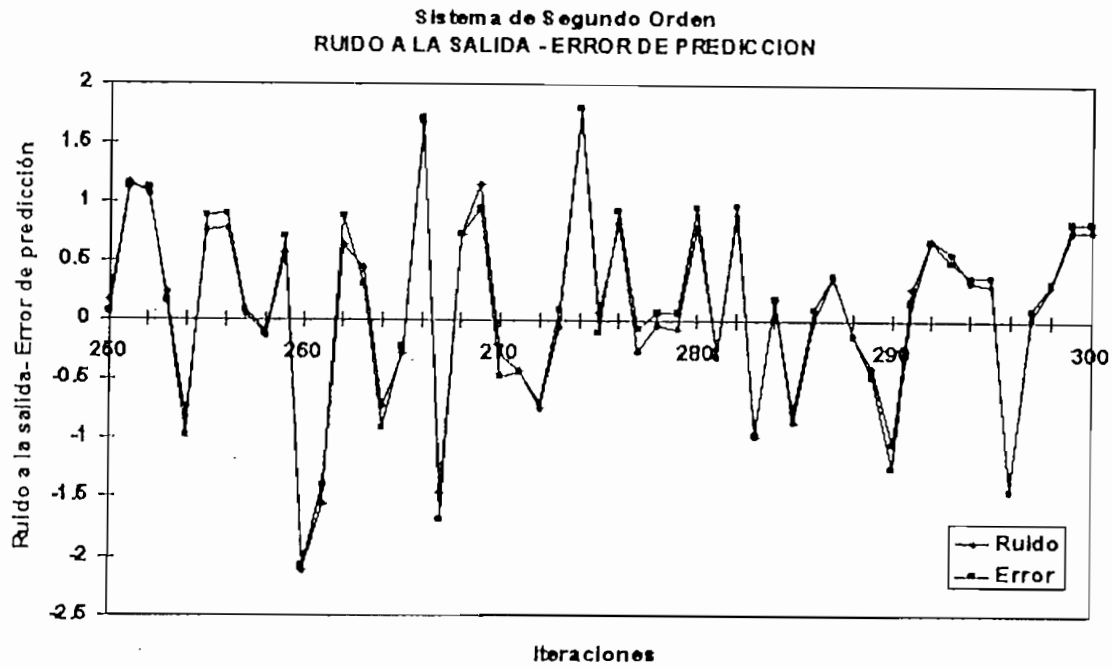

Fig. 4.1.2.5 Ruido blanco a la salida y Error de predicción

4.1.3 SISTEMA DE TERCER ORDEN (RUIDO BLANCO)

El sistema que se simula y que se va a identificar esta dado por la ecuación de diferencias:

$$y(k)=1.2y(k-1)-0.44y(k-2)+0.05y(k-3)+0.35u(k-1)+0.2u(k-2)$$

en donde los parámetros reales son:

$$\Theta(k)=\begin{bmatrix}1.2 & -0.44 & 0.05 & 0.35 & 0.25\end{bmatrix}^T$$

Este sistema tiene las siguientes características:
Entrada: Ruido randómico de amplitud 2.
Ruido a la salida: Ruido blanco estadístico 25%.

Al tener la salida ruido blanco se puede utilizar el método de mínimos cuadrados recursivos. En la tabla de resultados 4.1.3 se presenta el reporte generado por el programa de identificación.

MÍNIMOS CUADRADOS RECURSIVOS (MCR)
Identificación en simulación

Planta simulada		Modelo planteado	
Orden de A(z) = 3		Orden de A(z) = 3	
Orden de B(z) = 2		Orden de B(z) = 2	
Retardo = 1		Retardo = 1	
A(z)	B(z)	A(z)	B(z)
a1 = 1.2	b1 = 0.35	a1 = 1.16	b1 = 0.327
a2 = -0.44	b2 = 0.2	a2 = -0.40	b2 = 0.21
a3 = 0.05		a3 = 0.043	

Entrada: Ruido randómico (2)
Ruido a la salida: Blanco estadístico 25%
Número de iteraciones = 500

Tabla de resultados 4.1.3 IDENTIFICACIÓN POR MCR

A continuación se presenta los gráficos más importantes en el proceso de identificación.

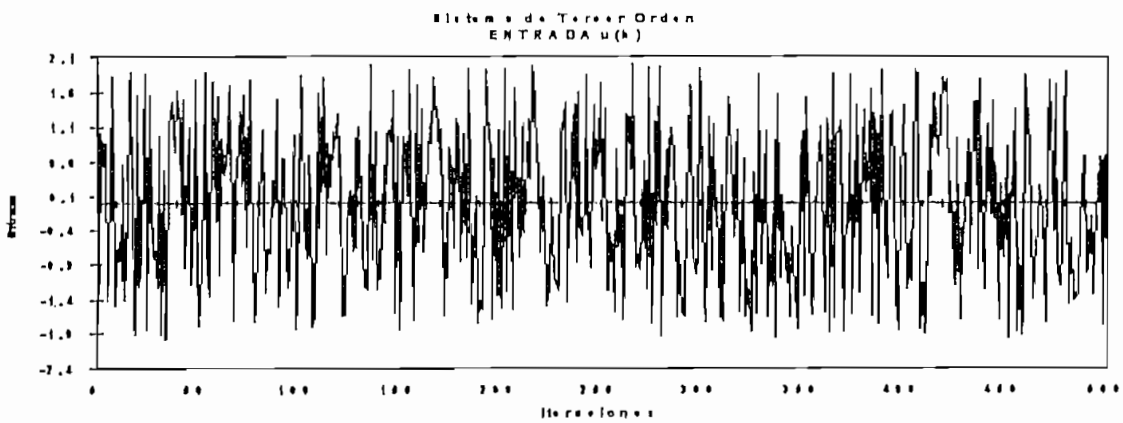

Fig. 4.1.3.1. Entrada al sistema

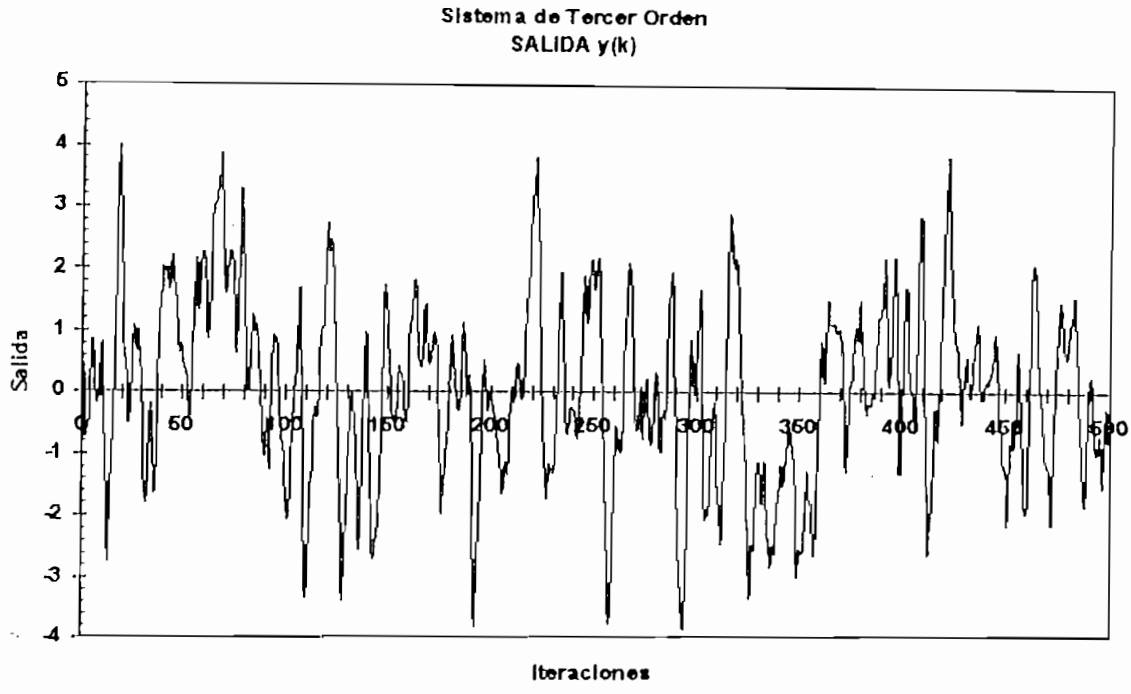

Fig. 4.1.3.2 Salida del sistema

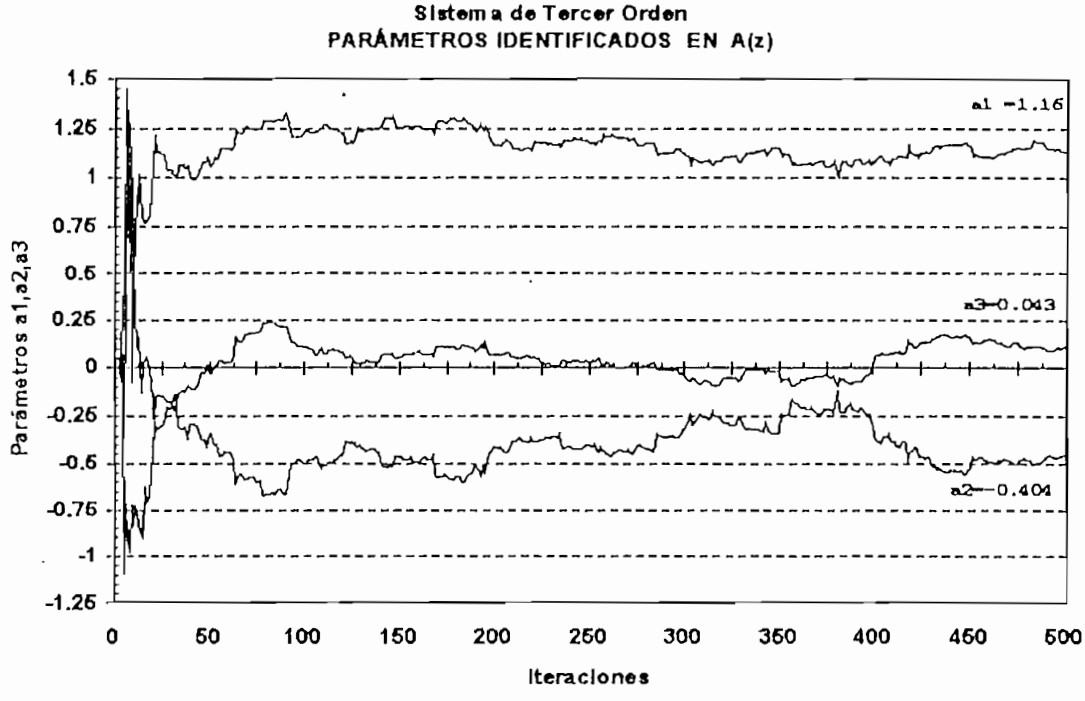

Fig. 4.1.3.3 Parámetros Identificados del polinomio A(z)

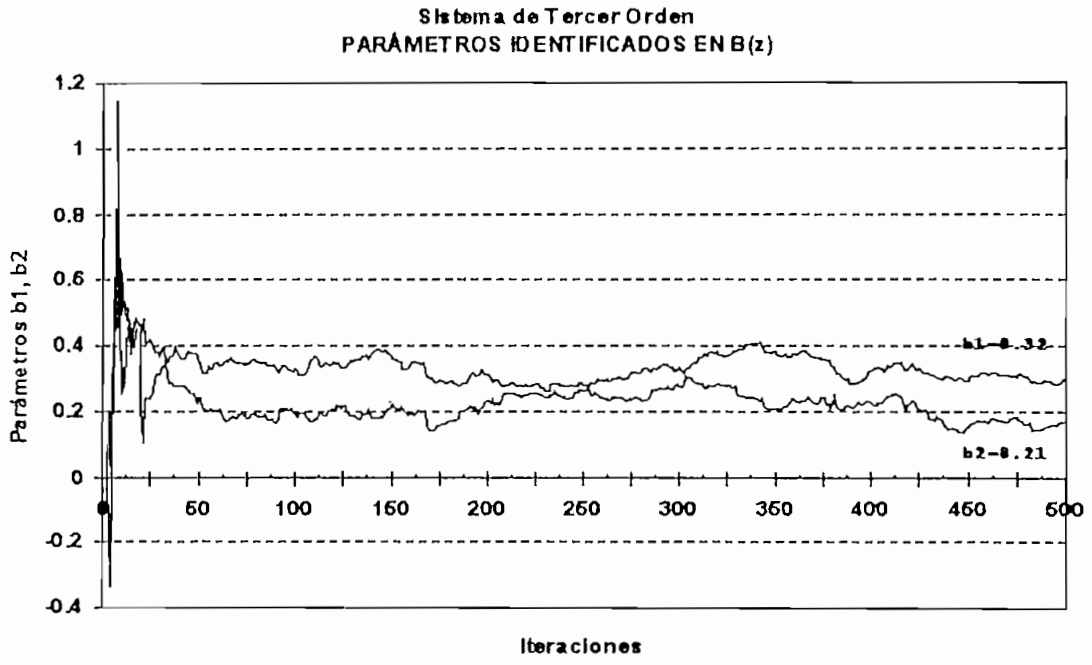

Fig. 4.1.3.4 Parámetros Identificados del polinomio B(z)

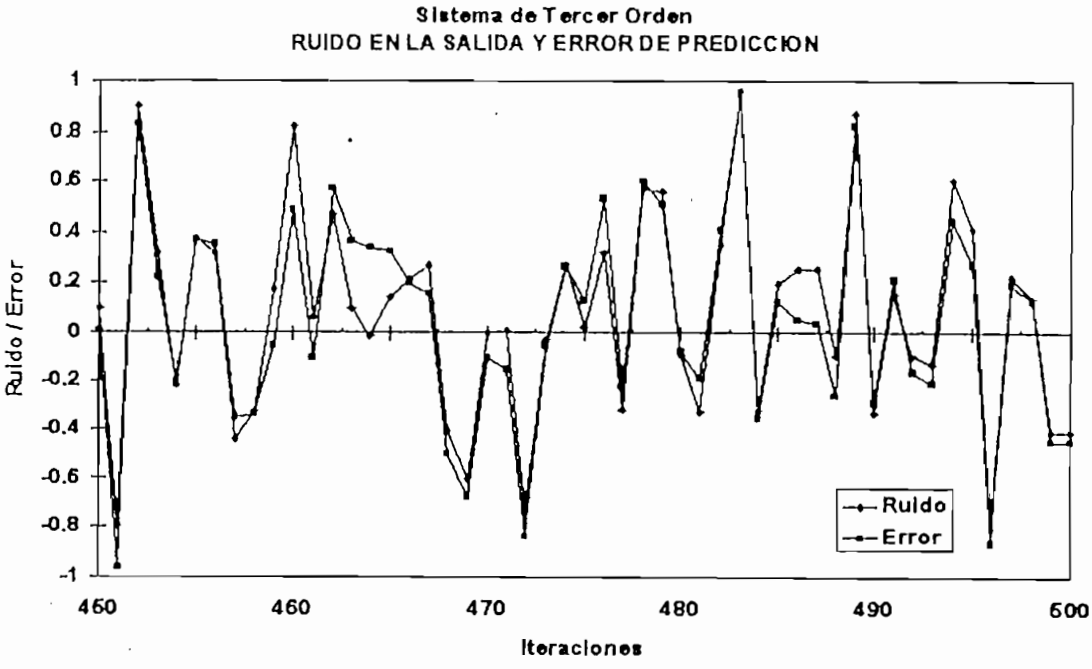

Fig. 4.1.3.5 Ruido y Error de predicción

4.1.4 SISTEMA DE PRIMER ORDEN (RUIDO CORRELACIONADO)

El sistema de primer orden que se simula y se identifica esta dado por la ecuación de diferencias:

$$y(k) = 0.4y(k-1) + 1.2u(k-2) + v(k) + 0.21v(k-1)$$

en donde los parámetros reales a identificar son:

$$\Theta(k)=\begin{bmatrix}0.4 & 1.2 & 0.21\end{bmatrix}^T$$

Este sistema tiene las siguientes características:
Entrada: escalón (1) + ruido randómico 100%.
Ruido a la salida: Ruido randómico correlacionado al 100%.

Aquí por ser el primer ejemplo que tiene la presencia de ruido correlacionado se presenta una tabla de resultados al utilizar el método de mínimos cuadrados recursivos para identificar el sistema. En tanto que se presenta toda la información cuando la identificación se la hace por medio del método de mínimos cuadrados extendidos y máximo de Likelihood.

MÍNIMOS CUADRADOS RECURSIVOS (MCR)
Identificación en simulación

Planta simulada	Modelo planteado
Orden de A(z) = 1	Orden de A(z) = 1
Orden de B(z) = 1	Orden de B(z) = 1
Orden de C(z) = 1	Retardo = 1
Retardo = 1	

A(z)	B(z)	C(z)	A(z)	B(z)
a1 = 0.4	b1 = 1.2	c1=0.21	a1 = 0.510	b1 = 1.0527

Entrada: Escalón (1) más ruido randómico (100%)
Ruido a la salida: Correlacionado randómico 100%
Número de iteraciones = 400

Tabla de resultados 4.1.4.1 por IDENTIFICACIÓN POR MCR

Siempre que se tiene ruido correlacionado a la salida se debe utilizar el método de mínimos cuadrados extendido como una primera aproximación para encontrar los parámetros y el método de máximo de Likelihood para estimar parámetros bajo la influencia de ruido correlacionado en un sistema.

A continuación se presenta los resultados de aplicar el método de mínimos cuadrados extendidos en la tabla de resultados 4.1.4.2.

MÍNIMOS CUADRADOS EXTENDIDOS (MCE)
Identificación en simulación

Planta simulada			Modelo planteado		
Orden de A(z) = 1			Orden de A(z) = 1		
Orden de B(z) = 1			Orden de B(z) = 1		
Orden de C(z) = 1			Orden de C(z) = 1		
Retardo = 1			Retardo = 1		
A(z)	B(z)	C(z)	A(z)	B(z)	C(z)
a1 = 0.4	b1 = 1.2	c1 = 0.21	a1=0.439	b1=1.16	c1=0.197

Entrada: Escalón (1) más ruido randómico (100%)
Ruido a la salida: Correlacionado randómico 100%
Número de iteraciones = 400

Tabla de resultados 4.1.4.2 IDENTIFICACIÓN POR MCE

A continuación se presenta los gráficos más importantes en la identificación del sistema.

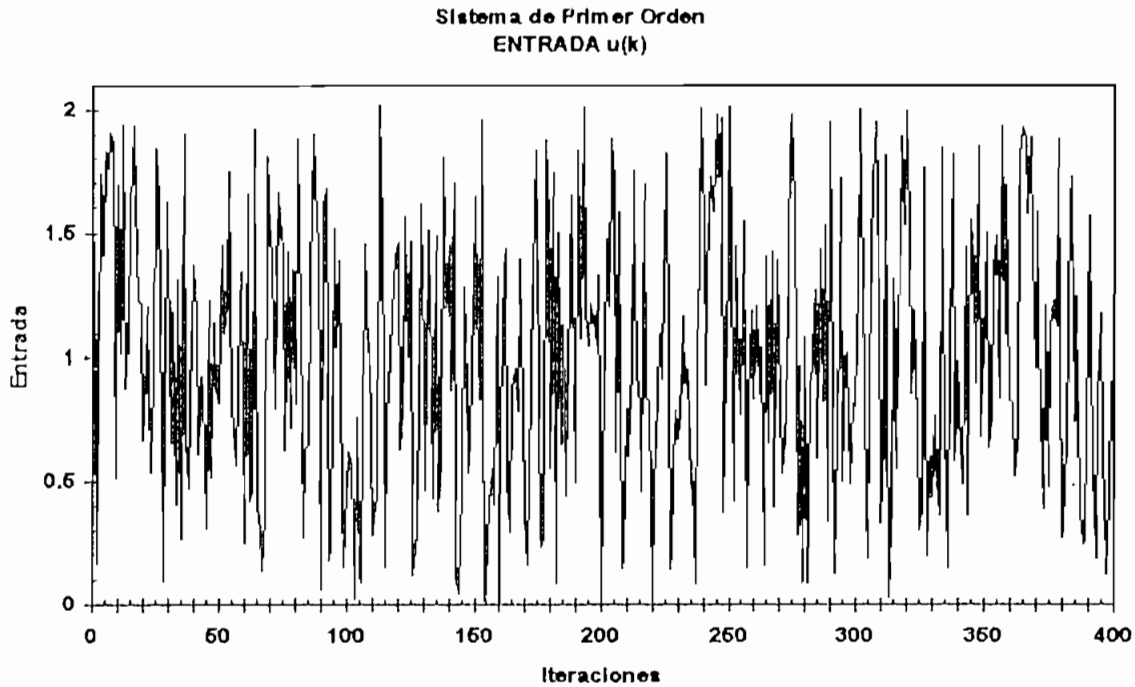

Fig. 4.1.4.1 Entrada al sistema

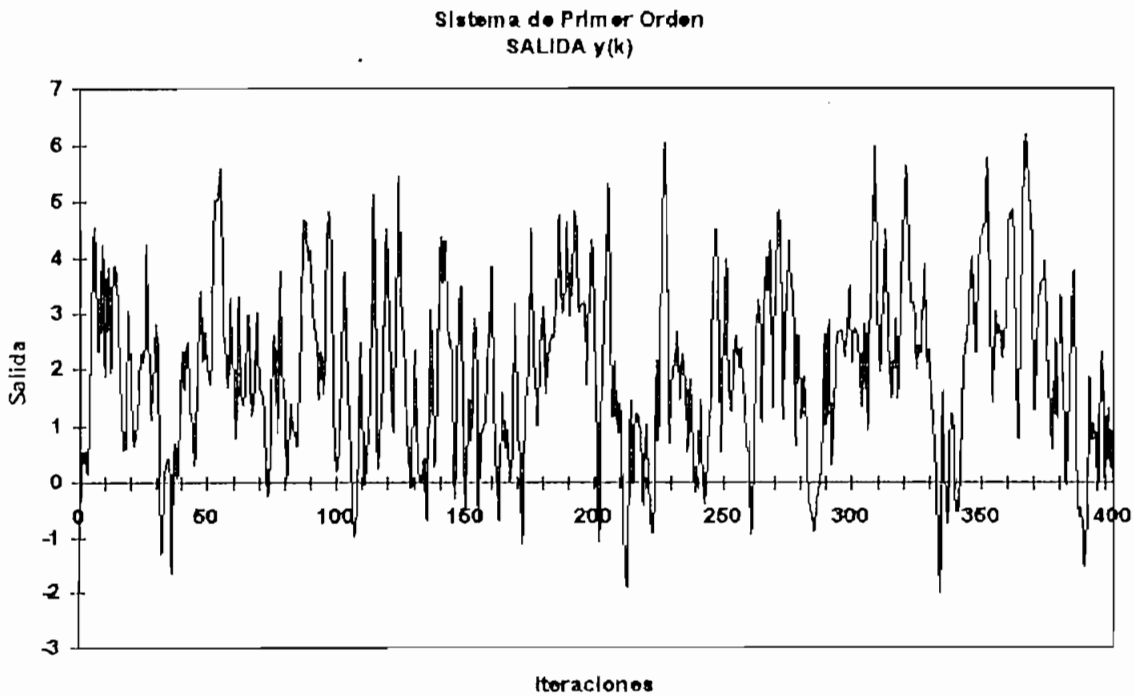

Fig. 4.1.4.2 Salida del sistema

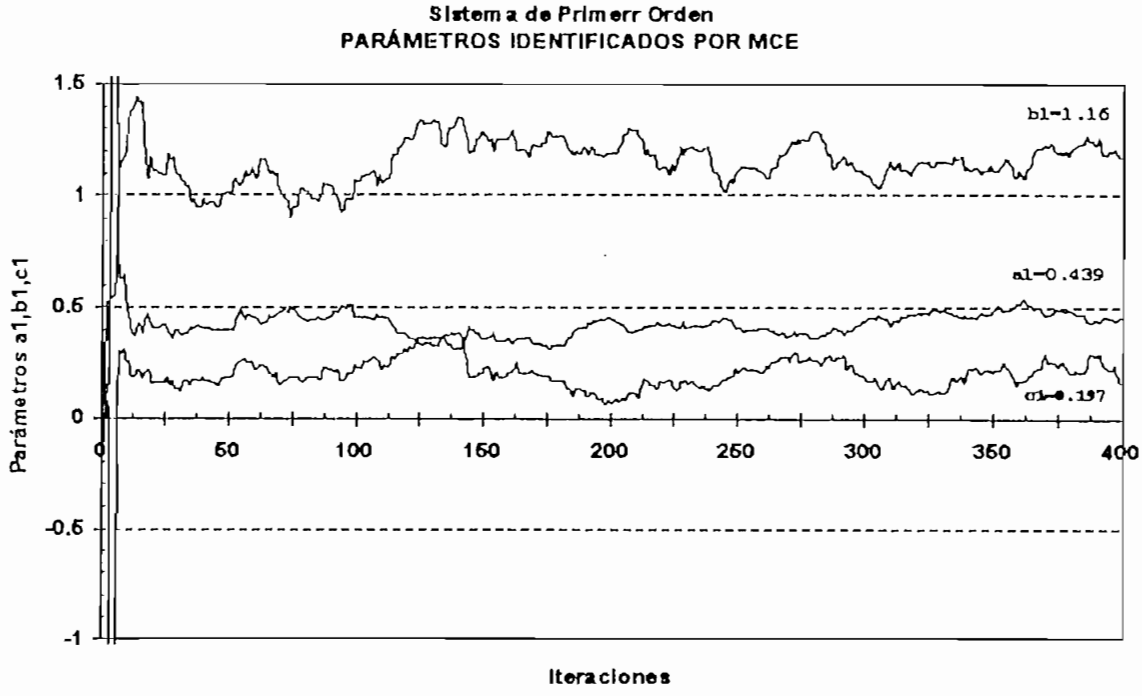

Fig. 4.1.4.3 Parámetros Identificados

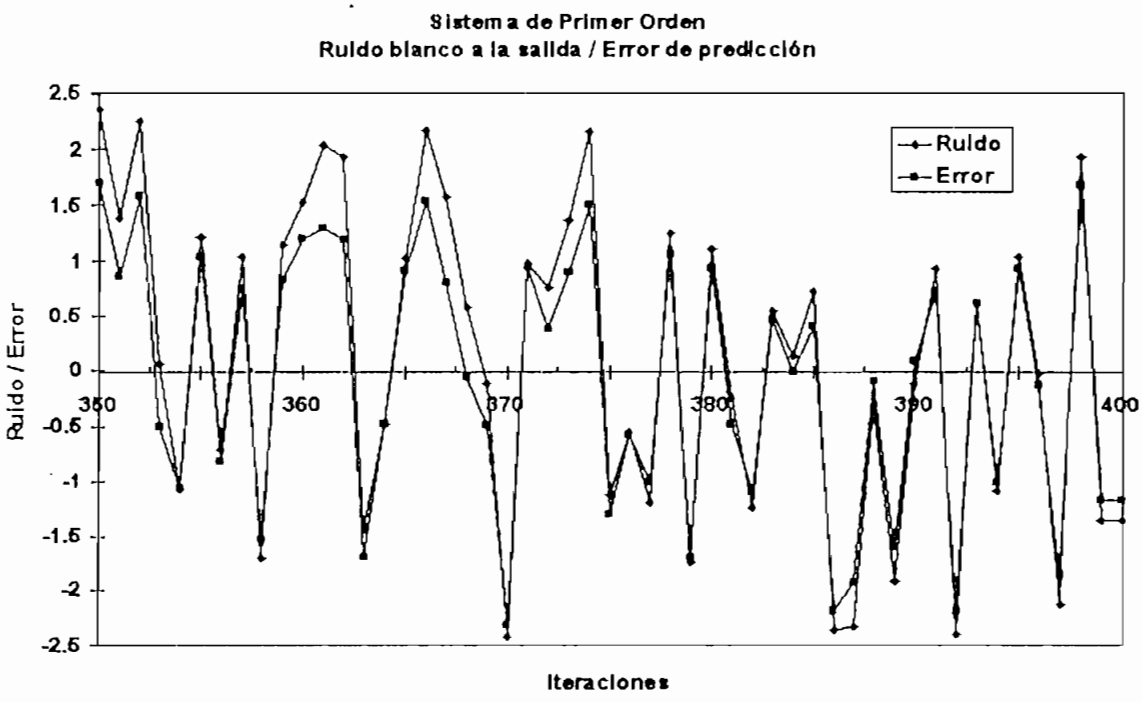

Fig. 4.1.4.4 Ruido blanco a la Salida y Error de Predicción

Al tener el sistema con ruido correlacionado se utiliza el método de máximo de Likelihood para identificarlo y se tiene el siguiente reporte de la tabla 4.1.4.3.

MÁXIMO DE LIKELIHOOD (MLH)
Identificación en simulación

Planta simulada			Modelo planteado		
Orden de A(z) = 1			Orden de A(z) = 1		
Orden de B(z) = 1			Orden de B(z) = 1		
Orden de C(z) = 1			Orden de C(z) = 1		
Retardo = 1			Retardo = 1		
A(z)	B(z)	C(z)	A(z)	B(z)	C(z)
a1 = 0.4	b1 = 1.2	c1 = 0.21	a1=0.423	b1=1.17	c1=0.217

Entrada: Escalón (1) más ruido randómico (100%)
Ruido a la salida: Correlacionado randómico 100%
Número de iteraciones = 400

Tabla de resultados 4.1.4.3 IDENTIFICACIÓN POR MLH

A continuación se presenta los gráficos más importantes en el proceso de identificación.

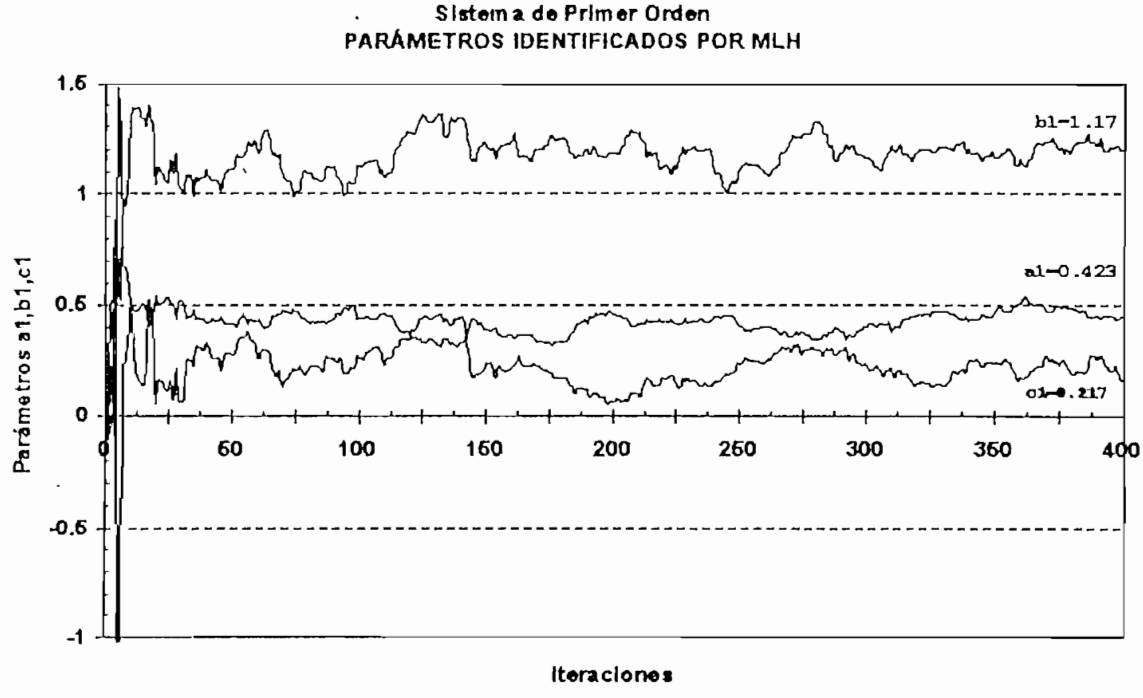

Fig. 4.1.4.5. Parámetros Identificados

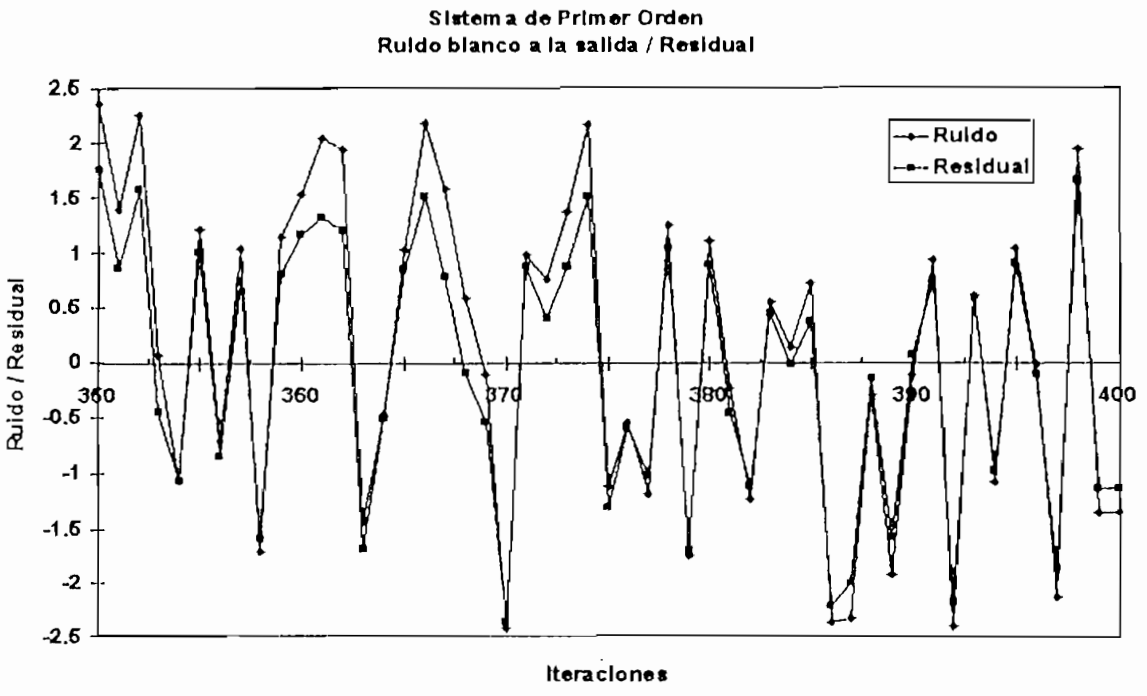

Fig. 4.1.4.6. Residual y Ruido blanco a la Salida

4.1.5 SISTEMA DE SEGUNDO ORDEN (RUIDO CORRELACIONADO)

El sistema de segundo orden que se simula y que se va a identificar esta dado por la ecuación de diferencias:

$$y(k)=1.2y(k-1)-0.35y(k-2)+0.2(k-3)+0.14u(k-2)+v(k)+0.9v(k-1)+0.14v(k-2)$$

en donde los parámetros a identificar son:

$$\Theta(k)=\begin{bmatrix}1.2 & -0.35 & 0.2 & 0.14 & 0.9 & 0.14\end{bmatrix}^T$$

Este sistema tiene las siguientes características:
Entrada: Ruido randómico (2)
Ruido a la salida: Ruido correlacionado estadístico 50%.

Fig. 4.1.5.1 Parámetros Identificados del polinomio A(z)

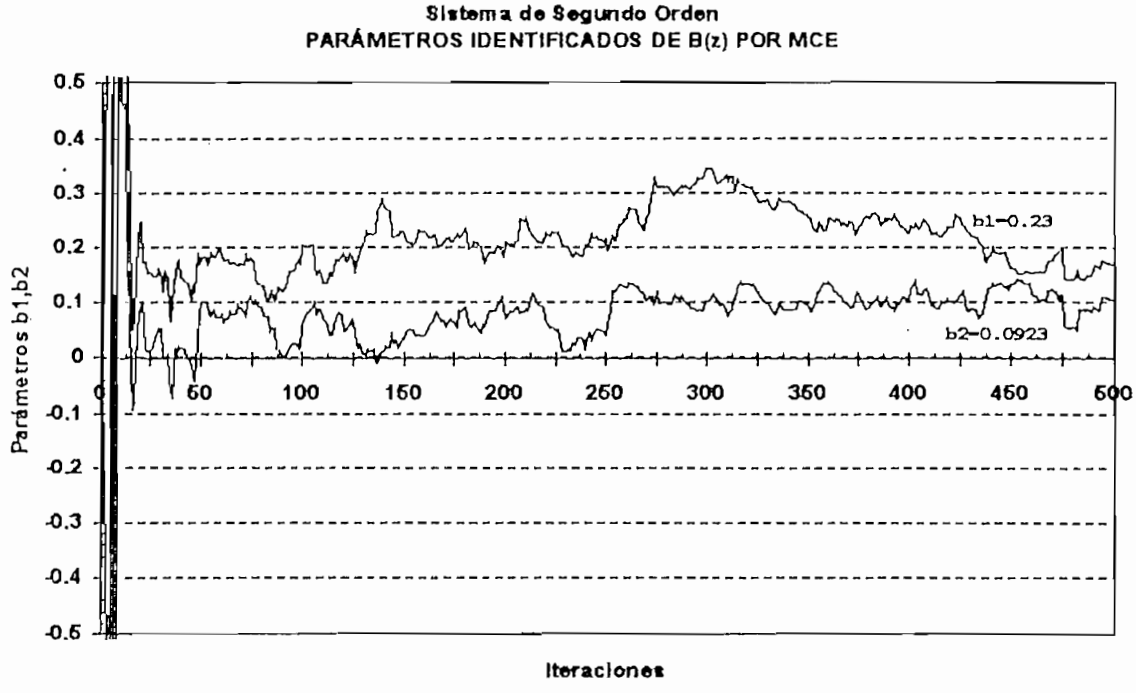

Fig. 4.1.5.2 Parámetros Identificados del polinomio B(z)

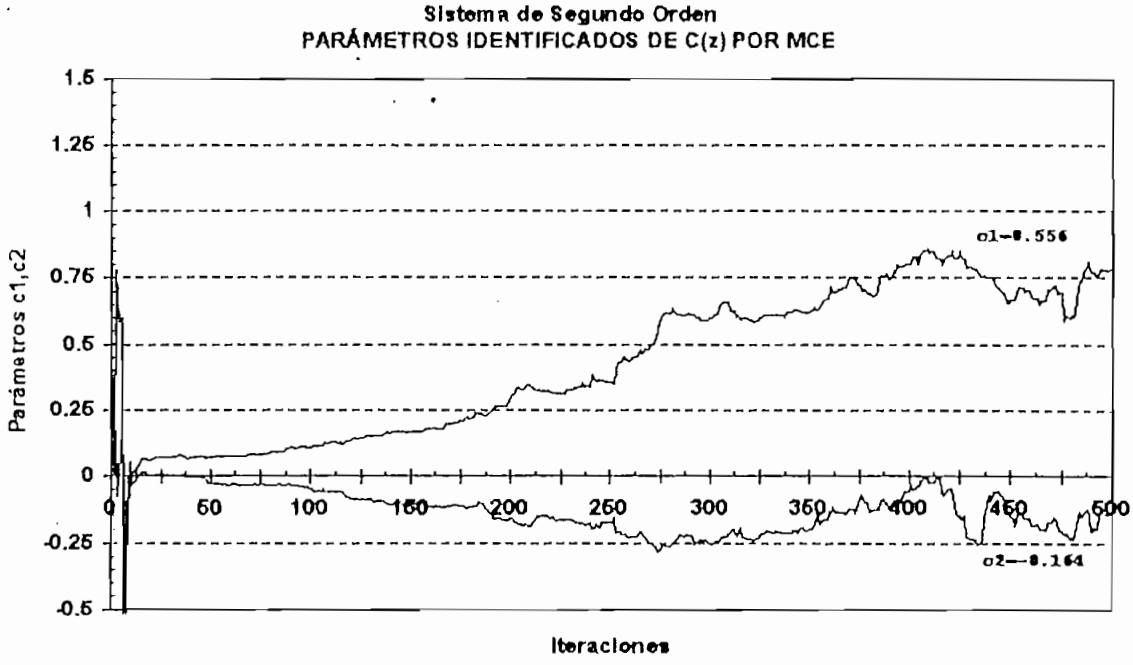

Fig. 4.1.5.3 Parámetros Identificados del polinomio C(z)

A continuación se presenta una tabla de resultados 4.1.5.2 que resulta de identificar el sistema mediante el método de máximo de Likelihood.

MÁXIMO DE LIKELIHOOD (MLH)
Identificación en simulación

Planta simulada			Modelo planteado		
Orden de A(z) = 2			Orden de A(z) = 2		
Orden de B(z) = 2			Orden de B(z) = 2		
Orden de C(z) = 2			Orden de C(z) = 2		
Retardo = 1			Retardo = 1		
A(z)	B(z)	C(z)	A(z)	B(z)	C(z)
a1 = 1.2	b1 = 0.2	c1 = 0.9	a1=1.2	b1=0.224	c1=0.85
a2 =-0.35	b2 = 0.14	c2 = 0.14	a2=-0.369	b2=0.151	c2=0.15

Entrada: Ruido randómico (2)
Ruido a la salida: Ruido correlacionado estadístico 50%
Número de iteraciones = 500

Tabla de resultados 4.1.5.2 IDENTIFICACIÓN POR MLH

A continuación se presenta los gráficos más importantes.

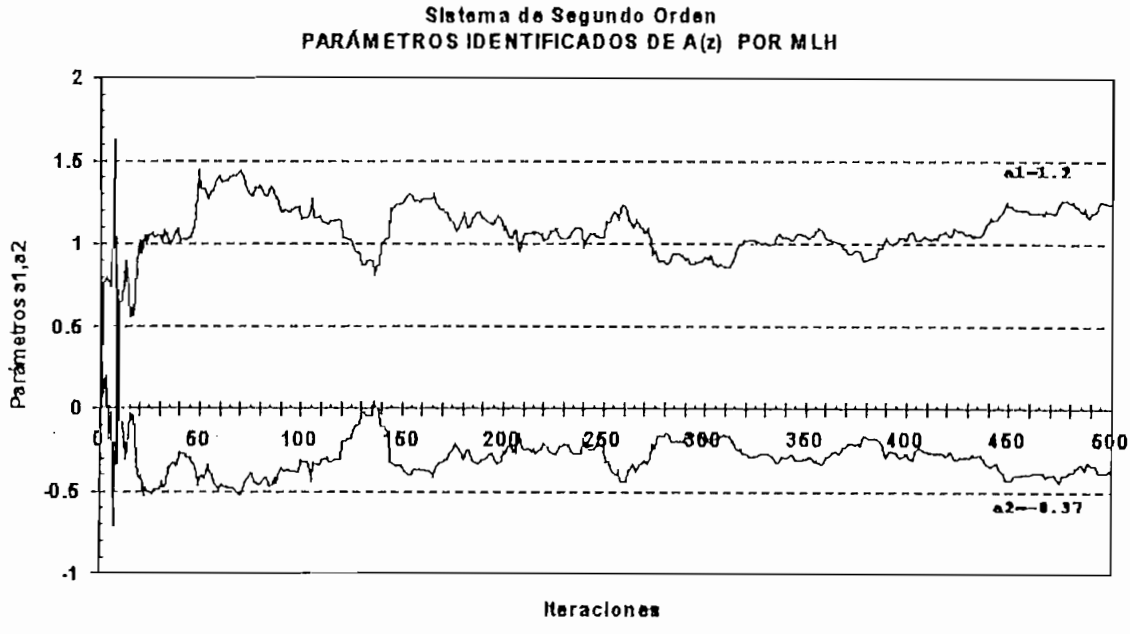

Fig. 4.1.5.4 Parámetros Identificados del polinomio A(z)

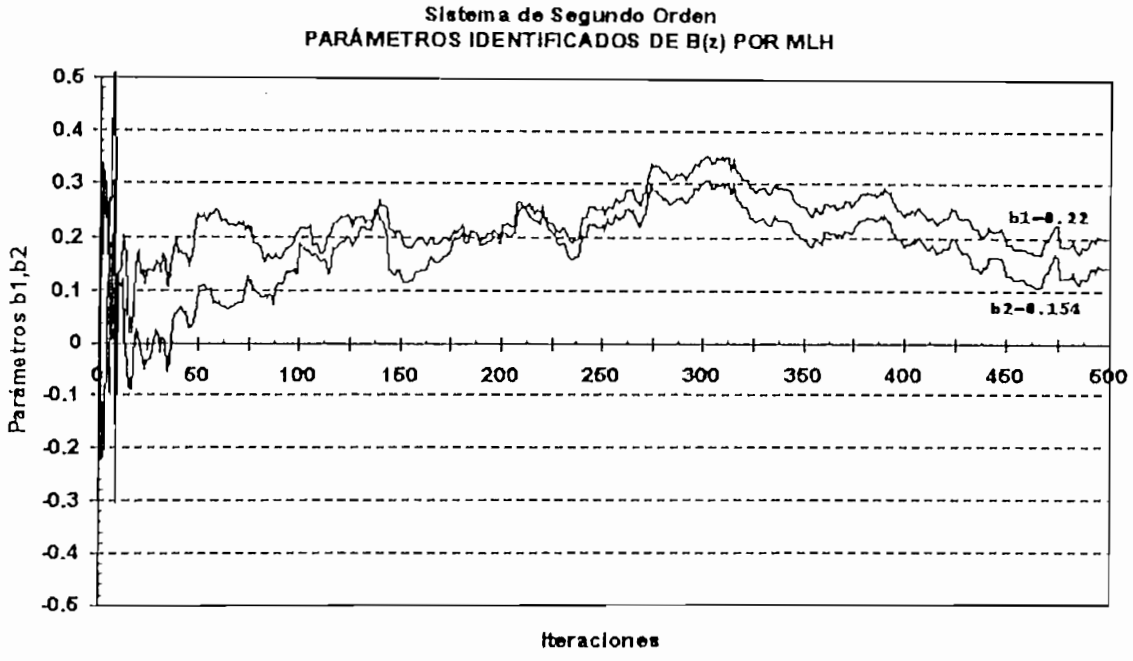

Fig. 4.1.5.5 Parámetros Identificados del polinomio B(z)

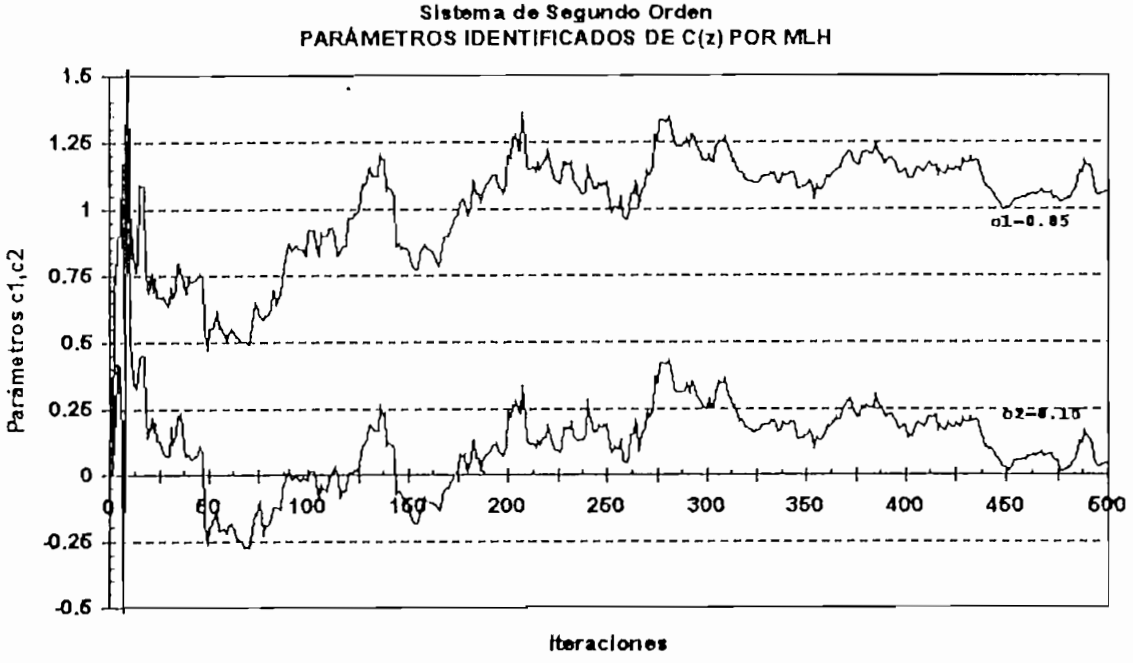

Fig. 4.1.5.6 Parámetros Identificados del polinomio C(z)

4.1.6 SISTEMA DE TERCER ORDEN (RUIDO CORRELACIONADO)

El sistema de tercer orden que se va a simular y que se identifica está dado por la ecuación de diferencias:

$$y(k)=-1.1y(k-1)+0.37y(k-2)+0.55y(k-3)+0.7u(k-1)+0.4u(k-2)+$$
$$+v(k)+0.9v(k-1)+0.14v(k-2)$$

en donde los parámetros a identificar son:

$$\Theta(k)=\begin{bmatrix}-1.1 & 0.37 & 0.55 & 0.7 & 0.4 & 0.9 & 0.14\end{bmatrix}^T$$

Este sistema tiene las siguientes características:
Entrada: Ruido PRBS de amplitud 2.
Ruido en la salida: Ruido correlacionado randómico al 50%.

A continuación se presenta un reporte de identificar el sistema mediante el método de mínimos cuadrados extendido.

MÍNIMOS CUADRADOS EXTENDIDOS (MCE)
Identificación en simulación

Planta simulada			Modelo planteado		
Orden de A(z) = 3			Orden de A(z) = 3		
Orden de B(z) = 2			Orden de B(z) = 2		
Orden de C(z) = 2			Orden de C(z) = 2		
Retardo = 1			Retardo = 1		
A(z)	B(z)	C(z)	A(z)	B(z)	C(z)
a1 = -1.1	b1 = 0.7	c1 = 0.9	a1=-0.99	b1= 0.733	c1 = 0.7
a2 = 0.37	b2 = 0.4	c2 = 0.14	a2= 0.411	b2= 0.33	c2 = 0.06
a3 = 0.55			a3= 0.512		

Entrada: Ruido PRBS (2)
Ruido a la salida: Correlacionado randómico 50%
Número de iteraciones = 800

Tabla de resultados 4.1.6.1 IDENTIFICACIÓN POR MCE

De la tabla se ve que se tiene desviación en los parámetros por lo que se identifica al sistema con el método de Likelihood. A continuación se presenta un reporte en la tabla 4.1.6.2.

MÁXIMO DE LIKELIHOOD (MLH)
Identificación en simulación

Planta simulada			Modelo planteado		
Orden de A(z) = 3			Orden de A(z) = 3		
Orden de B(z) = 2			Orden de B(z) = 2		
Orden de C(z) = 2			Orden de C(z) = 2		
Retardo = 1			Retardo = 1		
A(z)	B(z)	C(z)	A(z)	B(z)	C(z)
a1 = -1.1	b1 = 0.7	c1 = 0.9	a1=-1.09	b1= 0.716	c1= 0.83
a2 = 0.37	b2 = 0.4	c2 = 0.14	a2= 0.38	b2= 0.404	c2= 0.128
a3 = 0.55			a3= 0.552		

Entrada: Ruido PRBS (2)
Ruido a la salida: Correlacionado randómico 50%
Número de iteraciones = 800

Tabla de resultados 4.1.6.2 IDENTIFICACIÓN POR MLH

A continuación se presenta los gráficos más importantes.

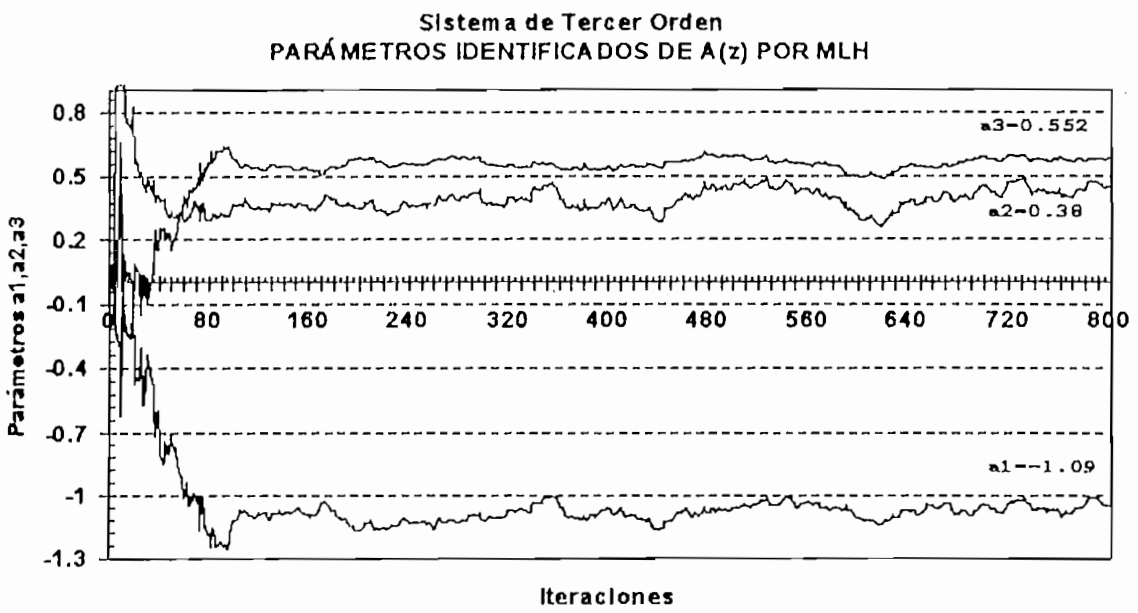

Fig. 4.1.6.1 Parámetros Identificados del polinomio A(z)

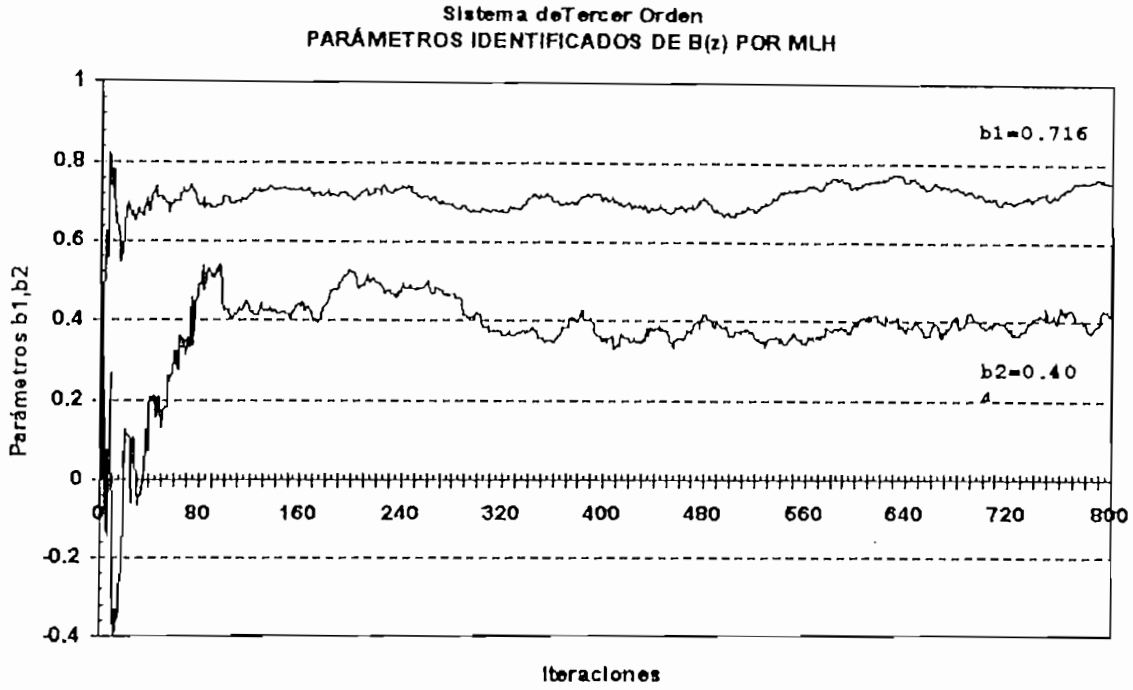

Fig. 4.1.6.2 Parámetros Identificados del polinomió B(z)

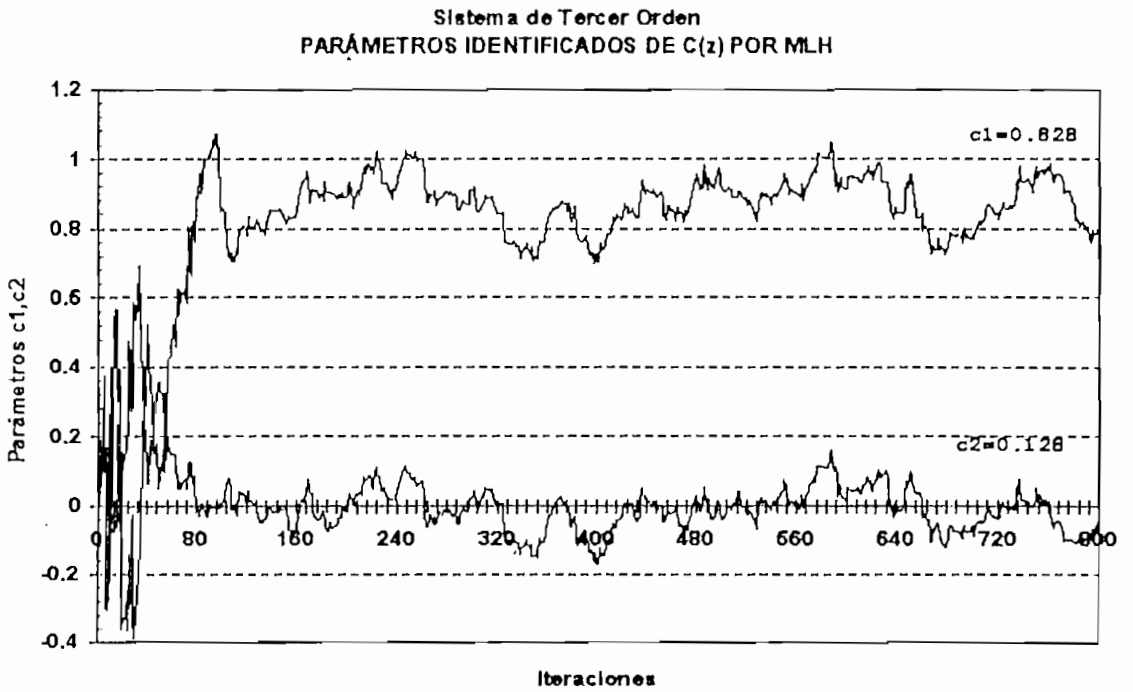

Fig. 4.1.6.3 Parámetros Identificados del polinomio C(z)

4.2 RESULTADOS EN TIEMPO REAL

Una forma muy valida de comprobar el funcionamiento del programa de computación es someterlo a identificar sistemas en tiempo real. Para este caso se identificará tres sistemas que son: 2 circuitos RC de primer y segundo orden así como también un prototipo de nivel de líquidos [6].

Los sistemas en tiempo real deben tener en general una constante de tiempo mayor al tiempo que se demora el programa en realizar una iteración de identificación, (este tiempo en una computadora COMPAQ - ProLinea a 66 Mhz es de 50 mseg), esto debido a que para muestrear adecuadamente un sistema es necesario tener un período de muestreo al menos 2 veces más pequeño que el período de la señal.

A continuación se presenta los resultados de cada prueba.

4.2.1 CIRCUITO RC DE PRIMER ORDEN

El circuito RC que se utilizó es el mostrado en la figura 4.2.1.1. Antes de presentar los resultados de la identificación, se procede a hallar los parámetros reales del sistema que se va a identificar, esto para tener la capacidad de comparar el funcionamiento del programa de identificación, pues en la mayoría de casos esta información es la que debe entregar el programa desarrollado.

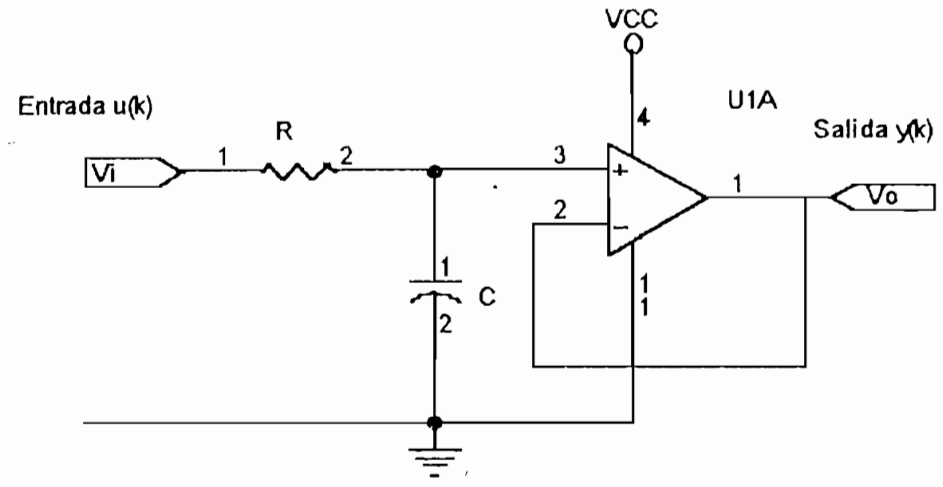

Fig. 4.2.1.1 Circuito RC de Primer Orden

Hallando la función de transferencia del sistema en el dominio s se tiene:

$$Vo(s)=\frac{\frac{1}{sC}}{\frac{1}{SC}+R}Vi(s)\text{; entonces } Vo(s)=\frac{1}{RCs+1}Vi(s)$$

$$\text{de donde: } G(s)=\frac{Vo(s)}{Vi(s)}=\frac{1}{RCs+1}$$

Para valores de:

$$C = 10\ \mu f$$

$$R = 1.2\ M\Omega$$

$$\text{se tiene que } G(s)=\frac{1}{12s+1}$$

Discretizando el modelo para obtener la ecuación de diferencias al período de muestreo T = 500 mseg, se tiene:

$$G(z)=\left(1-z^{-1}\right)\cdot z\left\{\frac{G(s)}{s}\right\}=\left(1-z^{-1}\right)\cdot z\left\{\frac{1}{s(12s+1)}\right\}$$

entonces:

$$G(z)=\frac{0.0408}{z-0.959}$$

por lo que la ecuación de diferencias es:

$$y(k) - 0.959y(k-1) = 0.0408u(k-1)$$

de donde los parámetros a identificar son:

$$\Theta(k) = \begin{bmatrix} 0.959 & 0.0408 \end{bmatrix}^T$$

El sistema presenta las siguientes características para su identificación en tiempo real.

Entrada: Ruido randómico de amplitud 2.
Ruido a la salida: Desconocido.

Identificando el sistema mediante el método de mínimos cuadrados recursivos en tiempo real se tiene un reporte similar al generado por el programa en la tabla 4.2.1.1.

```
MÍNIMOS CUADRADOS RECURSIVO (MCR)
Identificación en tiempo real
Modelo Planteado
Orden de A(z) = 1
Orden de B(z) = 1
Retardo = 1

A(z)              B(z)
a1 = 0.95132      b1 = 0.044389

Entrada: Ruido randómico (2)
Número de Iteraciones = 599
Período de muestreo = 500 [mseg]
```

Tabla de resultados 4.2.1.1 IDENTIFICACIÓN POR MCR

A continuación se muestra los gráficos más importantes en el proceso de identificación del sistema.

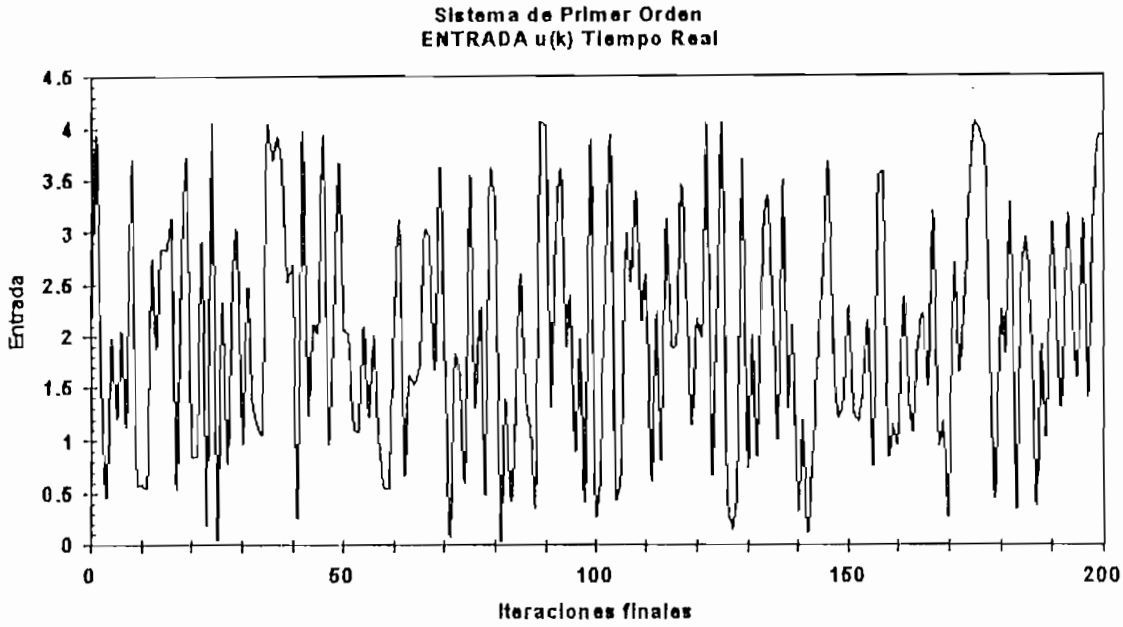

Fig. 4.2.1.2 Entrada al sistema

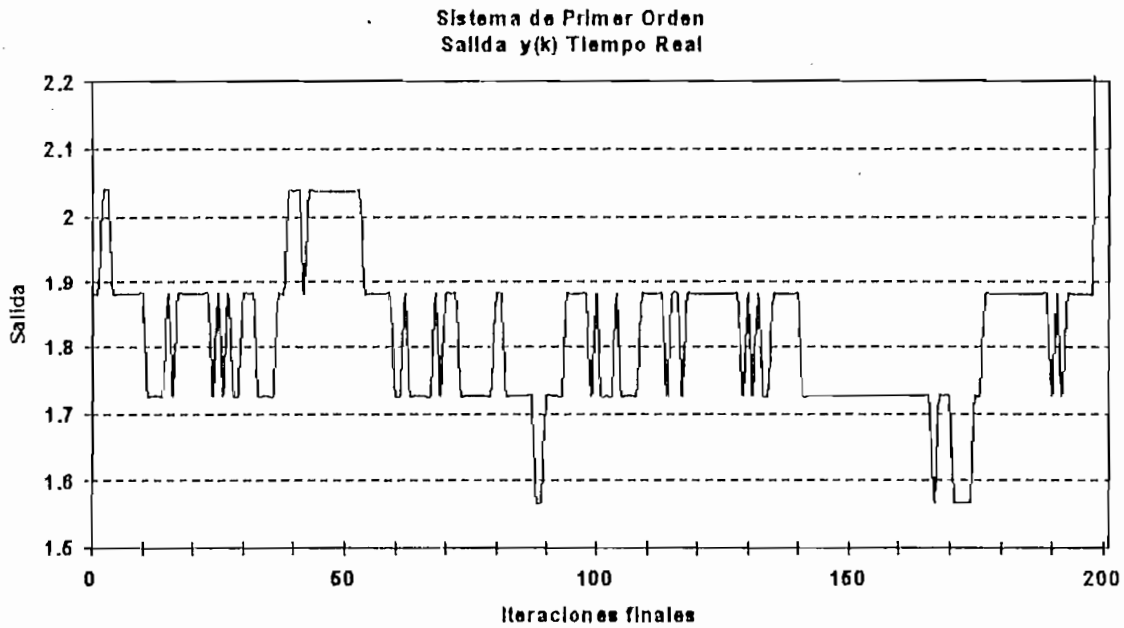

Fig. 4.2.1.3 Salida del sistema

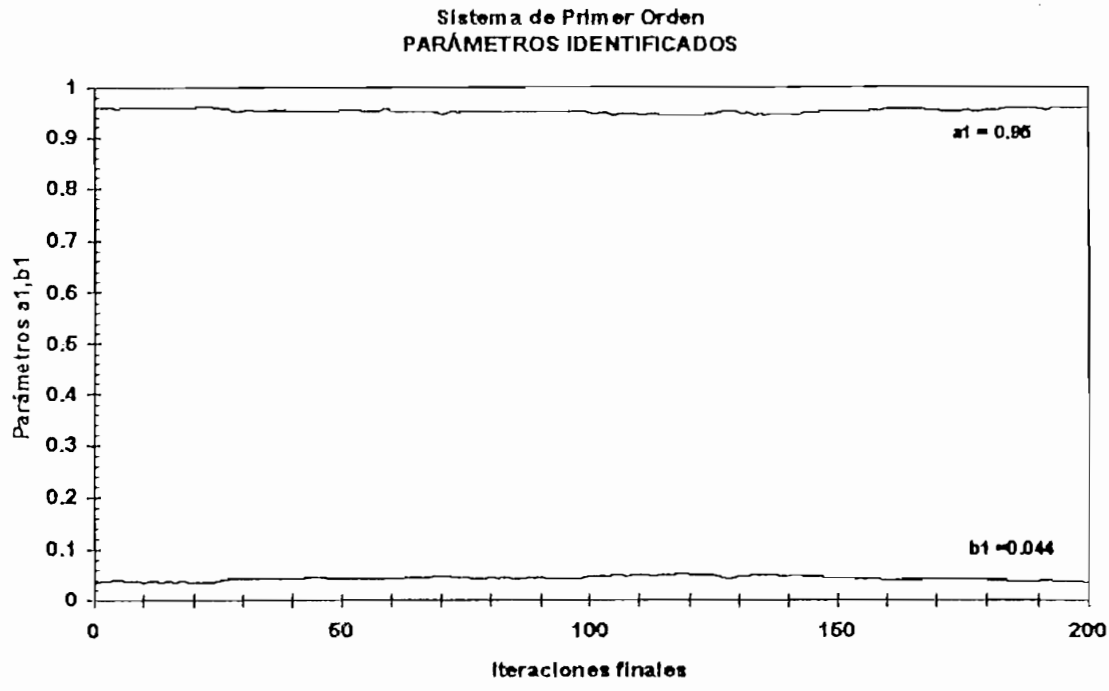

Fig. 4.2.1.4 Parámetros Identificados por MCR

4.2.2 CIRCUITO RC DE SEGUNDO ORDEN

El circuito RC de segundo orden que se utilizó en esta prueba es el mostrado en la figura 4.2.2.1. Al ser esta una prueba de análisis del funcionamiento del programa de identificación, se discretizó el sistema para un valor de período conocido y de esta manera tener la posibilidad de comparar los valores identificados, con el valor de los parámetros reales del sistema.

Para esta prueba se utiliza tanto el método de mínimos cuadrados recursivos, así como también el método de máximo de Likelihood.

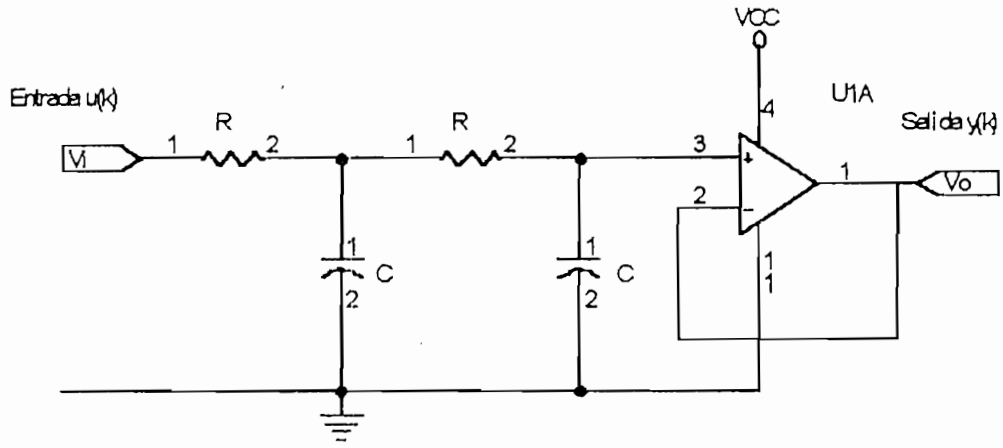

Fig. 4.2.2.1 Circuito RC de Segundo Orden

Hallando la función de transferencia del sistema en el dominio s se tiene:

$$G(s)=\frac{Vo(s)}{Vin(s)}=\frac{1}{R^2C^2s^2+3RCs+1}$$

Para valores de :

$$C = 10\ \mu f$$

$$R = 120\ k\Omega$$

se tiene que $$G(s)=\frac{1}{1.44s^2+3.6+1}$$

Discretizando la función de transferencia al período de muestreo T=550 mseg, se tiene:

$$G(z)=\frac{0.0423+0.066z}{z^2-1.14z+0.257}$$

por lo que la ecuación de diferencias es:

$$y(k)=1.14y(k-1)-0.257y(k-2)+0.066u(k-1)+0.0423u(k-2)$$

y los parámetros reales del circuito son:

$$\Theta(k)=\begin{bmatrix}1.14 & -0.257 & 0.066 & 0.0423\end{bmatrix}^T$$

El sistema presenta las siguientes características para su identificación en tiempo real.

Entrada: Ruido PRBS de amplitud 5.
Ruido a la salida: Desconocido.

Identificando el sistema mediante el método de mínimos cuadrados recursivos, se tiene el siguiente reporte en la tabla 4.2.2.1.

MÍNIMOS CUADRADOS RECURSIVO (MCR)
Identificación en tiempo real

Modelo Planteado
Orden de A(z) = 2
Orden de B(z) = 2
Retardo = 1

A(z)	B(z)
a1 = 1.125	b1 = 0.058
a2 = -0.22	b2 = 0.039

Entrada: Ruido PRBS (5)
Número de Iteraciones = 599
Período de muestreo = 550 [mseg]

Tabla de resultados 4.2.2.1 IDENTIFICACIÓN POR MCR

A continuación se presenta los gráficos más importantes en el proceso de identificación, es necesario indicar que la información esta referida a las 200 últimas iteraciones.

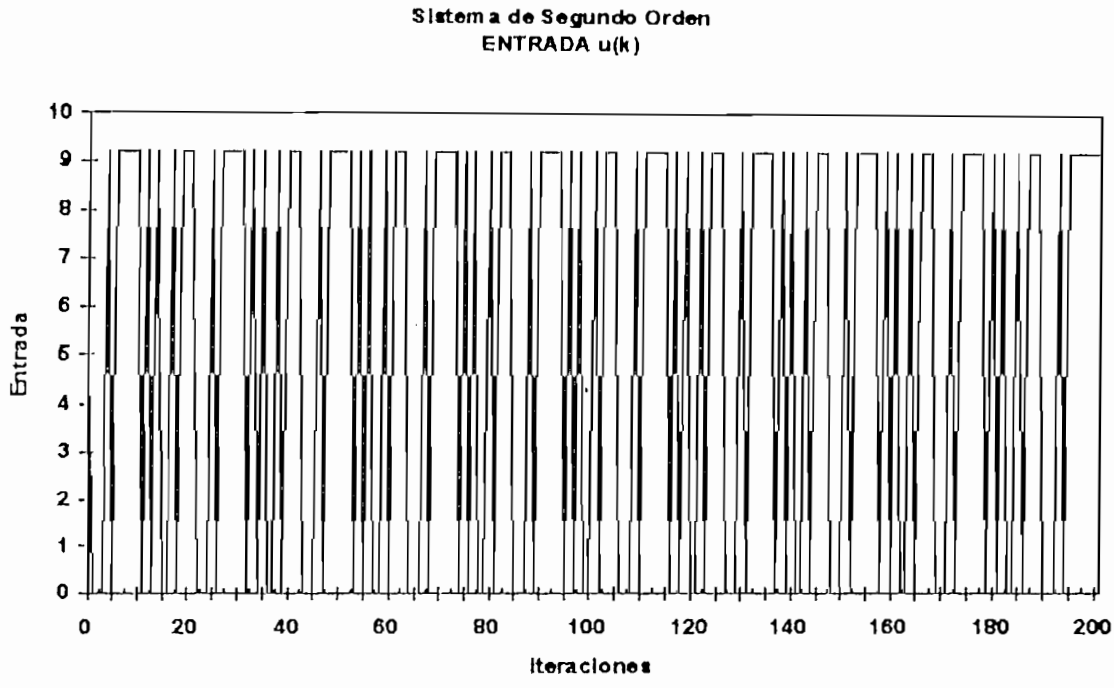

Fig. 4.2.2.2 Entrada al Circuito

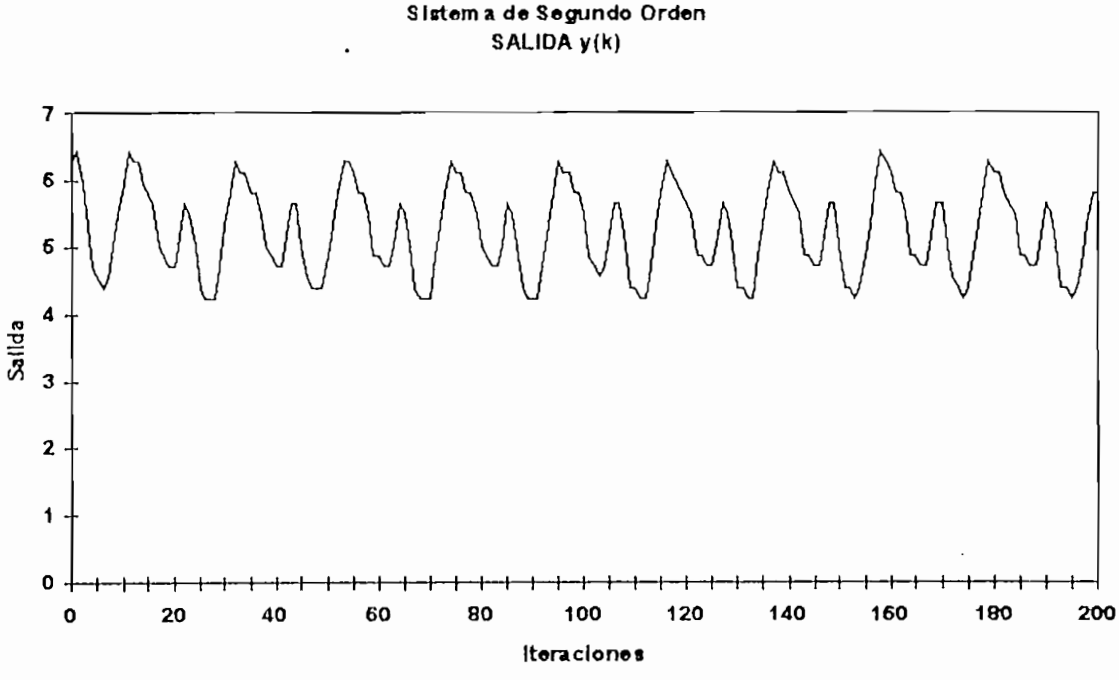

Fig. 4.2.2.3 Salida del Circuito

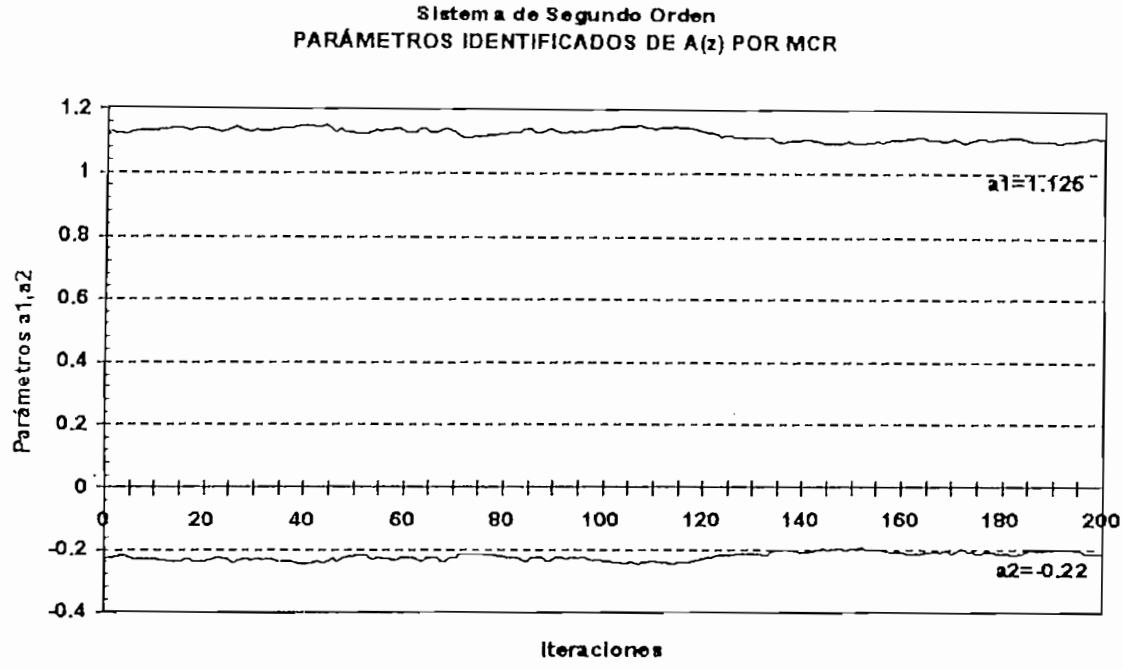

Fig. 4.2.2.4 Parámetros Identificados del polinomio A(z)

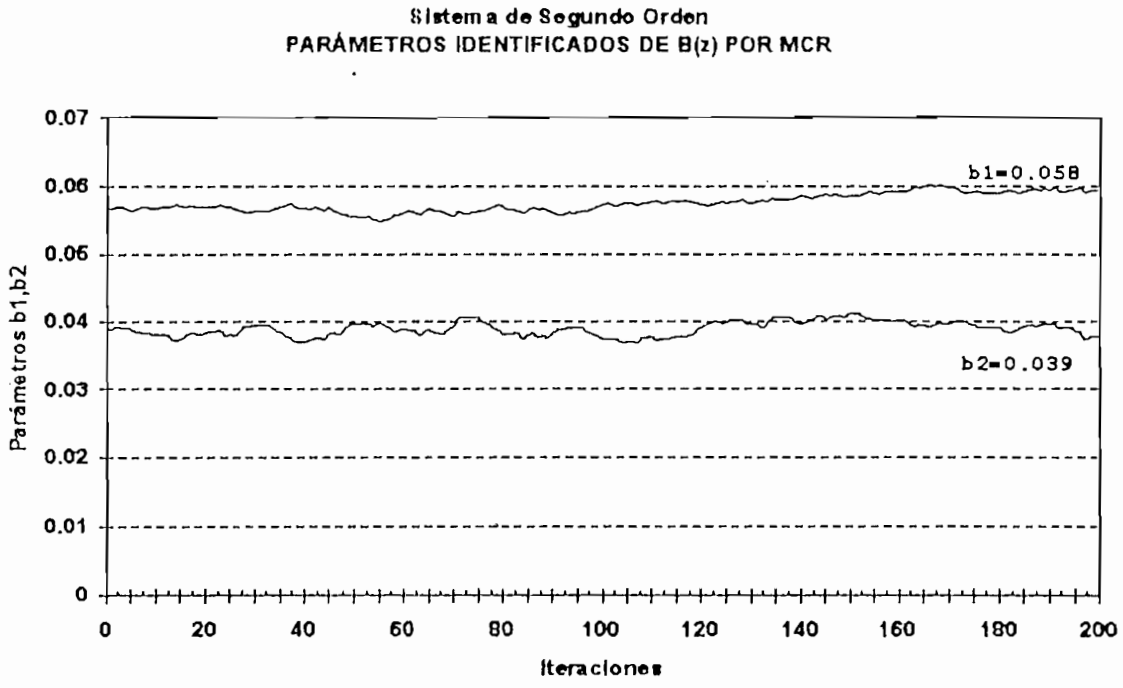

Fig. 4.2.2.5 Parámetros Identificados del polinomio B(z)

A continuación se presenta los resultados de identificar el sistema mediante el método de máximo de Likelihood, en la tabla 4.2.2.2.

MÁXIMO DE LIKELIHOOD (MLH)
Identificación en tiempo real

Modelo Planteado
Orden de A(z) = 2
Orden de B(z) = 2
Retardo = 1

A(z)	B(z)
a1 = 1.155	b1 = 0.057
a2 = -0.237	b2 = 0.0375

Entrada: Ruido PRBS (5)
Número de Iteraciones = 599
Período de muestreo = 550 [mseg]

Tabla de resultados 4.2.2.2 IDENTIFICACIÓN POR MLH

A continuación se presenta los gráficos más importantes en los 200 últimos valores.

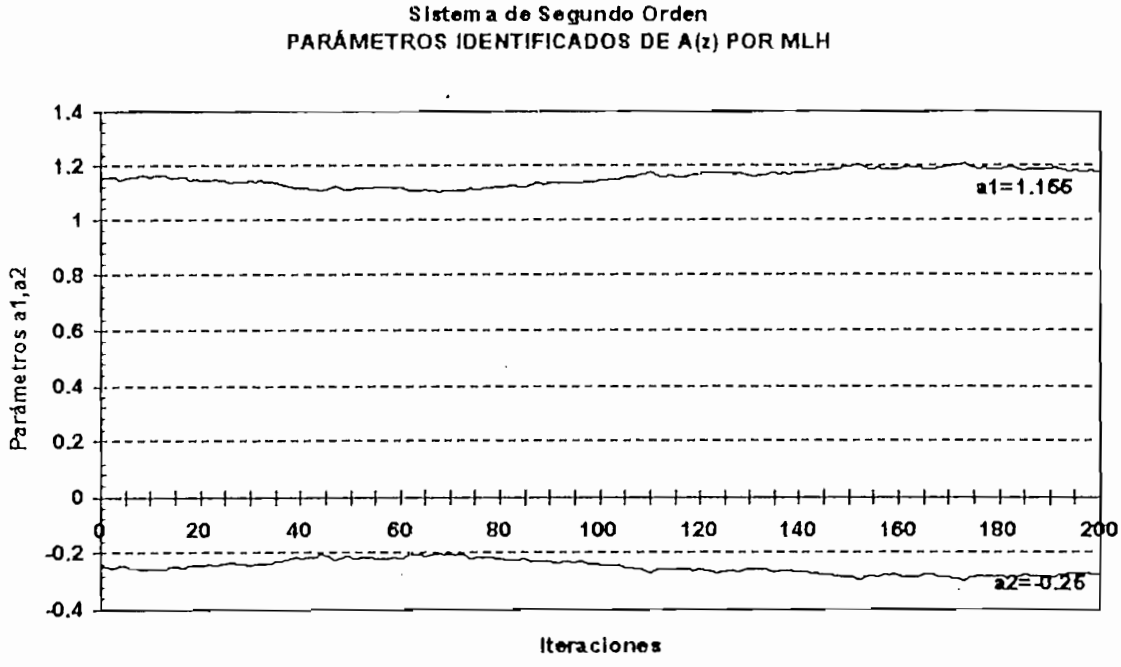

Fig. 4.2.2.6. Parámetros Identificados del polinomio A(z)

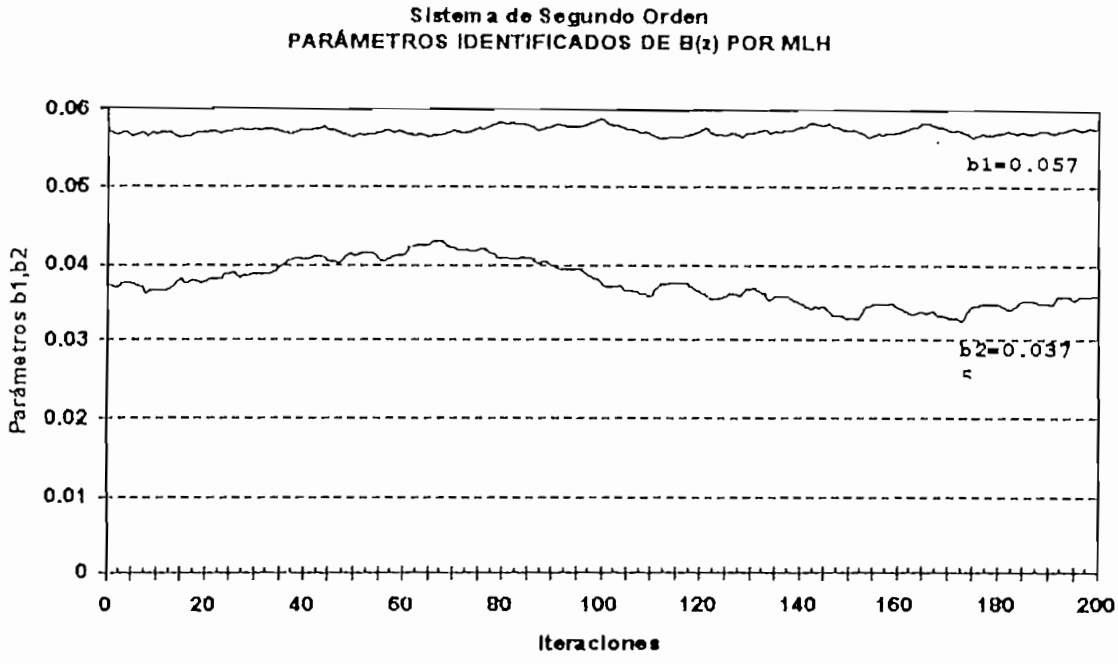

Fig. 4.2.2.7 Parámetros Identificados del polinomio B(z)

4.2.3 PROTOTIPO SISTEMA DE NIVEL DE LÍQUIDOS [6]

Un diagrama del sistema que se identificó se muestra en la figura 4.2.3.1.

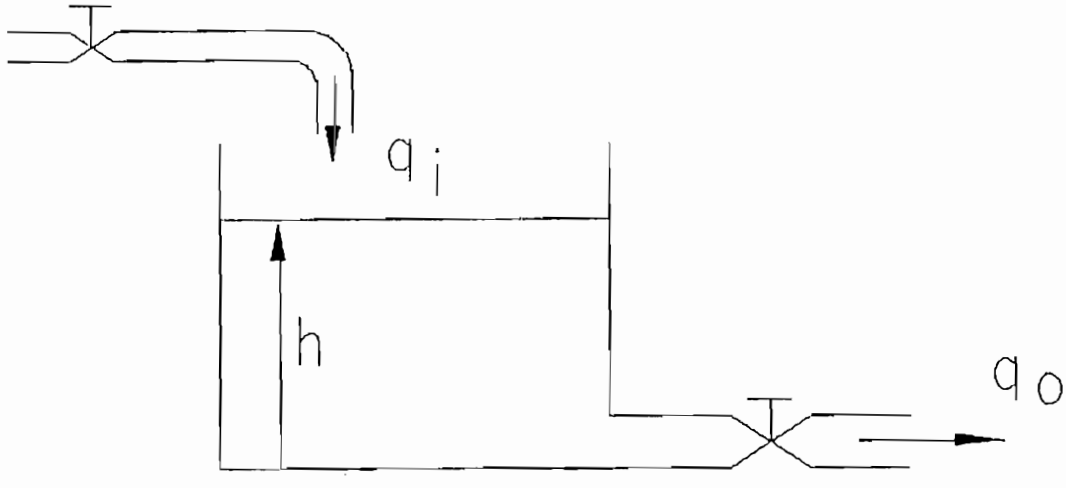

Fig. 4.2.3.1 Diagrama del prototipo de nivel de líquidos

Si el prototipo se encuentra funcionando en estado estable en donde la cantidad de agua que entra es la misma que sale, entonces se somete a pequeñas variaciones en su señal de entrada para identificar al sistema.

En estado estable se tiene que una entrada de 3.5 voltios en el módulo de la bomba, produce una altura en el tanque de aproximadamente 21 cm, que se mantiene constante. Las características de identificación del sistema son:
Entrada: Escalón 3.5 (V) amplitud más ruido randómico 20%.
Ruido a la salida: Desconocido.

Este sistema se identificó mediante el método de mínimos cuadrados recursivos y máximo de Likelihood sin embargo debido a que se tiene resultados similares se presenta un reporte solamente de los resultados por el método de mínimos cuadrados recursivos, mientras que por el método de máximo de Likelihood se presenta toda la información en la identificación del sistema. A continuación en la tabla de resultados 4.2.3 se presenta un reporte de identificación por el método de mínimos cuadrados recursivos.

MÍNIMOS CUADRADOS RECURSIVOS (MCR)
Identificación en tiempo real

Modelo Planteado
Orden de A(z) = 1
Orden de B(z) = 1
Retardo = 1

A(z)	B(z)
a1 = 0.94	b1 = 0.1

Entrada: Escalón (3.5 V) más ruido randómico (20%)
Número de Iteraciones = 599
Período de muestreo = 2000 [mseg]

Tabla de resultados 4.2.3.1 IDENTIFICACIÓN POR MLH

A continuación se presenta una reporte de la identificación del sistema mediante el método de máximo de Likelihood en la tabla 4.2.3.2.

MÁXIMO DE LIKELIHOOD (MLH)
Identificación en tiempo real

Modelo Planteado
Orden de A(z) = 1
Orden de B(z) = 1
Retardo = 1

A(z)	B(z)
a1 = 0.94	b1 = 0.08

Entrada: Escalón (3.5 v) más ruido randómico 20%
Número de Iteraciones = 599
Período de muestreo = 2000 [mseg]

Tabla de resultados 4.2.3.2 IDENTIFICACIÓN POR MLH

A continuación se muestra los gráficos más importantes.

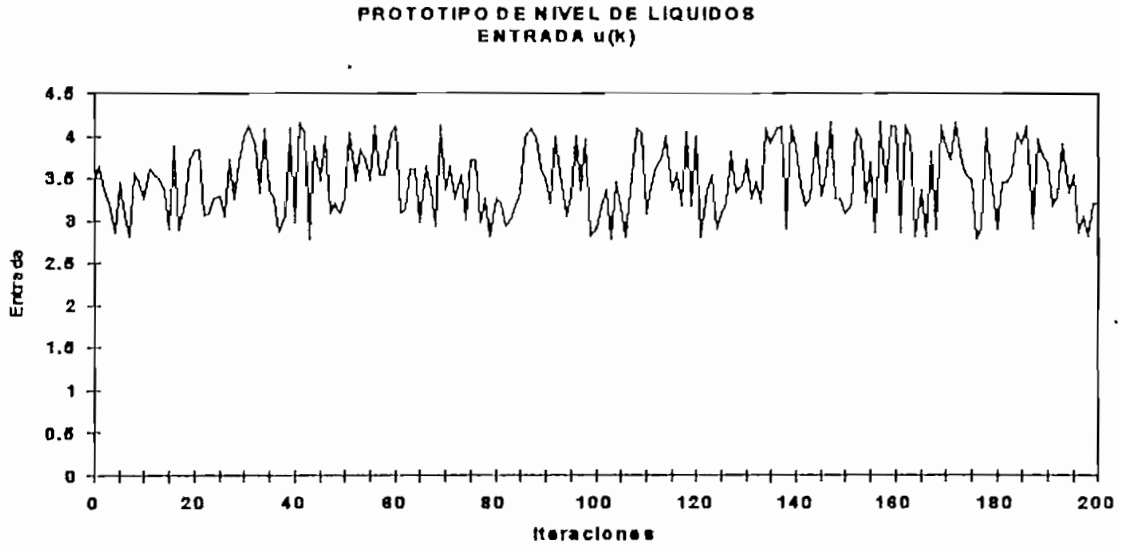

Fig. 4.2.3.2 Entrada al prototipo

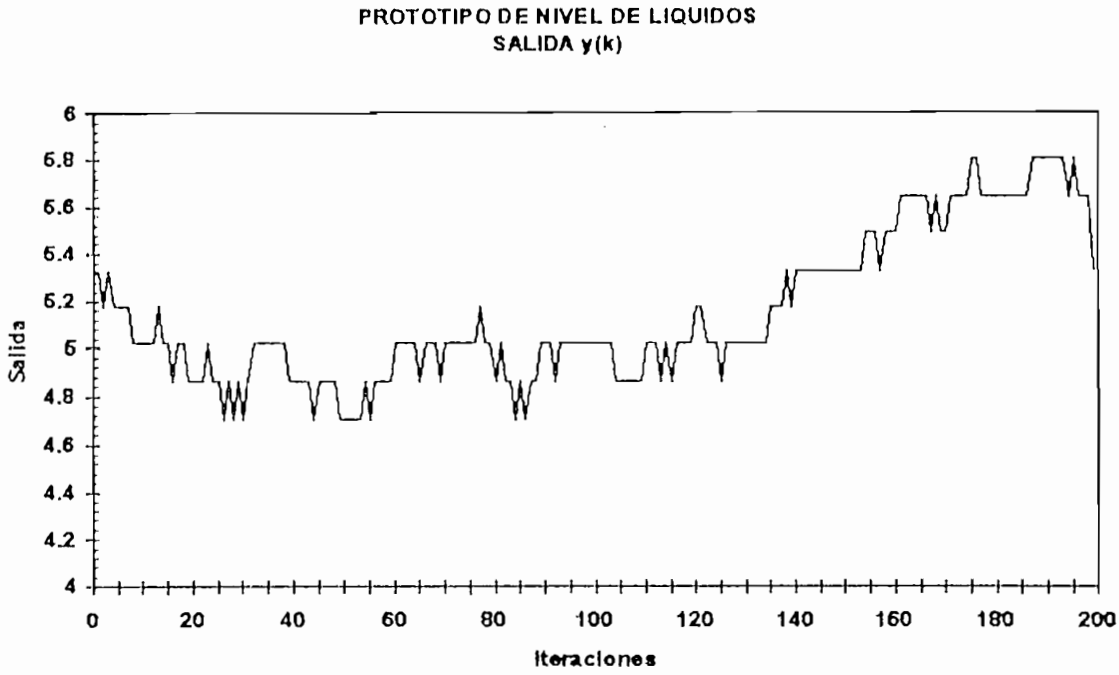

Fig. 4.2.3.4 Salida del prototipo

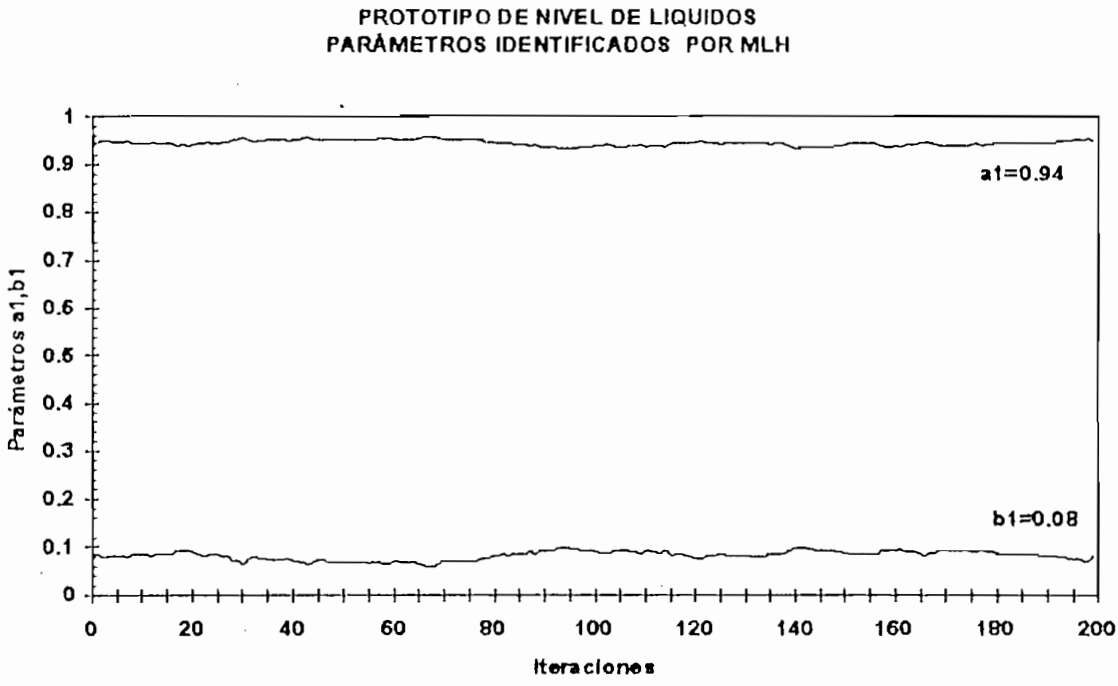

Fig. 4.2.3.6 Parámetros identificados

4.3 CONCLUSIONES

A continuación se presenta el análisis de las pruebas realizadas a nivel de simulación así como en tiempo real para verificar la eficiencia de los distintos algoritmos de identificación desarrollados y para observar como el ruido afecta a la estimación de parámetros. Cabe señalar que se hicieron múltiple pruebas en simulación y en tiempo real con las que se confirmo la validez de los programas desarrollados. El análisis de las pruebas más representativas se muestra a continuación.

Es necesario indicar que cada ejercicio, él cual a sido sometido a distintos algoritmos de identificación tuvo la misma señal de excitación y también el mismo porcentaje de ruido a la salida, ya que solamente de esta manera se puede comprobar las bondades de un método de identificación con respecto a los otros.

Pruebas a nivel de simulación:

El ruido blanco o ruido correlacionado que se suma a la salida de un sistema es un porcentaje de su valor final. El valor final de la salida del sistema se calcula de acuerdo al tipo de entrada que se tenga así: si el sistema tiene una entrada escalón, el valor final se calcula con el valor del escalón, mientras que si la señal de entrada es ruido se toma el valor absoluto del máximo ruido y con este valor se calcula el valor final del sistema.

Al identificar sistemas que tengan ruido blanco a la salida tal como los ejemplos 4.1.1, 4.1.2 y 4.1.3, el algoritmo de

mínimos cuadrados recursivos (MCR) identifica los parámetros adecuadamente, es decir no existe una desviación en el valor de los parámetros estimados. Esto pone de manifiesto el hecho de que el método de MCR con ruido blanco a la salida, da un algoritmo de identificación no desviado. Sin embargo a medida que el orden del sistema se incrementa es necesario incrementar el número de iteraciones, así como el ruido que se tenga a la salida no debe ser exageradamente alto para tener una adecuada identificación.

Al analizar las curvas de ruido a la salida junto con la curva de error de predicción (fig. 4.1.1.4, fig. 4.1.2.5 y fig. 4.1.3.5), se puede notar que cuando los parámetros tienden a estabilizarse el error de predicción sigue en forma fiel al ruido a la salida.

Las tablas de resultados (4.1.1, 4.1.2 y 4.1.3) que se muestran son tablas que presenta el programa de identificación cuando se pide la opción reporte en el cuadro de resultados, en esta tabla se encuentra información de la planta simulada, el modelo escogido, los parámetros reales y parámetros estimados, así como también información sobre la entrada al sistema el ruido a la salida, etc..

Al identificar sistemas con ruido correlacionado a la salida el método de mínimos cuadrados recursivos presenta una desviación en el valor de los parámetros identificados, ejercicio 4.1.4, tabla de resultados 4.1.4.1,. de donde se comprueba que el método de mínimos cuadrados recursivos no es adecuado para identificar sistemas con ruido correlacionado en la salida. En este caso se hace necesaria la utilización de algoritmos mas robustos de identificación

como son: el método de mínimos cuadrados extendidos (MCE) y método de máximo de Likelihood (MLH).

Al comparar las distintas tablas de resultados de los ejemplos (4.1.4, 4.1,5 y 4.1.6) entre el método de mínimos cuadrados extendido y el método de máximo de Likelihood se puede notar que siempre que se tiene ruido correlacionado a la salida los resultados generados por el método de máximo de Likelihood son mejores que los obtenidos a través del método de mínimos cuadrados extendidos, esto porque si bien el método de MCE sirve para identificar sistemas con ruido correlacionado a la salida, este método se basa en un desarrollo determinístico que lo que hace es extender el vector de información a los valores del error de predicción como una aproximación al ruido en la salida del sistema, mientras que el algoritmo de máximo de Likelihood es un método probabilístico desarrollado con la finalidad de trabajar cuando exista presencia de ruido correlacionado. Esta diferencia se acentua a medida que se tiene un mayor número de parámetros c a identificar, debido a que el error de predicción tiende a ser distinto que el ruido presente a la salida; y consecuentemente la aproximación en la identificación de parámetros se deteriora pues se presenta una desviación.

Cuando se tiene sistemas con presencia de ruido correlacionado, es necesario incrementar el número de iteraciones para tener una correcta identificación de los parámetros.

La señal de entrada para identificar a un sistema es muy importante en el proceso de identificación y es necesario indicar que dependiendo del orden del sistema se debe

escoger el tipo de entrada adecuada para excitar al sistema. Si el sistema es de primer orden se puede utilizar una señal de entrada escalón sumada un porcentaje de ruido, mientras que si el sistema a identificar es de orden superior, entonces se debe utilizar una señal aleatoria la cual es de excitación persistente o de cualquier orden de excitación, y es la recomendada para identificar sistemas, esto debido a que a través de esta se puede tener respuesta del sistema en una amplia gama de frecuencias ya que caso contrario cuando la señal de excitación no cumple los requerimientos de excitación persistente y se puede llegar a inestabilidad numérica en los algoritmos o que los parámetros identificados sean desviados.

A lo largo de las distintas pruebas se fueron cambiando ciertas condiciones para identificar un sistema, entre estas se puede mencionar como se ve afectado la identificación de un sistema cuando se varía el factor de olvido. Entonces al tener un factor de olvido constante y disminuir el valor gamma se nota que el valor final no cambia significativamente, sin embargo el gráfico de la convergencia del parámetro se vuelve más oscilante.

Cuando se utiliza el factor de olvido tipo filtro se tiene que este, suprime en un principio grandes picos de los parámetros y en estado estable se comporta en forma similar al factor de olvido constante.

Al utilizar el factor de olvido exponencial se tiene que los parámetros estimados tienden a los mismos valores que cuando se utiliza el factor de olvido constante, por lo que al no tener un cambio sensible es suficiente utilizar un factor de olvido constante.

Pruebas en tiempo real:

Al observar los resultados de los parámetros identificado en el sistema de primer utilizando el algoritmo de mínimos cuadrados tabla 4.2.1.1, se nota que estos convergen a los valores de los parámetros reales para el tiempo de muestreo analizado, así como la estabilidad que presentan es grande, hecho que se desprende de observar la fig. 4.2.1.4.

Al identificar el sistema de segundo orden se utilizó tanto el método de mínimos cuadrados recursivos así como el método de máximo de Likelihood y se llegó a las tablas de resultados 4.2.2.1 y 4.2.2.2 que muestran una marcada cercanía entre parámetros identificados, así como también un valor muy próximo a los valores reales.

El último ejemplo que se identificó en tiempo real fue una prototipo de nivel de líquidos, el cual trabajando en estado estable fue sometido a pequeñas variaciones para poder identificar el mismo, se utilizó tanto el método de mínimos cuadrados recursivos así como también el método de máximo de Likelihood llegando a tener resultados casi idénticos tabla 4.2.3.1 y tabla 4.2.3.2 lo que demuestra que para casos prácticos en donde se tiene prácticamente ruido blanco a la salida el método de mínimos cuadrados recursivos identifica un sistema sin offset en los parámetros identificados.

Es necesario notar que en tiempo real intervienen algunas condiciones que no aparecen en simulación debido a que los sistema reales presentan variaciones, perturbaciones debida a la dinámica de la planta así como también interacciones del sistema con otros sistemas cercanos como la tarjeta de

adquisición de datos. Además que el valor de resistencias así como el valor de los capacitores utilizados tienen tolerancia lo que produce en última instancia que los valores llamados reales de los parámetros sean una aproximación a los valores ideales y que en realidad el programa identifica parámetros tomando en cuenta todas estas situaciones.

A continuación se presentan algunas consideraciones sobre el programa desarrollado, el manejo de la tarjeta DAS-128, etc..

En general cuando se realiza un programa para la plataforma Windows, el programador se despreocupa del manejo de los periféricos ya que esto lo realiza Windows de una forma transparente al programador, con lo cual se gana que el programa desarrollado funcione en una mayor cantidad de computadoras y el programa no sufra casi ninguna distorsión en la salida de resultados tanto a nivel de pantalla como al manejar impresoras, etc..

Al utilizar el direccionamiento a memoria para manejar los puertos se abre la posibilidad de poder utilizar otra tarjeta distinta a la DAS-128, para la tarea de adquisición y salida de datos ya que lo que se debería cambiar en el cuadro de diálogo son las nuevas direcciones en memoria de los puertos que se va a utilizar, estas direcciones que se tienen para el puerto de entrada y salida se debe ingresar como un valor decimal.

Al trabajar con los circuitos RC de primer y segundo orden es necesario desacoplar impedancias a la salida del capacitor y la entrada de la tarjeta de adquisición de datos

ya que en caso de no hacerlo el capacitor nunca llega a cargarse al valor total y no se puede identificar al sistema adecuadamente.

La identificación de sistemas en tiempo real que se realizó con el programa *"IPD.EXE"* muestra la aplicación práctica del mismo, ya que luego de identificar un sistema y tener una ecuación de diferencias entonces se puede empezar a trabajar en el diseño de un control que cumpla ciertas especificaciones, y todo esto sin necesidad de conocer mayor información del sistema.

Luego de haber terminado este trabajo se han cumplido los objetivos que se propuso al inicio de la tesis y que fueron; desarrollar un programa de computación para identificar sistemas a nivel de simulación y en tiempo real utilizando los métodos de mínimos cuadrados recursivo, mínimos cuadrados extendidos y el método de máximo de Likelihood en un ambiente agradable de trabajo que en este caso es la plataforma Windows.

APENDICE A

A.1 MANUAL DEL USUARIO

A.2 ESTRUCTURA DE LOS PROGRAMAS

IDENTIFICACIÓN PARAMÉTRICA DISCRETA "IPD.EXE"

A.1 MANUAL DEL USUARIO

El programa IPD.EXE fue desarrollado para trabajar en el ambiente Windows versión 3.1 o superior, por lo cual es necesario tener lo requerimientos mínimos para que la plataforma Windows funcione adecuadamente y que son:

- Computador 386 o superior.
- Sistema Operativo MS-DOS ver 5.0 ó posterior y la plataforma Windows 3.1.
- Memoria mínima 4 MB.

Para ingresar al programa de identificación paramétrica discreta es necesario tener funcionando la plataforma Windows, y en el grupo de programa de identificación paramétrica discreta realizar doble click sobre el icono que identifica al programa *"IPD.EXE"* (Único en el grupo) ó como forma alternativa se puede escribir en la opción archivos\ejecutar del administrador de programas, el path en donde se encuentre el programa de identificación, con el nombre del programa.

Por ejemplo c:\ipd\ipd.exe

Al ser un programa para Windows su manejo es similar al de otros paquetes comerciales en donde se puede seleccionar opciones con el mouse, aceptar información presionando el botón *OK*, o cancelar el ingreso de información presionando el botón *CANCEL*.

A continuación se explica las distintas capacidades del programa, a través de sus múltiples pantallas.

En la figura a.1 se muestra la pantalla inicial del programa de identificación.

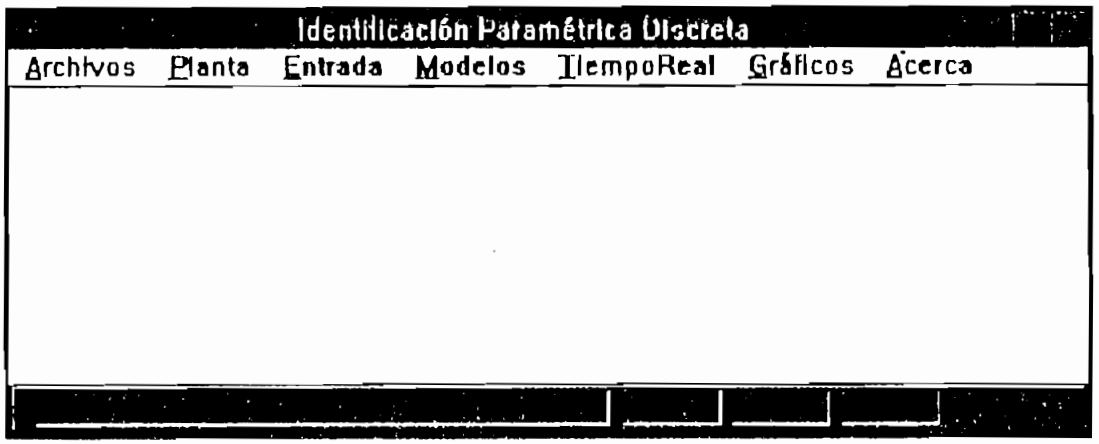

Fig. a.1 Menú Principal del programa IPD.EXE

A continuación se realiza una descripción de las opciones del programa principal.

a.1.1 Opción Archivos

La estructura del submenu ***Archivos*** se muestra en la figura a.2.

La opción ***Abrir*** es la encargada de desplegar el cuadro de diálogo estándar de Windows para recuperar información desde un archivo, este archivo debe ser de extensión *"dat"*, es necesario recalcar que cuando se recupere este archivo las opciones de tipo de entrada, tipo de ruido a la entrada y salida así como el número de iteraciones deben ser las mismas que cuando se grabó el archivo. La información

recuperada va a los vectores de ruido a la entrada y ruido a la salida. Esta opción funciona solamente a nivel de simulación.

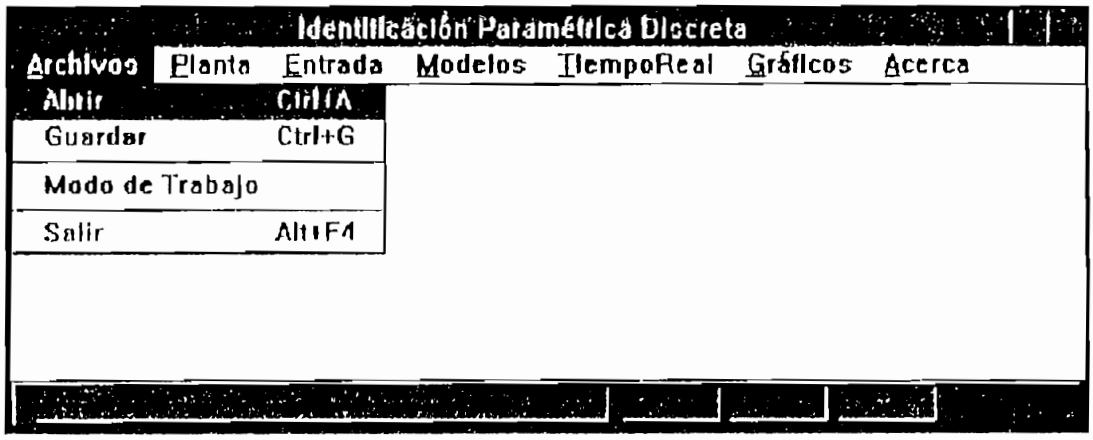

Fig. a.2 Opción Archivos

La opción ***Guardar*** se encarga de desplegar el cuadro de grabar estándar de Windows y permite guardar en un archivo todos los datos sobre la identificación de un sistema ya sea en simulación o en .tiempo real con extensión *"ipd"*, mientras que si el archivo se guarda con extensión *"dat"* se almacenan los vectores de ruido que luego pueden recuperarse con la opción anterior. La información almacenada es:

- A nivel de simulación: vectores de ruido a la entrada y salida, entrada y salida del sistema, ruido blanco a la salida, error de predicción y los parámetros estimados en orden a1,a2.., b1,b2.., c1,c1...

- A nivel de tiempo real: entrada y salida al sistema, error de predicción o residual de acuerdo al método de identificación y los parámetros estimados en orden a1,a2.., b1,b2.., c1,c1...

Toda esta información almacenada puede ser recuperada a través de cualquier hoja de electrónica de cálculo.

En la opción ***Modo de Trabajo*** se despliega un cuadro de diálogo que en donde se tiene la capacidad de comunicarle al programa si se va ha a trabajar a nivel de simulación o en tiempo real. Al seleccionar la opción de trabajar a nivel de tiempo real, se deben ingresar el período de muestreo y las direcciones en memoria para la adquisición y salida de datos desde y hacia la planta respectivamente, dando de esta manera libertad al usuario de poder escoger los canales de la tarjeta DAS-128 que se pueden utilizar.

La opción ***Salir*** *ALT-F4* característica en WINDOWS sirve para terminar la ejecución del programa.

a.1.2 Opción Planta

Este submenu esta disponible solamente cuando se trabaja a nivel de simulación. La estructura del submenu ***Planta*** se muestra en la figura a.3.

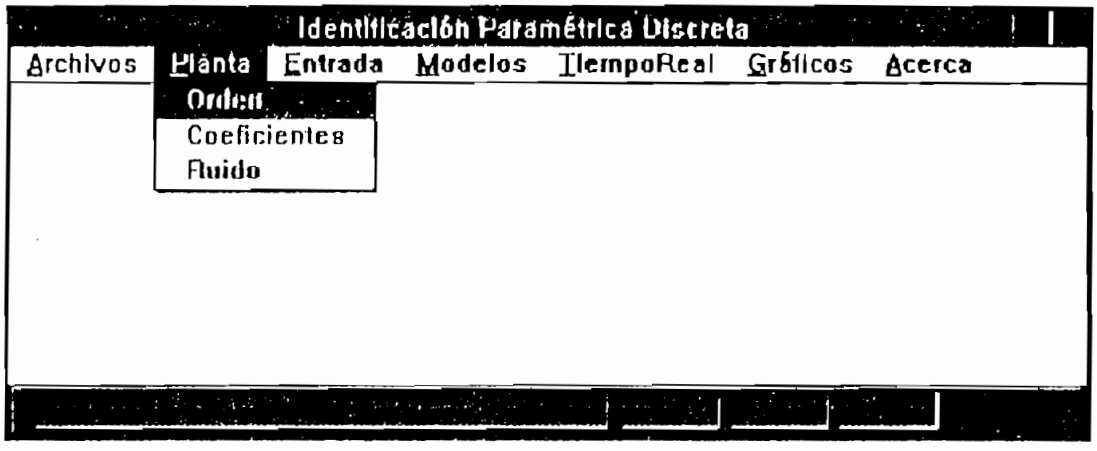

Fig. a.3 Opción Planta

En la opción ***Orden*** se despliega un cuadro de diálogo en donde se ingresan el orden de los polinomios y retardo de la planta que se va a simular, el número de parámetros en total no debe ser superior a 10.

En la opción ***Coeficientes*** se despliega un cuadro de diálogo en el que se deben ingresar los coeficientes de los polinomios de la planta que se va a simular. Por ejemplo si se desea identificar el sistema:

$$y(k)-0.5y(k-1)=1.2u(k-1)$$

entonces se despeja el valor de $y(k)$ y se tiene:

$$y(k)=0.5y(k-1)+1.2u(k-1)$$

de donde los coeficientes se ingresan como:

a1=0.5
b1=1.2

En este cuadro de diálogo se tiene un casillero para ingresar los coeficientes de $A(z)$ y otro para ingresar los coeficientes de $B(z)$ se debe indicar que la información ingresada solamente se conoce en el programa cuando el usuario presiona el botón *+A( )* para los coeficientes del polinomio $A(z)$ y el botón *+B( )* para los coeficientes del polinomio $B(z)$. Además de estos botones existen otros 2 botones que son *-A( )* y *-B( )* que permiten moverse a través de los coeficientes ingresados y realizar cualquier

corrección que sea necesaria. Al presionar cualquiera de los 4 botones explicados anteriormente se presenta en una etiqueta el nombre del coeficiente que se esta viendo.

En la opción ***Ruido*** se despliega un cuadro de diálogo en el cual se tiene la opción de sumar ruido a la salida ya sea ruido blanco o correlacionado, este tipo de ruido puede ser de tres tipos: randómico, PRBS, o estadístico. En este cuadro de diálogo se puede cambiar el porcentaje de ruido adicionado así como también en caso de ingresar ruido correlacionado a la salida se debe ingresar el orden del polinomio $C(z)$, y se presiona el botón de *COEF*, que permite ingresar los coeficientes de este polinomio en forma similar que cuando se ingresa los coeficientes de la planta. El valor del coeficiente $c(0)$ por definición se toma como 1.

a.1.3 Opción Entrada

La estructura de este submenu se muestra en la figura a.4.

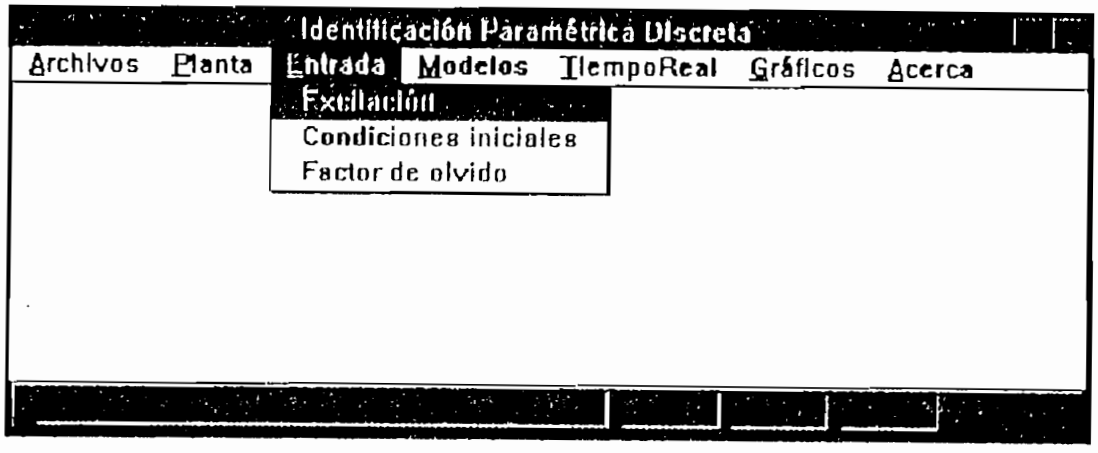

Fig. a.4 Opción Entrada

En la opción ***Excitación*** se despliega un cuadro de diálogo en el cual se tiene que escoger el tipo de entrada que se va a utilizar en el proceso de identificación, esta entrada puede variar entre: una entrada escalón sumada ruido ó solamente uno de los tres tipos de ruidos que puede generar el programa.

En la opción ***Condiciones iniciales***, se despliega un cuadro de diálogo donde se elige el valor inicial alfa de la matriz de covarianza, numero de iteraciones, porcentaje de datos que se utilizará para calcular el valor final de los parámetros estimados, y por último se puede escoger valores iniciales para los parámetros a identificar. Al ser valores esenciales en la identificación existe una verificación que los datos ingresados sea coherentes.

En la opción ***Factor de olvido***, se despliega un cuadro de diálogo donde se escoge el tipo de factor de olvido a utilizar y que puede ser de tres tipos: factor de olvido constante, tipo filtro y exponencial.

a.1.4 Opción Modelos

La estructura de esta opción se muestra en la figura a.5.

Al seleccionar este submenu se presenta el cuadro de diálogo de la fig a.5, en el cual se tiene la posibilidad de escoger el tipo de modelo a utilizar en la identificación y que puede ser: mínimos cuadrados recursivos (MCR), mínimos cuadrados generalizados (MCG) y máximo de Likelihood (MLH).

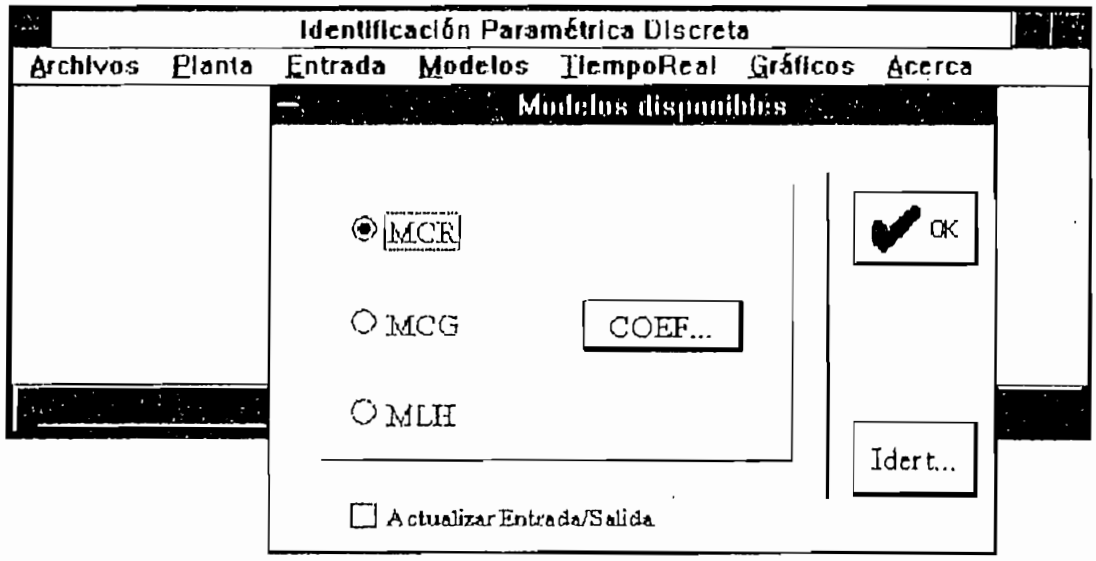

Fig. a.5 Opción Modelos

Se pueden escoger el número de coeficientes que debe tener cada modelo presionando el botón *COEF*, en este momento se despliega un cuadro de diálogo de acuerdo al modelo seleccionado en donde se ingresa esta información. A nivel de simulación se puede utilizar el botón de *Ident* para ver los resultados del proceso de identificación.

La casilla de verificación de este cuadro de diálogo sirve para cuando se desea comparar los métodos de identificación así en modo normal se actualiza cada vez el vector de entrada y salida de ruido al identificar un sistema, pero al desabilitar esta casilla estos vectores permanecen constante con lo cual se puede identificar un sistema por cualquiera de los tres métodos y comparar los resultados con otros valores encontrados por los otros métodos de identificación. Es necesario indicar que cuando se encuentra desactivada esta casilla de verificación los únicos parámetros que se pueden cambiar son el porcentaje de ruido a la entrada y el porcentaje de ruido a la salida, ya que si se escoge otro tipo de ruido esta selección no se toma en cuenta. A nivel

de tiempo real esta casilla de verificación también se encuentra disponible.

a.1.5. Opción Tiempo Real

Este submenu se encuentra solamente disponible cuando se trabaja a nivel de tiempo real y la estructura asociada se muestra en la figura a.6.

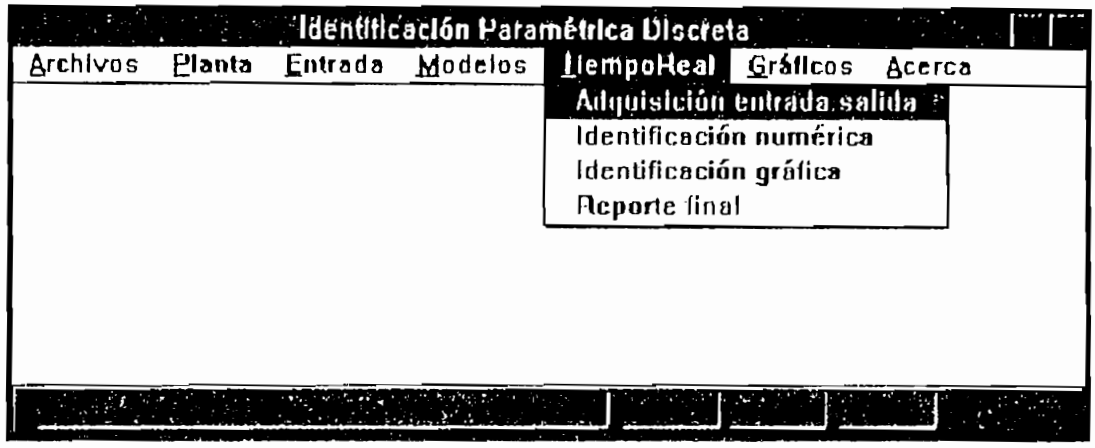

Fig. a.6 Opción Tiempo Real

La opción ***Adquisición entrada salida***, se despliega un cuadro de diálogo donde se puede ver en forma gráfica los valores de la entrada y salida al sistema en esta opción no se identifica ningún parámetro, sino más bien lo que se trata es de ver si la salida varia lo suficiente con la entrada seleccionada para luego identificar parámetros. En este cuadro de diálogo se tiene los siguientes botones:

Iniciar: Activa el temporizador del sistema.
Parar: Desactiva el temporizador del sistema.

Las casillas de verificación que se tiene son:

Entrada u(k): permite visualizar la entrada en la pantalla.
Salida y(k): permite visualizar la salida en la pantalla.

Además existen etiquetas donde se despliega la información referente al numero de la medición, período de muestreo valor de entrada y salida.

La opción ***Identificación numérica***, despliega un cuadro de diálogo que permite identificar al sistema de acuerdo al modelo seleccionado y los resultados los presenta en forma numérica. En este cuadro de diálogo se tiene los botones de *Iniciar* y *Parar* que como en el caso anterior cumplen la misma función, además de información relativa al proceso de identificación.

La opción ***Identificación gráfica***, despliega un cuadro de diálogo en el que se debe ingresar el parámetro a identificar así como un valor tentativo para la escala, luego se presiona el botón *Ver* y se despliega un nuevo cuadro de diálogo en el que se tiene los botones de *Iniciar*, *Parar*, *Escalar por*, *Cambiar a*.

Los botones de *Iniciar* y *Parar* sirven para empezar y detener el proceso de identificación, luego de que el programa empieza a identificar un sistema es posible que la escala seleccionada no sea adecuada por lo cual en el cuadro de edición bajo el botón *Escalar por* se pone el factor por el cual se desea multiplicar el máximo valor de escala, luego se presiona el botón *Escalar por* y el gráfico se actualiza al nuevo valor de escala.

Al estar identificando es necesario ver como van variando los distintos parámetros del modelo, por lo cual se puede ingresar en la casilla de edición bajo el botón de *Cambiar a* el nombre de un nuevo parámetro a visualizar, luego se presiona el botón *Cambiar a* y se actualiza la pantalla con el nuevo gráfico del parámetro que se ingresó.

Es necesario indicar que existen algunos cuadros de mensaje que alertan al usuario cuando se ingresa un factor de escala fuera de rango o el nombre de un parámetro que no existe.

La opción ***Reporte final***, muestra los resultados de la última identificación en tiempo real y en este momento existe un botón para imprimir este resultado, utilizando el cuadro de dialogo estándar de Windows para imprimir documentos.

a.1.6 Opción Gráficos

El submenú asociado se muestra en la figura a.7

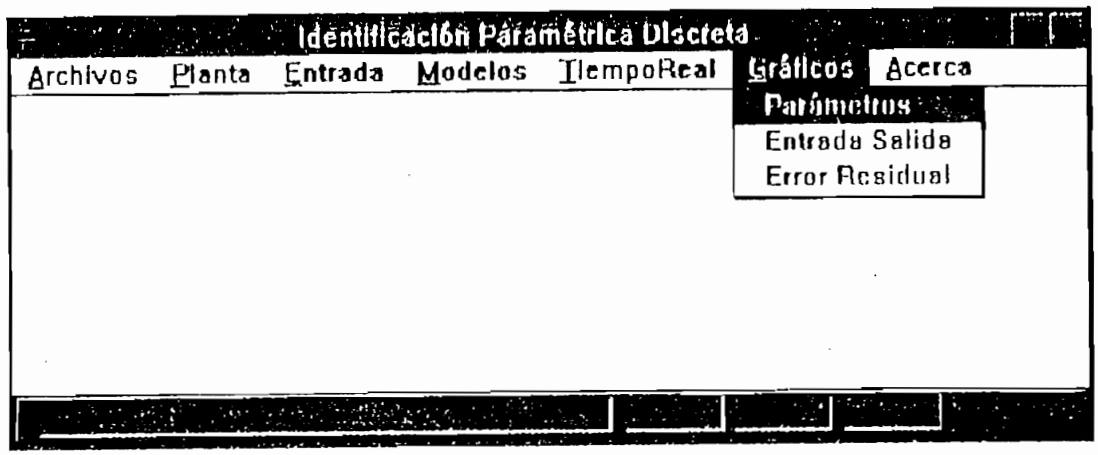

Fig. a.7 Opción Gráficos

En la opción ***Parámetros*** se despliega un cuadro de diálogo en donde se debe ingresar el parámetro a observar de la última identificación sea que se trabaje a nivel de simulación o en tiempo real. Luego de haber ingresado el nombre del parámetro a observar (a1,a2,b1,b2,c1,c2, en este formato) se presiona el botón *Ver* y se despliega un cuadro de diálogo donde se presenta el gráfico del parámetro, en este cuadro de diálogo se tiene un botón para *Escalar* el gráfico, así como se muestra el valor final a que convergió el parámetro.

En la opción ***Entrada Salida*** se presenta los gráficos de la entrada y salida del sistema.

La opción ***Error Residual***, se encuentra habilitada solamente a nivel de simulación y a través de esta se tiene la capacidad de ver como el error de predicción para el método de mínimos cuadrados recursivos y mínimos cuadrados extendidos o el residual para el máximo de Likelihood, se aproxima al ruido blanco en la salida del sistema. Se tiene una opción de *Igualar escalas* para que la comparación sea más adecuada.

a.7 Opción Acerca

En esta opción se despliega un cuadro de diálogo referente a datos del autor así como la fecha del desarrollo del software.

A.2. ESTRUCTURA DE LOS PROGRAMAS

El programa de identificación paramétrica discreta esta elaborado como un archivo de proyecto por lo cual existe una serie de archivos que lo conforman y que interactuan con este para tener el archivo IPD.EXE. Para el funcionamiento del programa es necesario que en el directorio de trabajo se encuentren cuatro archivos de extensión dll que son bibliotecas que permiten realizar llamadas a los cuadros de diálogo así como también la posibilidad de recoger información desde los mismos.

A continuación se presenta una lista de los archivos que se deben tener en el subdirectorio de trabajo para que el programa de identificación paramétrica discreta funcione adecuadamente.

Archivos *.dll

- bwcc.dll
- commdlg.dll
- pvplus.dll
- winct.dll

Archivos *.exe

- ipd.exe

Archivos *.obj

- acerca.obj
- addat.obj
- coefc.obj
- coefps.obj
- condinic.obj
- entrada.obj
- grafmcr.obj
- graftr.obj
- grafvar.obj
- identr.obj
- ipd.obj

- modelos.obj
- modotrab.obj
- ordenmcg.obj
- odenmcr.obj
- ordenmlh.obj
- ordenps.obj
- pantgraf.obj
- pantmcr.obj
- reporte.obj
- reptr.obj
- resmcr.obj
- salidaruy.obj
- satatusli.obj

archivos *.res

- ipd.res

A nivel de archivos fuente el proyecto se pueden dividir en:

a.- Archivo *.prj que es el que contiene la definición de todos los archivos que se deben incluir en la compilación.

- ipd.prj

b.- Archivos *.c son los archivos base de programación, estos archivos estan ligados a un cuadro de diálogo por lo cual en estos se encuentra la administración de los mensajes propios de cada cuadro de diálogo y son:

- acerca.c
- addat.c
- coefc.c
- coefps.c
- condinic.c
- entrada.c
- grafmcr.c
- graftr.c
- grafvar.c
- identr.c
- ipd.c
- modelos.c
- modotrab.c
- ordenmcg.c

- odenmcr.c
- ordenmlh.c
- ordenps.c
- pantgraf.c
- pantmcr.c
- reporte.c
- reptr.c
- resmcr.c
- salidaruy.c
- satatusli.c

c.- Archivos *.h son los archivos de cabecera en estos se encuentran la mayoria de funciones desarrolladas para identificar un sistema y son:

- calmcg.h
- calmcr.h
- calmlh.h
- esruido.h
- ingca.h
- ingcb.h
- ingcc.h
- ipd.h
- mcg.h
- mcr.h
- mlh.h
- ordenn.h
- repsim.h
- reslmcg.h
- resmcr.h
- resmlh.h
- resource.h
- resultr.h
- ruido.h
- simular.h
- valorb.h
- valorc.h
- valorp.h
- variab.h

d.- Archivos *.dlg son los que contienen la información de los cuadros de diálogos y son los mismo que se tienen en los archivos *.c, pero con extensión .dlg.

e.- Archivo *.res es el archivo de recurso que se genera cuando se compila el programa el paquete ProtoGen+.

- ipd.res

f.- Archivo *.rc es el archivo de recursos del programa IPD.EXE, este archivo es un archivo gráfico.

- ipd.rc

g.- Archivo *.mnu es el archivo que contiene la información sobre el menu que se utiliza en el programa de identificación.

- ipd.mnu

h.- Archivo *.pva este archivo se genera en ProtoGen+ y es el que guarda el entorno de desarrollo de la aplicación.

- ipd.pva

i.- Archivos *.cur y archivo *.ico son archivos del cursor y el ícono que se utilizan en el programa de computación.

- hand.cur
- graf04.ico

Todos los archivos fuente se encuentran impresos en el anillado que se encuentra anexado a la tesis bajo el nombre de *"Programas fuente de IPD.EXE"*.

BIBLIOGRAFIA

1.- Åström k.J.; "Introduction to Stochastic Control Theory"; Vol 70; Academic Press; New York.

2.- Åström K.J.; "MAXIMUM LIKELIHOOD AND PREDICTION ERROR METHODS"; Automática Vol 16; pp 551-574;1980.

3.- Åström K.J. & Wittenmark B.; "ADAPTIVE CONTROL"; Lund Institute of Technology; Addison-Wesley Publishing Company; 1989.

4.- Bennett B.S.; "ANALYSIS AND DESIGN OF LINEAR SISO CONTROL SYSTEMS"; Introduction-Systems and Models; Control Systems Centre, University of Manchester Institute of Science and Technology (UIMS), Manchester 1980.

5.- Bennett B.S. & Linkens D.; "COMPUTER CONTROL OF INDUSTRIAL PROCESSES"; Peter Peregrinus LTD. on beholf of the Institution of Electrical Engineers; 1982.

6.- Benitez D.; "DISEÑO Y CONSTRUCCIÓN DE UN SISTEMA PARA CONTROL DE NIVEL DE LIQUIDOS"; Anales XVI; Jornadas de Ingeniería Eléctrica y Electrónica; E.P.N.; Quito-Ecuador; 1995.

7.- Cevallos Fausto; "DISEÑO Y CONSTRUCCIÓN DE UNA TARJETA DE ADQUISICIÓN DE DATOS PARA COMPUTADORES PERSONALES"; E.P.N.; 1992.

8.- Clarke D.W.; "GENERALIZED LEAST SQUARES ESTIMATION OF THE PARAMETERS OF A DYNAMIC MODEL"; Identification and Automatic Control Systems; New College Oxford; 1967.

9.- Cordero Patricio; Tesis; "MÍNIMOS CUADRADOS GENERALIZADO"; E.P.N.; 1986.

10.- Cruz Ricardo; "IDENTIFICACIÓN DE PROCESOS POR EL MÉTODO RECURSIVO DE PROBABILIDAD MÁXIMA"; pp 39-47; Anales de Jornadas en Ingenieria Eléctrica y Electrónica; E.P.N.; 1989.

11.- Cruz Ricardo; Tesis: "IDENTIFICACIÓN DE PROCESOS POR EL MÉTODO RECURSIVO DE PROBABILIDAD MÁXIMA"; E.P.N.; 1989.

12.- Faison T.; "BORLAND C++ 3.1 PROGRAMACION ORIENTADA A OBJETOS"; Prentice-Hall; 1993.

13.- Franklin G.F. & Powell I.D.; "DIGITAL CONTROL OF DYNAMIC SYSTEMS"; Addison-Wesley, U.S.A. 1980.

14.- Godfrey K.R.; "CORRELATION METHODS"; Automatica IFAC; Vol 16; 527-534; 1980.

15.- Harris C.J. & Billings S.A.; "SELF TUNNING AND ADAPTIVE CONTROL: THEORY AND APPLICATIONS"; Peter Peregrinus LTD. on beholf of the Institution of Electrical Engineers; 1981.

16.- Isermann Rodolf; "DIGITAL CONTROL SYSTEMS"; Spring Verlag; New York ; 1981.

17.- Leigh J.R.; "MODELLING AND SIMULATION"; Peter Peregrinus LTD. on behalf of the Institution of Electrical Engineers; 1983.

18.- Murray W. & Pappas C; "PROGRAMACION EN WINDOWS 3.1"; Osborne/McGraw-Hill; 1993.

19.- Ogata Katsuhiko; "Ingenieria de Control Moderna"; Prentice Hall; 1993.

20.- Paez Rodrigo; TESIS: "TECNICAS DE IDENTIFICACION PARA CONTROL DIGITAL"; E.P.N.; 1991.

21.- Rojas V. & Ñacato J.; "TECNICAS DE FLUJOGRAMAS"; Cicetronic; Quito; 1980.

22.- Schildt Helbert; "TURBO C/C++ MANUAL DE REFERENCIA"; Osborne/McGraw-Hill; 1992.

23.- Söderström T. & Storca P.; "COMPARISON OF SOME INSTRUMENTAL VARIABLE METHODS - CONSISTENCY AND ACCURACY ASPECTS"; Identification Federation of Automatic Control (IFAC); 1981.

24.- Strejc V.; "LEAST SQUARE PARAMETER ESTIMATION"; Automática Vol 10; pp 535-550; 1980.

25.- Valenzuela Ramiro; "IDENTIFICACION DE SISTEMAS"; E.P.N.; 1896.

26.- Warwick K. & Rees D.; "INDUSTRIAL DIGITAL CONTROL SYSTEMS"; Peter Peregrinus LTD. on beholf of the Institution of Electrical Engineers; 1988.

27.- Wellstead P.E. & Prager D.L.; "INTERACTIVE MAXIMUM LIKELIHOOD ESTIMATION"; Control Systems Centre, University of Manchester Institute of Science and Technology; Sackvilla Street; Manchester; 1979.

28.- Wellstead P.E.; "SELF TUNNING CONTROL"; Part One; UIMS; 1982.

29.- Zarrop Martin B.; "VARIABLE FORGETTING FACTORS IN PARAMETER ESTIMATION"; Automática Vol 19; N.3; pp 295-298; 1983.

30.- Bendat J., Piersol A.; "RANDOM DATA: ANALYSIS AND MEASUREMENT PROCEDURES"; Wiley Interscience; 1971.

31.- Rodriguez A, Tesis; "SIMULACIÓN ESTADÍSTICA", E.P.N., 1983.