ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

CONTROL ADAPTATIVO EN TIEMPO REAL

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERA EN ELECTRÓNICA Y CONTROL

JANETH CAROLINA GODOY ORTEGA
godoycarolina621@yahoo.com

DIRECTOR: Prof. PATRICIO BURBANO R.
pburbanor@hotmail.com

Quito, Marzo del 2011

DECLARACIÓN

Yo Janeth Carolina Godoy Ortega, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

_____

Janeth Carolina Godoy Ortega

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por la Srta. Janeth Carolina Godoy Ortega bajo mi supervisión.

_____

Prof. Patricio Burbano R.
DIRECTOR DEL PROYECTO

AGRADECIMIENTOS

Mi más sincero agradecimiento al Ing. Patricio Burbano por su acertada dirección durante la elaboración del presente Proyecto de Titulación.

Gratitud para un amigo muy especial, Carlos, ya que gracias a su fe en mí pude continuar y culminar con mis estudios en la Escuela Politécnica Nacional.

Un reconocimiento a mis amigos politécnicos que colaboraron para la realización del presente Proyecto y que estuvieron presentes durante mi paso por la “Poli”.

A Juan Pablo, quien siempre tuvo esa palabra precisa en los momentos en los que la necesitaba.

DEDICATORIA

Esta Tesis está completamente dedicada a Mis Padres.

Mi Madre, quien ha sido mi inspiración y mi fuerza durante toda mi vida. Madre eres un ejemplo a seguir y todo lo bueno que soy te lo debo a ti. No existen palabras en este mundo para poder expresar el enorme amor y respeto que te tengo, la manera que encontré de demostrártelo es haciéndote sentir orgullosa de mi.

Mi Padre, quien aunque no está físicamente junto a mí, desde el cielo me envía sus bendiciones.

Nunca los voy a defraudar.

RESUMEN

El presente Proyecto de Titulación está enfocado a los Sistemas de Control, específicamente al Control Adaptativo Autosintonizable; se analiza su funcionamiento y su diseño, para lo cual se debe realizar la Identificación de los Parámetros de la Planta necesaria para el Cálculo del Controlador.

El proceso de Identificación se lo ha realizado por diferentes métodos utilizando Matlab, con la finalidad de que el estudiante que consulte este Proyecto, analice y utilice la mejor opción para su aplicación.

Una vez obtenidos los parámetros de la planta correctamente identificados, se procede al cálculo del Controlador Autosintonizable por el método de asignación de polos.

Los procesos anteriormente mencionados se realizan tanto en Simulación, utilizando modelos de diferente orden, como en Tiempo Real, utilizando simuladores de plantas de segundo orden basados en circuitos pasivos y activos mediante amplificadores operacionales, que en principio permiten simular cualquier planta. Se ha realizado una plataforma de desarrollo en el programa computacional Matlab, además de que para la adquisición de datos se ha utilizado la Tarjeta Ni-DAQ 6008.

PRESENTACIÓN

El Proyecto de Titulación “Control Adaptativo en Tiempo Real”, se encuentra dividido en seis capítulos que se describen a continuación.

El Capítulo I titulado Introducción, muestra un pequeño resumen del Proyecto en sí, está dividido en tres subcapítulos, los mismos que explican qué es la Identificación de Sistemas y los métodos utilizados para la misma, además explica brevemente de que se trata el Control Adaptativo y presenta un resumen de la plataforma de desarrollo.

El Capítulo II titulado Identificación de Sistemas, explica de manera detallada los métodos de identificación utilizados en el presente Proyecto, siendo estos los Mínimos Cuadrados, Mínimos Cuadrados Recursivo y Mínimos Cuadrados Extendido. Asimismo, se expone la identificación utilizando el Matlab-Simulink con sus diferentes formas de obtener los parámetros identificados de la planta. Estas formas son: utilizando los Comandos de Matlab, Programación en archivos.m, el Toolbox de Identificación de Sistemas propio de Matlab y por supuesto el Simulink. Se indica un ejercicio de Identificación de Parámetros con todos los métodos anteriormente mencionados para poder validar los resultados.

El Capítulo III titulado Estudio del Control Adaptativo, analiza tres clases de controladores adaptativos, Programación de Ganancias, Control Adaptativo con Modelo de Referencia (Model Reference Adaptative Control) y Regulador Autosintonizable; realizando un breve análisis sobre los dos primeros y profundizando el tercer tipo de controlador. Se analiza su funcionamiento, se realiza el diseño del mismo y por último se presenta la simulación.

El Capítulo IV titulado Descripción de la Plataforma en Matlab, detalla las plantas utilizadas para la verificación del Controlador, siendo dichas plantas dos circuitos

eléctricos, uno activo y uno pasivo, obteniendo sus datos por medio de la Tarjeta de Adquisición Ni-DAQ 6008 de la cual se indica su funcionamiento. Además, se explica el control en tiempo real con el cálculo de un controlador conocido, el PID, para los dos circuitos.

El Capítulo V titulado Pruebas y Resultados, explica con ejemplos los resultados obtenidos en cada proceso de diseño, es decir durante la Identificación y durante el cálculo del controlador, tanto en simulaciones como en tiempo real.

En el Capítulo VI se exponen las conclusiones a las cuales se llegaron durante el desarrollo de este Proyecto, y ciertas recomendaciones que se esperan sean de utilidad para futuros trabajos.

Se anexa el Manual de la Tarjeta de Adquisición de Datos Ni-DAQ 6008, para comprender el manejo de la misma. Conjuntamente, se agrega el Manual de Usuario del Control Adaptativo para utilizar la plataforma de desarrollo en Matlab; y por último se encuentran los programas de Control Adaptativo realizados en archivo.m en Matlab.

La Bibliografía utilizada también se presenta al final de este Proyecto.

CONTENIDO

DECLARACIÓN……..i

CERTIFICACIÓN……..ii

AGRADECIMIENTO……iii

DEDICATORIA……iv

RESUMEN……v

PRESENTACIÓN……....vi

CAPÍTULO I 1

INTRODUCCIÓN 1

1.1. IDENTIFICACIÓN 1

1.2. CONTROL ADAPTATIVO 3

1.3. PLATAFORMA DE DESARROLLO 5

CAPÍTULO II 7

IDENTIFICACIÓN DE SISTEMAS 7

2.1. IDENTIFICACIÓN USANDO MÍNIMOS CUADRADOS 7

2.2. IDENTIFICACIÓN USANDO MÍNIMOS CUADRADOS RECURSIVO 12

2.3. IDENTIFICACIÓN USANDO MÍNIMOS CUADRADOS EXTENDIDO 15

2.4. IDENTIFICACIÓN USANDO MATLAB-SIMULINK 17

2.4.1. IDENTIFICACIÓN USANDO COMANDOS DEL MATLAB 17

2.4.2. IDENTIFICACIÓN USANDO PROGRAMACIÓN (ARCHIVO.M) 20

2.4.3. IDENTIFICACIÓN USANDO EL SYSTEM IDENTIFICATION TOOLBOX 22

2.4.4. IDENTIFICACIÓN USANDO SIMULINK 28

CAPÍTULO III 40

ESTUDIO DEL CONTROL ADAPTATIVO 40

3.1. CLASES DE CONTROLADORES ADAPTATIVOS 40

3.2. REGULADOR AUTOSINTONIZABLE (STR 44

3.3. CÁLCULO DE CONTROLADOR 48
3.4. SIMULACIÓN 51
CAPÍTULO IV 53
PLATAFORMA DE DESARROLLO PARA EL CONTROL ADAPTATIVO 53
4.1. DESCRIPCIÓN DE LA PLATAFORMA EN MATLAB 53
4.2. ADQUISICIÓN DE DATOS 54
4.3. SALIDA DE DATOS 55
4.4. CONTROL EN TIEMPO REAL 57
CAPÍTULO V 62
PRUEBAS Y RESULTADOS 62
5.1. RESULTADOS DE SIMULACIÓN 62
5.2. RESULTADOS DE TIEMPO REAL 76
CAPÍTULO VI 86
CONCLUSIONES 86
6.1. CONCLUSIONES 86
6.2. RECOMENDACIONES 87
REFERENCIAS BIBLIOGRÁFICAS 89

ANEXOS
A.1 Manual de la Tarjeta Ni- DAQ 6008
A.2 Manual de Usuario del Control Adaptativo

CAPÍTULO I
INTRODUCCIÓN

1.1. IDENTIFICACIÓN

"La identificación de sistemas es la teoría y el arte de construir modelos matemáticos de sistemas dinámicos basándonos en las entradas y salidas observadas".[1]

Para realizar el control de sistemas dinámicos es necesario conocer un modelo que describa la dinámica de dicho sistema; este modelo se lo obtiene ajustando los parámetros de la planta a un conjunto de ecuaciones capaces de detallar el comportamiento de la misma.

La Identificación Paramétrica se puede realizar en forma Off Line, es decir que el computador simplemente es una herramienta que sirve para el cálculo de los parámetros buscados y no forma parte del lazo de control. En esta técnica se debe tomar en cuenta una estructura para el modelo, cuyos parámetros son calculables mediante ciertos criterios que se dan entre el proceso como tal y el sistema modelado.

Las técnicas utilizadas para la identificación paramétrica son:

En el dominio del tiempo: Minimiza el error temporal entre el proceso y el modelo del mismo.

En el dominio de la frecuencia: Minimiza el error entre la respuesta de frecuencia real del proceso y la respuesta de frecuencia del modelo.

[1] GARRIDO, Santiago. "Identificación, Estimación y Control de Sistemas No-lineales mediante RGO". 1990

En la Identificación Paramétrica en Sistemas, se toman en cuenta principalmente dos técnicas: determinísticas y estocásticas.

Las técnicas determinísticas logran identificación a partir de la respuesta de frecuencia ante señales de entrada sinusoidales; y en el dominio del tiempo, las señales de entrada son funciones paso o impulso. Las técnicas estocásticas logran identificación a partir de la respuesta de frecuencia y el dominio del tiempo con señales de entrada aleatorias.

La Identificación Paramétrica se puede programar On Line, es decir el PC está en el lazo de control por lo que se denomina Control en Tiempo Real, puesto que en este Proyecto se va a trabajar en tiempo real, el Esquema de Control a implementar es el de la figura 1.1.

Para la identificación se utilizan diferentes métodos, entre los algoritmos de identificación se pueden mencionar el de los mínimos cuadrados ordinario (MCO), mínimos cuadrados recursivo (MCR), y mínimos cuadrados extendido (MCE).

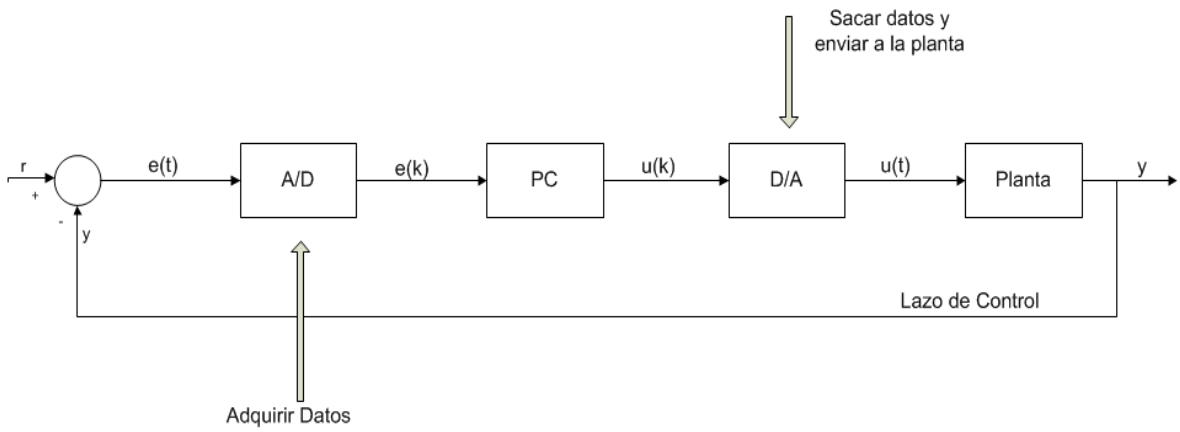


Figura 1.1 Control en Tiempo Real

El método de los Mínimos Cuadrados Ordinario (MCO) se utiliza para obtener los parámetros y también en el proceso de estimación de parámetros. Su ecuación en diferencia viene dada por:

$$y(k) + a_1.y(k-1) + a_2.y(k-2) + \cdots + a_n.y(k-n) = b_1.u(k-1) + b_2.u(k-2) + \ldots + b_n.u(k-n)$$

El Método de Mínimos Cuadrados Recursivo, es un proceso que como su nombre mismo lo dice realiza recursión, es decir, actualiza los parámetros de la planta en cada iteración, dichos parámetros al igual que en MCO son *a* y *b*; en este método necesariamente se utiliza conversión A/D para la adquisición de datos y conversión D/A para salida de datos, ya que son valores obtenidos desde una planta en un proceso en tiempo real, así:

Figura 1.2. Conversión Análoga Digital y Digital Análoga

En el Método de los Mínimos Cuadrados Extendido, se realiza lo mismo que en MCR, es decir se obtiene el modelo de la planta, pero se le añade el modelo de la perturbación, ya que en la mayoría de los sistemas reales existe presencia de éste, como se puede observar en la figura 1.3.

1.2. CONTROL ADAPTATIVO

El Control Adaptativo es un tipo de control en el cual los parámetros de su controlador pueden cambiar de acuerdo a la dinámica del proceso, logrando así que el comportamiento del sistema en lazo cerrado conserve las características de diseño requeridas.

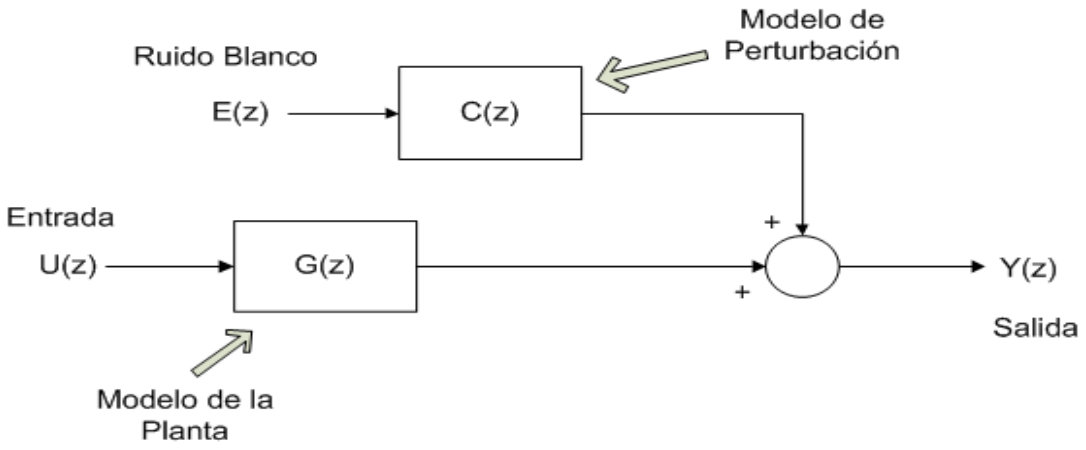


Figura 1.3. Sistema Real con Ruido

Como se puede observar en la figura 1.4, el control adaptativo está formado por dos lazos, el lazo de realimentación compuesto por el proceso y un controlador; y el lazo de ajuste de parámetros, el cual es más lento que el lazo normal de realimentación.

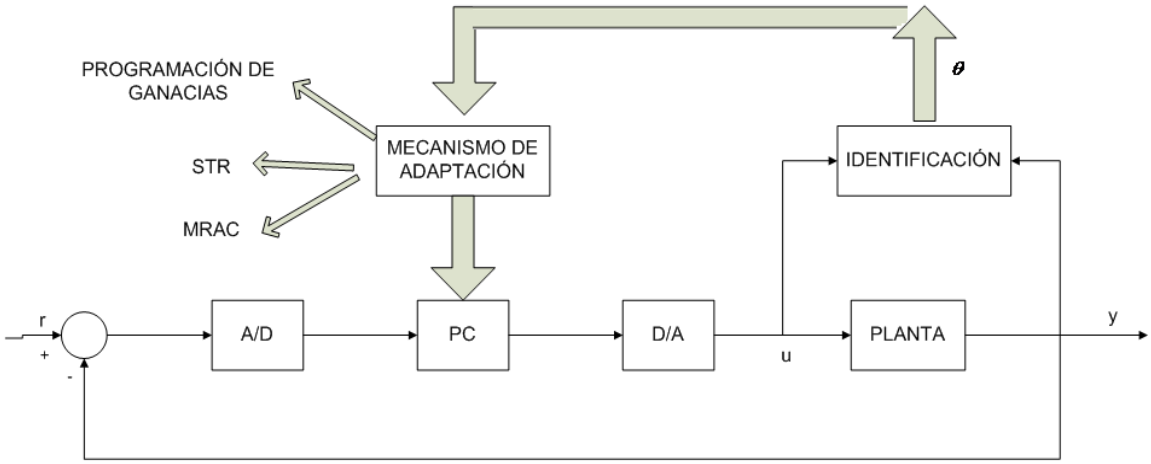


Figura 1.4. Esquema de Control Adaptativo

El proceso de adaptación es regulado por un controlador de parámetros ajustables.

Existen varios métodos para crear el controlador que logran satisfacer los criterios de diseño si el proceso es conocido. Entonces, el problema del Control Adaptativo es encontrar un método de ajuste del controlador si no se conoce la dinámica del proceso.

En el control adaptativo directo, los parámetros del controlador se ajustan sin considerar las características del proceso y las perturbaciones del mismo. En el control adaptativo indirecto, el modelo del proceso y las perturbaciones son determinados previamente. En base a esta información, se diseñan los parámetros del controlador.

El Control Adaptativo tiene muchas aplicaciones entre las cuales se destaca la sintonización automática de controladores, que no es más que los parámetros del controlador sintonizados automáticamente según los requerimientos del operador, y después de dicha sintonización los parámetros permanecen constantes. La ventaja principal de usar un sintonizador automático es que ayuda a mejorar la calidad del control.

1.3. PLATAFORMA DE DESARROLLO

Lo que se observó en la figura 1.1 Control en Tiempo Real, se va a implementar en la práctica, donde lo que se encuentra representado como conversores D/A y A/D es la tarjeta de adquisición de datos NI-DAQ 6008, la misma que realizará el proceso de adquisición de datos de las plantas reales formadas tanto por un circuito pasivo como por un circuito activo. Además se utilizará el programa Matlab versión 2009 para realizar los programas tanto de identificación de parámetros como para el diseño del controlador.

Para realizar la identificación, existen varias opciones, se puede optar por utilizar los Comandos del Matlab, el Simulink, el Toolbox de Identificación y realizar una

Programación en el archivo.m, es decir, el usuario puede escoger el método de cálculo que más le convenga.

Para el manejo de la NI-DAQ se debe primero leer su manual de usuario, el cual se lo presenta como Anexo al final de este Proyecto de Titulación. Para la programación en Matlab se puede recurrir a las ayudas existentes en el mismo paquete computacional, además de lugares de consulta en la Web.

CAPÍTULO II
IDENTIFICACIÓN DE SISTEMAS

La identificación de sistemas se la realiza mediante diferentes métodos numéricos, entre los cuales se pueden mencionar Mínimos Cuadrados Ordinarios, Mínimos Cuadrados Recursivo, Mínimos Cuadrados Extendido.

2.1. IDENTIFICACIÓN USANDO MÍNIMOS CUADRADOS

Karl Friedrich Gauss fue quien formuló el principio de los mínimos cuadrados a finales del siglo XVIII, y lo utilizó para determinar las órbitas de los planetas y los asteroides. Gauss dijo que, de acuerdo a este principio, los parámetros desconocidos de un modelo matemático pueden ser escogidos de una manera tal que: "la suma de la diferencia de los cuadrados entre las observaciones actuales y los valores calculados, multiplicado por el número de mediciones del grado de precisión, es mínima."[2]

Al ser MCO un método de identificación paramétrica, se debe utilizar un modelo de sistema que tiene una característica apropiada para la identificación, el cual es ARMA (AutoRegressive Moving Average), que resulta de utilidad para predecir valores futuros en una sucesión de valores.

El modelo AR es principalmente un filtro de respuesta infinita al impulso IIR (es una sigla en inglés para *Infinite Impulse Response* o *Respuesta infinita al impulso*; se trata de un tipo de filtro digital en el que, como su nombre indica, si la entrada es una señal impulso, la salida tendrá un número infinito de términos no nulos, es decir, nunca vuelve al reposo), con retardo a la salida, que se lo puede describir como:

[2] http://books.google.com.ec/books. MORENO, Luis. "Ingeniería de Control: modelado, análisis y control de sistemas dinámicos"

$$y(k) + a_1.y(k-1) + a_2.y(k-2) + \cdots + a_n.y(k-n) = b_0.u(k)$$

El modelo MA es un filtro de respuesta finita al impulso FIR (es un acrónimo en inglés para *Finite Impulse Response* o *Respuesta finita al impulso*. Se trata de un tipo de filtros digitales en el que, como su nombre indica, si la entrada es una señal impulso, la salida tendrá un número finito de términos no nulos), que presenta retardo a la entrada, el cual se lo describe así:

$$y(k) = b_0.u(k) + b_1.u(k-1) + \cdots + b_n.u(k-n)$$

El modelo ARMA combina modelos AR y MA, y se lo describe así:

$$\begin{aligned} &y(k) + a_1.y(k-1) + a_2.y(k-2) + \cdots + a_n.y(k-n) \\ &= b_1.u(k-1) + b_2.u(k-2) + \ldots + b_n.u(k-n) \end{aligned}$$

Donde $\mathrm{b}_0 = 0$, por el conversor A/D, es decir existe un retardo.
Escrita como Ecuación en Diferencias en el dominio del Tiempo Discreto, en donde se ha despejado la salida, se tiene:

$$\begin{aligned} y(k) = -a_1.y(k-1) - a_2.y(k-2) - \cdots - a_n.y(k-n) + b_1.u(k-1) + b_2.u(k-2) \\ + \ldots + b_n.u(k-n) \end{aligned}$$

Donde el vector de Parámetros $\theta = \begin{bmatrix} -a_1 \\ -a_2 \\ . \\ . \\ . \\ -a_n \\ b_1 \\ b_2 \\ . \\ . \\ . \\ b_n \end{bmatrix}$

El Vector de Información x está dado por:

$$x = [y(k-1) \dots y(k-n)\ u(k-1) \dots u(k-n)]$$

Entonces el Modelo ARMA se lo representa de la siguiente manera:

$$y(k) = x(k).\theta(k) \qquad \textit{PROCESO DETERMINÍSTICO}$$

Para utilizar el método de Mínimos cuadrados se debe tomar como referencia el error de la ecuación:

$$\varepsilon(k) = y(k) - x(k).\theta(k)$$

$$y(k) + a_1 y(k-1) + a_2 y(k-2) + \cdots + a_n y(k-n)$$

$$-b_1 u(k-1) - \cdots - b_n u(k-n) = \varepsilon(k;\theta). \qquad \textbf{(2.1)}$$

La técnica de mínimos cuadrados es un procesamiento de datos en “batch”, es decir, en lotes.

Para trabajar con mínimos cuadrados ordinarios se parte de un conjunto de valores tanto de entrada como de salida, es decir se debe conocer la dinámica del sistema a través de mediciones de entrada y salida.

$$\{u(0), u(1), \dots, u(N), y(0), y(1), \dots, y(N)\}$$

Se desea calcular los valores para

$$\theta = [-a_1 \dots -a_n b_1 \dots b_n]^T$$

el mismo que se ajusta a los datos observados.

El primer error puede ser $\varepsilon(n;\theta)$, ya que $y(k)$ depende de los n primeros valores, y el vector de error queda definido por la ecuación (2.1) para $k = n, n+1, \dots N$. El resultado está dado por la ecuación (2.2).

$$y(n) = \varphi^T(n)\theta + \varepsilon(n;\theta),$$

$$y(n+1) = \varphi^T(n+1)\theta + \varepsilon(n+1;\theta),$$

$$.$$

$$.$$

$$.$$

$$y(N) = \varphi^T(N)\theta + \varepsilon(N;\theta), \qquad \textbf{(2.2)}$$

para esto se asume que el estado del modelo ARMA es el siguiente:

$$\varphi(k) = [y(k-1)y(k-2) \dots u(k-1) \dots u(k-n)]^T$$

Se introduce la siguiente notación matricial:

$$Y(N) = [y(n) \dots y(N)]^T,$$

$$\varphi(N) = [\varphi(n)\varphi(n+1) \dots \varphi(N)]^T,$$

$$\epsilon(N;\theta) = [\varepsilon(n) \dots \varepsilon(N)]^T,$$

$$\theta = [-a_1 \dots - a_n b_1 \dots b_n]^T \qquad \textbf{(2.3)}$$

Donde $\varphi(\mathrm{N})$ es una matriz de $2\mathrm{n}$ columnas por $\mathrm{N} - \mathrm{n} + 1$ filas, por lo que se puede escribir la ecuación de error como:

$$Y = \varphi\theta + \epsilon(N;\theta) \qquad \textbf{(2.4)}$$

Los Mínimos Cuadrados es un procedimiento en el cual se debería tomar el valor de θ para realizar la suma de cuadrados de $\varepsilon(k)$ lo más pequeña posible.

Entonces se define un índice de comportamiento o función de costo que debe minimizarse.

$$J(\theta) = \sum_{k=n}^{N} \varepsilon^2(k;0), \qquad \textbf{(2.5a)}$$

y en términos de (2.4) se tiene:

$$J(\theta) = \epsilon^T(N;\theta)\epsilon(N;\theta) \qquad \textbf{(2.5b)}$$

Lo que se requiere encontrar es $\hat{\theta}_{LS}$, los mínimos cuadrados estimados de $\theta°$, que tiene la propiedad de θ:

$$J(\hat{\theta}_{LS}) \leq J(\theta) \qquad \textbf{(2.6)}$$

Donde:

$J(\theta)$ es una función cuadrática de $2\mathrm{n}$ parámetros en θ y las derivadas parciales de $\mathrm{J}(\theta)$ con respecto a $\theta = \hat{\theta}_{\mathrm{LS}}$ debe ser cero, entonces:

$$\left[\frac{\partial J}{\partial \theta_i}\right] = -2Y^T\varphi + 2\theta^T\varphi^T\varphi = 0 \qquad \textbf{(2.7)}$$

Si se obtiene la transpuesta de (2.7) y $\theta = \hat{\theta}_{LS}$, entonces:

$$\varphi^T\varphi\hat{\theta}_{LS} = \varphi^T Y \qquad \textbf{(2.8)}$$

A estas ecuaciones se las llama *ecuaciones normales* del problema, y mediante su solución se obtendrá $\hat{\theta}_{LS}$. La solución de estas ecuaciones depende de cómo se elige además de cómo se utilizan las señales de entrada $\{u(k)\}$.

El parámetro θ tiene la propiedad de que uno y sólo un valor de θ hace de $\mathrm{J}(\theta)$ mínimo.

Nuevamente para (2.2), en el caso específico de tercer orden, los errores son:

$$y(3) = a_1y(2) + a_2y(1) + a_3y(0) + b_1u(2) + b_2u(1) + b_3u(0) + \varepsilon(3),$$

.

.

.

$$y(N) = a_1y(N-1) + a_2(N-2) + a_3(N-3) + b_1u(N-1) + b_2u(N-2) + b_3u(N-3) + \varepsilon(N),$$

(2.9)

Se asume entonces que las entradas $\mathrm{u(k)}$ están persistentemente excitadas, además que θ es identificable y que $\varphi^{\mathrm{T}}\varphi$ es no-singular; por lo que se escribe:[3]

$$\hat{\theta}_{LS} = (\varphi^T\varphi)^{-1}\varphi^T Y \qquad \textbf{(2.10)}$$

2.2. IDENTIFICACIÓN USANDO MÍNIMOS CUADRADOS RECURSIVO

Debido a que en el control adaptativo los datos son obtenidos secuencialmente en tiempo real, resulta conveniente hacer cálculos de manera recursiva para guardar los datos obtenidos en recursiones anteriores.

La solución de la ecuación (2.10) para el problema de mínimos cuadrados ordinarios puede ser escrita de manera recursiva. Entonces $\hat{\theta}(k-1)$ muestra la estimación de mínimos cuadrados basado en $(k-1)$ medidas. Se asume que la matriz $\varphi^T\varphi$ es no singular para todo k.

$$P(k) = (\varphi^T(k)\varphi(k))^{-1} \qquad \textbf{(2.11)}$$

De la definición de $\mathrm{P(k)}$ en la ecuación (2.11) se obtiene que:

[3] FRANKLIN, Gene F.; POWELL, J. David. "Digital Control of Dynamic Systems". 1980

$$P^{-1}(k) = P^{-1}(k-1) + \varphi(k)\varphi^T(k) \qquad \textbf{(2.12)}$$

Los mínimos cuadrados estimados $\hat{\theta}(k)$ están dados por la siguiente ecuación:

$$\hat{\theta}(k) = P(k)(\sum_{i=1}^{k} \varphi(i)y(i))$$

Al resolver las ecuaciones (2.10) y (2.12) se llega a:

$$\sum_{i=1}^{k-1} \varphi(i)y(i) = P^{-1}(k-1)\hat{\theta}(k-1) = P^{-1}(k)\hat{\theta}(k-1) - \varphi(k)\varphi^T(k)\hat{\theta}(k-1)$$

La estimación a la muestra k puede ser escrita como:

$$\hat{\theta}(k) = \hat{\theta}(k-1) - P(k)\varphi(k)\varphi^T(k)\hat{\theta}(k-1) + P(k)\varphi(k)y(k)$$

$$\hat{\theta}(k) = \hat{\theta}(k-1) + P(k)\varphi(k)(y(k) - \varphi^T(k)\hat{\theta}(k-1))$$

Donde

$$K(k) = P(k)\varphi(k)$$

$$\varepsilon(k) = y(k) - \varphi^T(k)\hat{\theta}(k-1)$$

$\varepsilon(k)$ es el error de predicción.

Para poder continuar es mejor resolver una ecuación recursiva para $P(k)$ que para $P^{-1}(k)$ como en la ecuación (2.12), por lo que se va a utilizar el lema de la Matriz Inversa, que dice:

Dados $A, C, y\ C^{-1} + DA^{-1}B$ matrices cuadradas no singulares. Entonces $A + BCD$ es invertible, y

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

Para aplicar este Lema a $\mathrm{P(k)}$ y utilizando la ecuación (2.12), se tiene:

$A = \gamma P^{-1}(k-1)$
$B = \varphi(k) \equiv \varphi$
$C = \omega(k) \equiv a$
$D = \varphi^T(k) \equiv \varphi^T$

$$P(k) = (P(k-1)^{-1} + \varphi(k)\varphi^T(k))^{-1}$$

$$P(k) = P(k-1) - P(k-1)\varphi(k)\big(I + \varphi^T(k)P(k-1)\varphi(k)\big)^{-1}\varphi^T P(k-1)$$

Esto implica que:

$$K(k) = P(k)\varphi(k) = P(k-1)\varphi(k)\big(I + \varphi^T(k)P(k-1)\varphi(k)\big)^{-1}$$

Las ecuaciones y los cálculos anteriores pueden resumirse en el teorema de la estimación de los mínimos cuadrados recursivos (RLS).

Se asume que la matriz $\varphi(\mathrm{k})$ es de rango completo, que $\varphi^{\mathrm{T}}(\mathrm{k})\,\varphi(\mathrm{k})$ es no singular para todo $\mathrm{k} \geq \mathrm{k}_0$. Dados $\hat{\theta}(\mathrm{k}_0)$ y $\mathrm{P}(\mathrm{k}_0) = \left(\varphi^{\mathrm{T}}(\mathrm{k}_0)\varphi(\mathrm{k}_0)\right)^{-1}$, las ecuaciones de mínimos cuadrados estimados $\hat{\theta}(\mathrm{k})$ son:

$$\hat{\theta}(k) = \hat{\theta}(k-1) + K(k).\varepsilon(k) \qquad \textbf{(2.13)}$$

$$\varepsilon(k) = \left[y(k) - \varphi^T(k+1).\hat{\theta}(k-1)\right] \qquad \textbf{(2.14)}$$

$$K(k) = \frac{P(k-1).\varphi(k+1)}{I+\varphi^T(k+1).P(k-1).\varphi(k+1)} \qquad \textbf{(2.15)}$$

$$P(k+1) = [I - K(k).\varphi^T(k+1)]P(k-1) \qquad \textbf{(2.16)}$$

2.3. IDENTIFICACIÓN USANDO MÍNIMOS CUADRADOS EXTENDIDO

El caso de Mínimos Cuadrados Extendido se basa en el método de MCR pero añadiendo el cálculo de un nuevo coeficiente $\boldsymbol{c_1 \dots c_n}$, el mismo que se presenta debido al ruido existente en el proceso, y se ve representado por un modelo ARMAX.

Entonces el cálculo del vector $\boldsymbol{\theta}$ se extiende a:

$$\theta = \begin{bmatrix} -a_1 \\ -a_2 \\ . \\ . \\ . \\ -a_n \\ b_1 \\ b_2 \\ . \\ . \\ . \\ b_n \\ c_1 \\ c_2 \\ . \\ . \\ . \\ c_n \end{bmatrix} \qquad \textbf{(2.17)}$$

El Modelo ARMAX es el mismo modelo ARMA con una entrada externa que constituye una perturbación o simplemente ruido, es decir es una variable aleatoria.

El modelo ARMAX queda representado por la ecuación 2.18:

$$y(k) + a_1 y(k-1) + \cdots + a_n y(k-n) \\ = b_1 u(k-1) + \cdots + b_n u(k-n) + e(k) + c_1 e(k-1) + \cdots + c_n e(k-n)$$

(2.18)

Donde

$$e(k) + c_1 e(k-1) + \cdots + c_n e(k-n)$$

Representa al Ruido Correlacionado, que no es más que el ruido blanco pasado a través de un filtro, así:

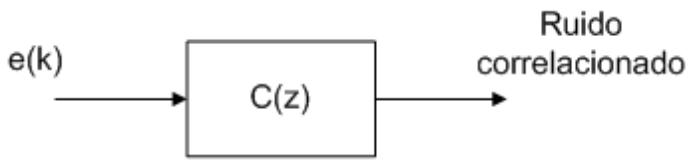


Figura 2.1. Ruido Correlacionado

Modelo ARMAX simplificado:

$$A(z)Y(z) = B(z)U(z) + C(z)e(z)$$ *PROCESO ESTOCÁSTICO*

Donde $A(z), B(z), C(z)$ son los polinomios obtenidos mediante la identificación y $e(z)$ representa el ruido blanco. Los parámetros del polinomio C describen la correlación que existe entre las perturbaciones.

Al ser los errores contenidos en $e(z)$ desconocidos, se los puede obtener mediante aproximaciones, utilizando:

$$\varepsilon(k) = \left[y(k) - \varphi^T(k+1).\hat{\theta}(k-1)\right]$$

Donde el vector θ está descrito por la ecuación (2.17)

Modelo ARMAX con ruido blanco:

$$A(z)y = B(z)u + e(k)$$

En la mayoría de las aplicaciones se puede considerar al ruido simple como ruido blanco en primera aproximación.

2.4. IDENTIFICACIÓN USANDO MATLAB-SIMULINK

Matlab tiene la ventaja de proporcionar varias opciones para realizar una adecuada programación, la cual puede ser muy simple gracias a las librerías que contiene el paquete computacional o puede complicarse un poco más al realizar los programas en archivos con extensión .m. A continuación se explicará cada una de las opciones existentes.

2.4.1. IDENTIFICACIÓN USANDO COMANDOS DEL MATLAB

Para la Identificación de parámetros se utiliza el Comando *iddata*, el cual crea un data object a partir de una entrada, salida de datos y un tiempo de muestreo dados, el formato de este comando es:

dat=iddata (y,u,ts)

Donde:
dat es el nombre con el que se va a reconocer al data object
y es el vector de datos de salida
u es el vector de datos de entrada
ts es el tiempo de muestreo

Una vez que se tiene el data object, se procede a realizar la identificación de parámetros utilizando un comando en Matlab usado para representar modelos en

el Toolbox de Identificación, el mismo que entrega la estimación de *a* y *b* de acuerdo al modelo deseado que puede ser ARMA o ARMAX; éstos valores obtenidos deben ser lo más próximos a los valores del sistema real discretizado.

El formato en Matlab para identificación de parámetros es el siguiente:

sistema=funcion([y u],[nn])

Donde:

sistema es el nombre que va a tener el sistema identificado

funcion es el modelo que se va a utilizar para la estimación

[y u] es el vector de datos de entrada y salida pudiendo ingresar éste como un data object creado con el comando iddata, o simplemente dos vectores de datos uno de entrada y otro de salida.

[nn] son las especificaciones del modelo, es decir el grado del numerador, denominador, y retardos.

Entonces los pasos para realizar identificación por medio de Comandos de Matlab son los siguientes:

1. Ingresar una función de transferencia y discretizarla
2. Generar datos de entrada y salida a partir de la función de transferencia ingresada en el paso anterior. La generación de datos se la realiza creando un vector de datos de entrada utilizando el comando *rand*, después se procede a utilizar el comando *dlsim* para obtener los datos de salida a partir de una función de transferencia dada.
3. Crear un data object con el comando *iddata*
4. Realizar la estimación de los parámetros *a* y *b* con el comando *th*
5. Comparar los parámetros obtenidos en el paso 4 con los verdaderos valores obtenidos al realizar la discretización de la función de transferencia en el paso 1.

Ejemplo:

Si se ingresa la función de transferencia

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

Al ser un sistema de primer orden se deben calcular a_1 y b_1.

Discretizándola con zoh y con un tiempo de muestreo igual a 1 seg. Se obtiene:

$$G(z) = \frac{0.6321}{z - 0.3679}$$

$$G(q) = \frac{0.6321q^{-1}}{1 - 0.3679q^{-1}}$$

Es decir $\boldsymbol{b_1 = 0.6321}$ y $\boldsymbol{a_1 = 0.3679}$

Generando 20 valores de salida y de entrada a partir de G(s), se tiene la tabla 2.1.

Si se desea obtener un modelo ARMA aproximado, se observó después de algunas pruebas que con 11 valores de la tabla anterior, se llegó a los valores de a_1 y b_1 , y se obtuvo lo siguiente:

th= arx([y x],[1 1 1])

Discrete-time IDPOLY model: A(q)y(t) = B(q)u(t) + e(t)

A(q) = 1 - 0.3679 q^-1

B(q) = 0.6321 q^-1

Estimated using ARX

y	x
0	0.8288
0.3750	0.9898
0.6235	0.9759
0.7802	1.0319
0.8830	1.0313
0.9175	0.9135

0.9344	0.9970
0.9570	0.9835
0.9855	1.0628
1.0275	1.1093
1.0613	1.1109
1.0377	0.9136
1.0079	1.0077
0.9791	0.8786
0.9402	0.8886
0.9437	0.9993
1.0007	1.1533
1.0119	0.9230
1.0009	1.0371
1.0025	0.9774

Tabla 2.1 Datos Generados para realizar Identificación

2.4.2. IDENTIFICACIÓN USANDO PROGRAMACIÓN (ARCHIVO.M)

Utilizar los archivos.m de Matlab es muy simple, sólo se debe conocer el lenguaje de programación C; en el presente trabajo se realizaron programas para realizar Identificación con MCO, MCR y MCE, después de haber generado los datos a partir de una función de transferencia ingresada por el usuario.

Además, el usuario tiene la opción de ingresar por teclado los siguientes datos:

Orden del sistema ≡ *n*.
Número de datos a generar ≡ *v*
Número de retardos del sistema ≡ *k*.

El objetivo de los Mínimos Cuadrado es obtener los parámetros a y b del sistema, se debe aclarar que se obtendrán tantos valores de a como sea el orden del sistema, los valores de b quedan restringidos por el usuario. Se debe aclarar que el número de retardos, restringe los valores de b. Así; según la tabla 2.2.

Orden del Sistema	Retardos	Parámetros a calcular
n=na=1	k=1	a_1, b_1
nb=1	k=2	a_1
n=na=2	k=1	a_1, a_2, b_1, b_2
nb=2	k=2	a_1, a_2, b_2

Tabla 2.2 Parámetros a obtener dependiendo del Orden del Sistema y de los Retardos

Se sabe que al momento de realizar la adquisición de datos, se produce un retardo por default, el cual no altera los valores de a, los parámetros b se ven afectados a partir del segundo retardo.

Para el caso de Mínimos Cuadrados Recursivo, el usuario tiene la opción de ingresar:

El número de Iteraciones ≡ *t,* se debe aclarar que el sistema es discreto, pero se ha escogido la letra *t* sólo para el caso de representación. Estas iteraciones representan cuantos vectores theta (θ) de parámetros *a* y *b* se van a obtener.
Para la Identificación con Mínimos Cuadrados Extendido, además de calcular los parámetros *a* y *b* se deberá obtener un parámetro adicional, *c*, el mismo que representa las perturbaciones.

Se realiza la gráfica de la convergencia de los parámetros *a* y *b* para saber en qué valor se estabiliza cada uno de ellos, ya que no se conoce el valor real, y se estima el valor.

2.4.3. IDENTIFICACIÓN USANDO EL SYSTEM IDENTIFICATION TOOLBOX

Matlab presenta un toolbox de identificación GUI (Graphical User Interface) que resulta de mucha utilidad para realizar la estimación de los parámetros *a* y *b* de una manera rápida y sencilla, dicho toolbox requiere de un estudio a profundidad para utilizar todas las herramientas existentes aquí.

Con tan solo escribir el comando *ident* en el workspace, se desplegará una ventana útil para realizar identificación.

En la esquina superior izquierda, en el lugar donde está escrito *Import data*, el usuario puede ingresar un data object creado con anterioridad en el workspace, o simplemente se escoge la opción *Time domain data,* y se ingresará valores de entrada y salida como se observa en la figura 2.2.

En la mitad de la ventana de GUI, se encuentran opciones de las diferentes operaciones que se deseen realizar. El usuario puede escoger entre algunas opciones, para el presente análisis se utilizará *remove means*, es decir que de los datos importados se quitará la media de los mismos; en el siguiente análisis se escogerá *Quick start*, el cual separará los datos obtenidos en dos partes (figura 2.3).

Al recuadro de *working data*, se debe arrastrar el gráfico de los datos a partir de los cuales se va a realizar la estimación de parámetros. En *estimate* se escogerá *Linear parametric models,* es decir el modelo que se va a utilizar, pudiendo ser ARMA, ARMAX (figura 2.4).

A continuación se realizará un ejemplo con las siguientes características:

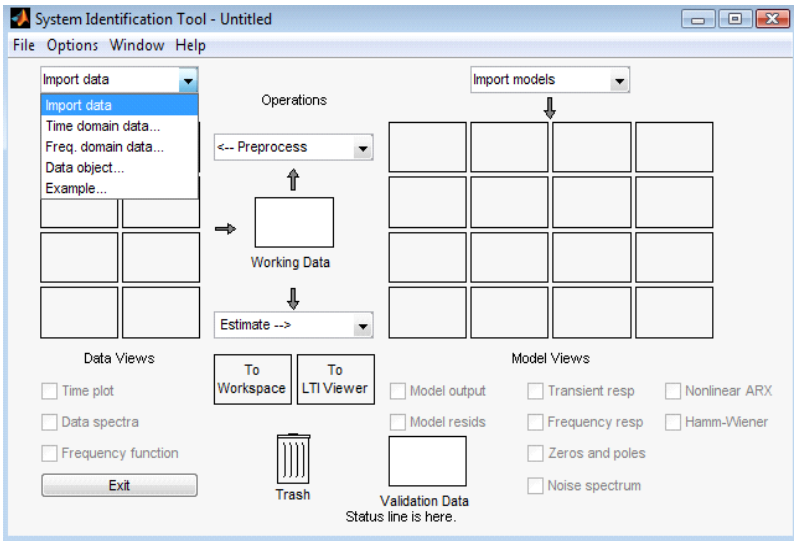


Figura 2.2 GUI Paso 1

Import data: data object

Preprocess : Quick start (es decir separará el data object en dos partes)

Estimate: Linear parametric models

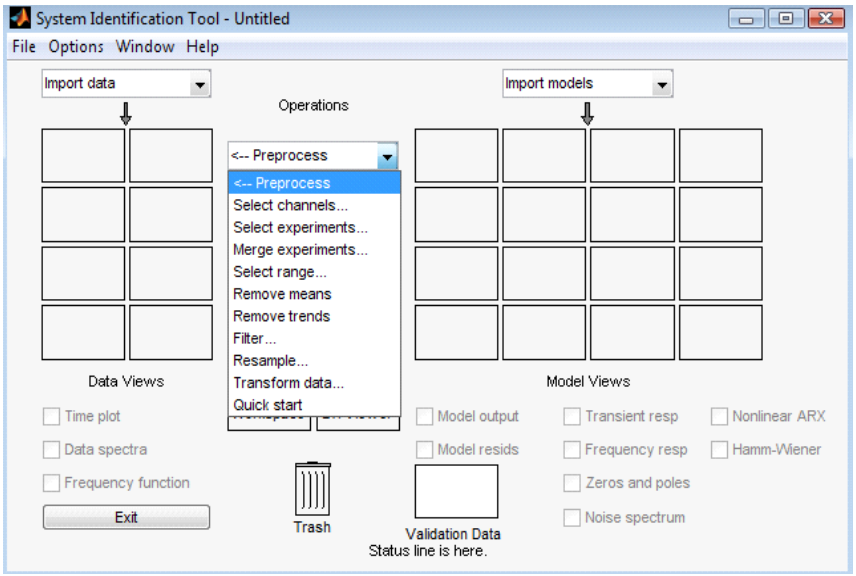


Figura 2.3 GUI Paso 2

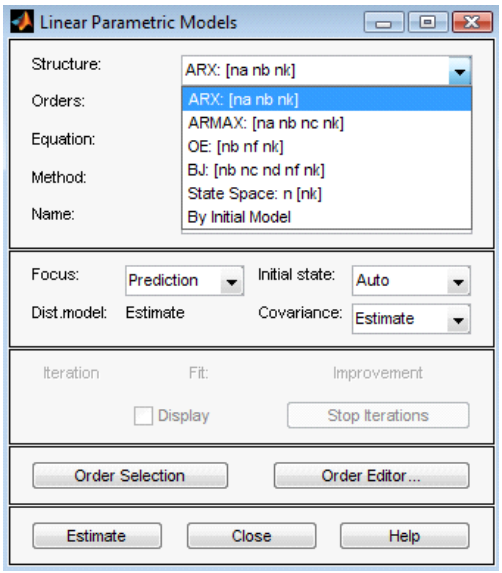


Figura 2.4 GUI Paso 3

1. Se crea un data object de 20 datos en el workspace de Matlab, se debe guardarlo con un nombre para luego importarlo, en este caso el nombre escogido es *dato.* (figura 2.5).

2. Como operación en el Preprocess se escogerá *Quick start,* es decir se separará el conjunto de datos importados en dos partes, para este caso los dos grupos de datos se llaman *datode* y *datodv,* además se visualizará los datos de entrada y salida. (figura 2.6).

3. El modelo a utilizar debe seguir el modelo real. Se realizó varios intentos, en donde:

   modelo1 = arx(datode,[1 1 1]
   modelo2 = arx(datode,[1 2 1]
   modelo3 = armax(datode,[1 1 1 1]

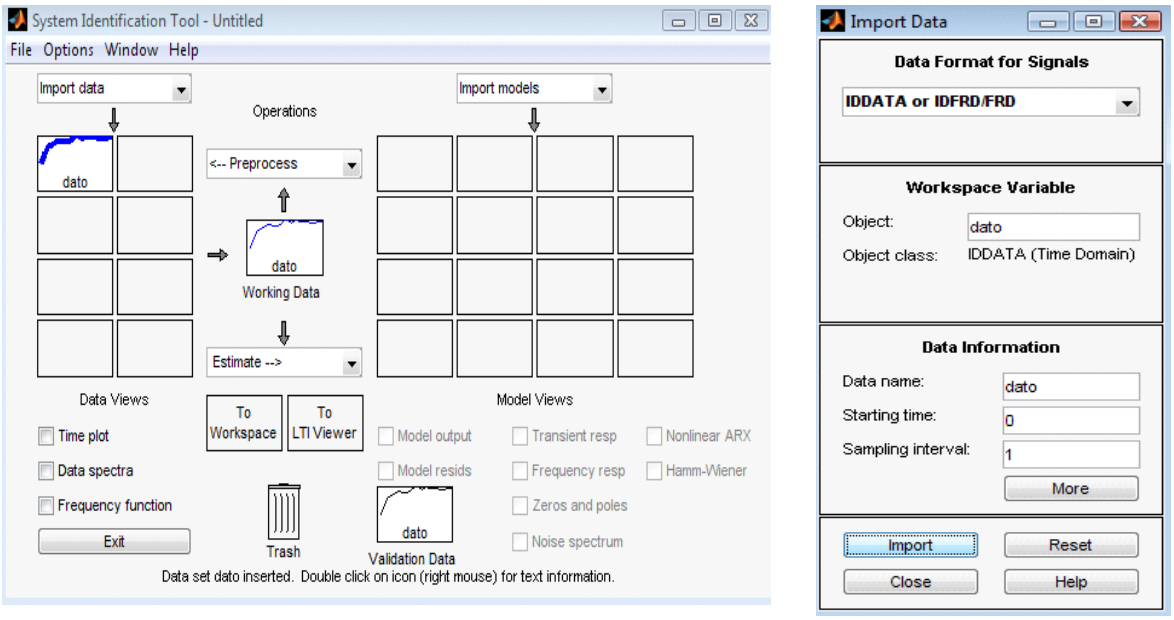


Figura 2.5 GUI Paso 4

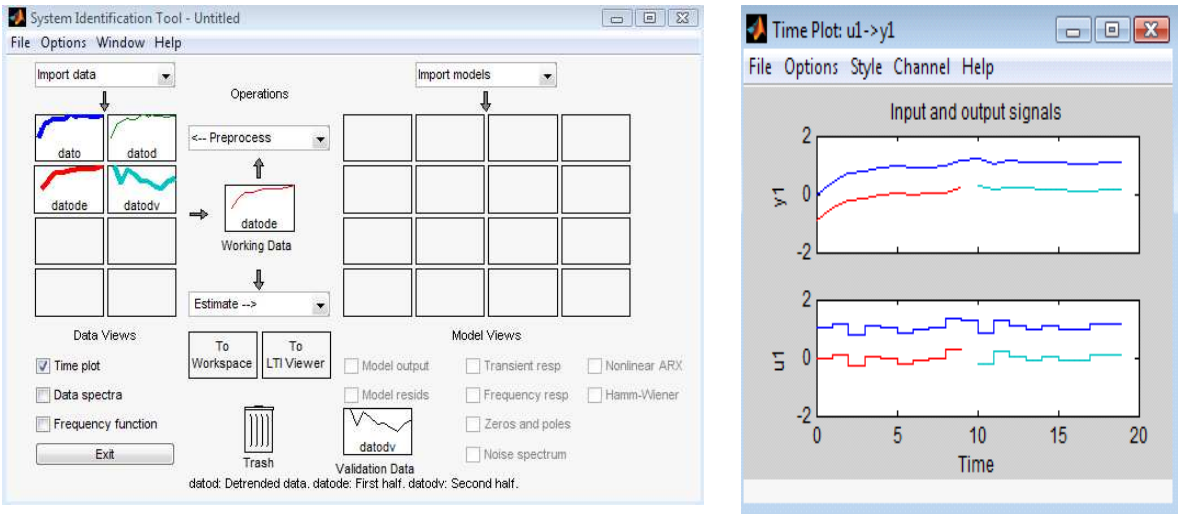


Figura 2.6 GUI Paso 5

Nótese que el conjunto de datos escogido para esta Identificación es el de *datode.* (figura 2.7).

Se observa que el modelo que mejor se aproxima al sistema real, representado con color negro, es el modelo 2 representado con color verde; a la derecha de la gráfica, se puede apreciar valores, los mismos que quieren decir qué tanto se aproxima al modelo real, para el caso del modelo2 se ve que es casi 72 % del modelo original como se puede observar en la gráfica derecha de la figura 2.7.

Figura 2.7 GUI Paso 6

Para observar los valores del modelo estimado, sólo se debe arrastrar el recuadro del modelo hacia el recuadro que dice *To Workspace*, de esta manera se sabrá los valores de los polinomios A y B.

Para la siguiente Identificación, se utilizó el conjunto de datos llamado *datodv,* y los modelos utilizados presentan las siguientes características:

modelo4 = arx(datodv,[1 1 1])
modelo5 = arx(datodv,[2 2 1])
modelo6 = armax(datodv,[2 2 2 1])
modelo7 = armax(datodv,[1 1 1 1])

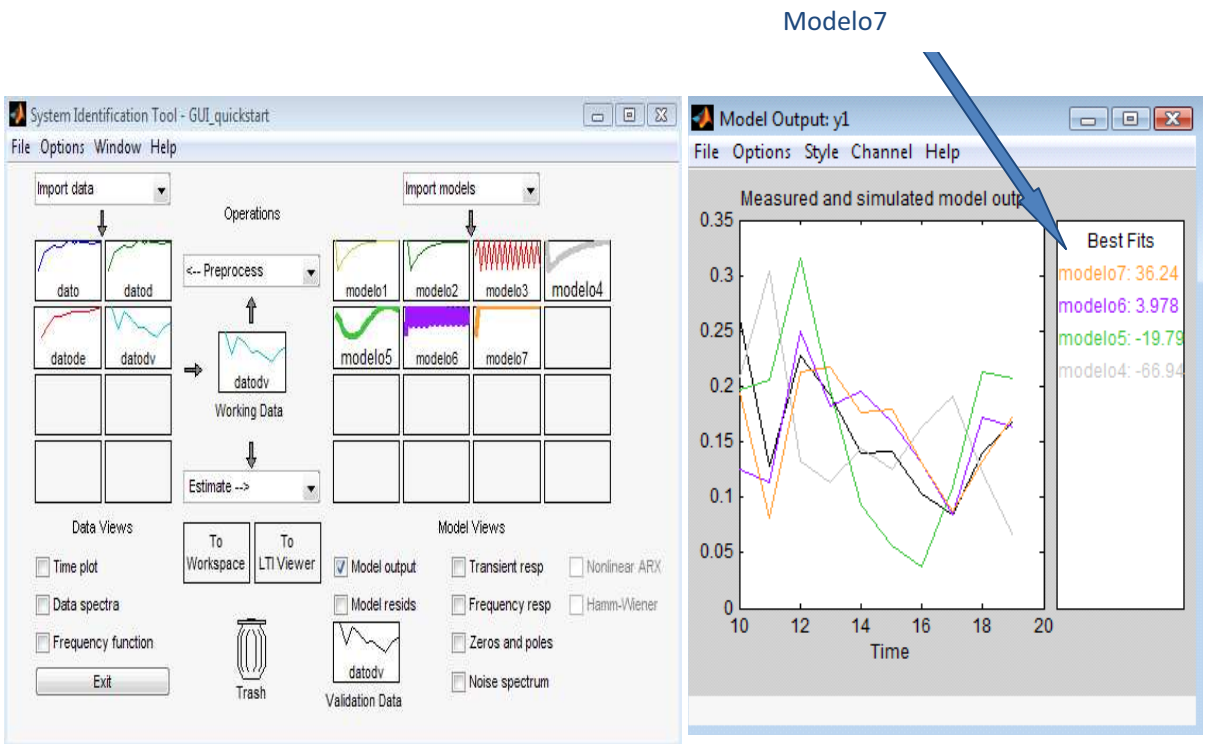


Figura 2.8 GUI Paso 7

Se observa que el modelo que más se aproxima al sistema original es el modelo7.

El siguiente ejercicio se realizó bajo las siguientes condiciones:

Import data: Time domain data
Preprocess : Remove means
Estimate: Linear parametric models

1. Se crean 20 valores de entrada x y salida y, para importarlos al toolbox de ident. (figura 2.9).

2. Como operación se escogerá *Remove media*, es decir del conjunto de datos importados se mueve la media. (figura 2.10).

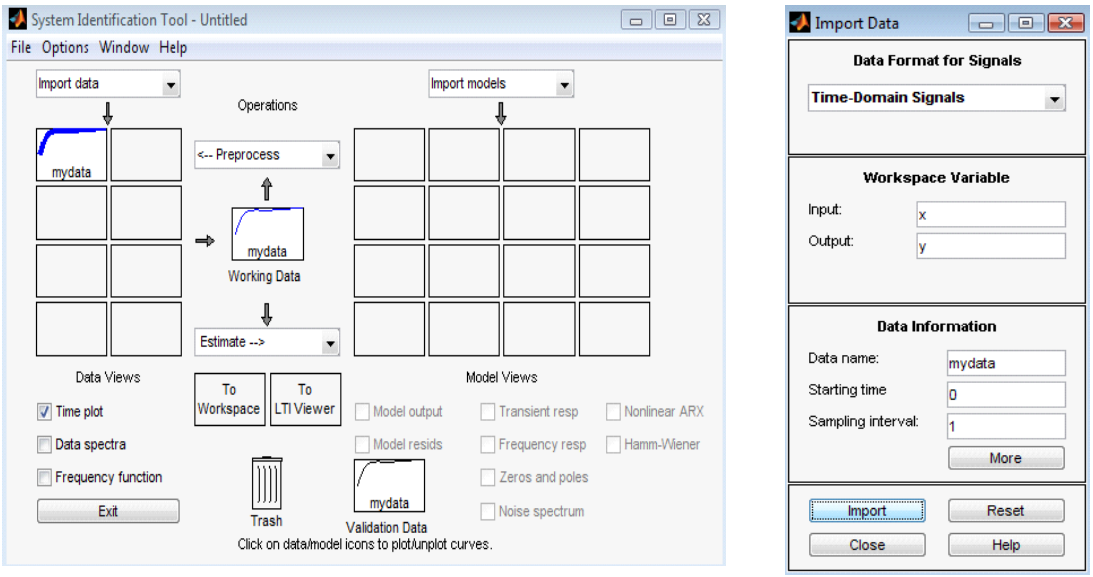


Figura 2.9 GUI Paso 8

3. Después de realizar algunas pruebas con diferentes modelos, se observó que la mejor aproximación da el modelo 6.

modelo1 = arx(mydata,[1 1 1]
modelo2 = armax(mydata,[2 2 2 1]
modelo3 = armax(mydata,[1 1 1 1]
modelo4 = armax(mydata,[2 1 2 1]
modelo5 = arx(mydata,[2 1 1]
modelo6 = arx(mydata,[1 2 1]

2.4.4. IDENTIFICACIÓN USANDO SIMULINK

Simulink es una herramienta de mucha utilidad para la identificación de parámetros, además de ser sencillo su manejo, sólo se debe buscar en la librería System Identification Toolbox, y escoger el bloque que representa el modelo del sistema a utilizar, para el presente proyecto se utiliza el bloque ARX o ARMAX, los cuales representan a un modelo ARMA y ARMAX respectivamente.

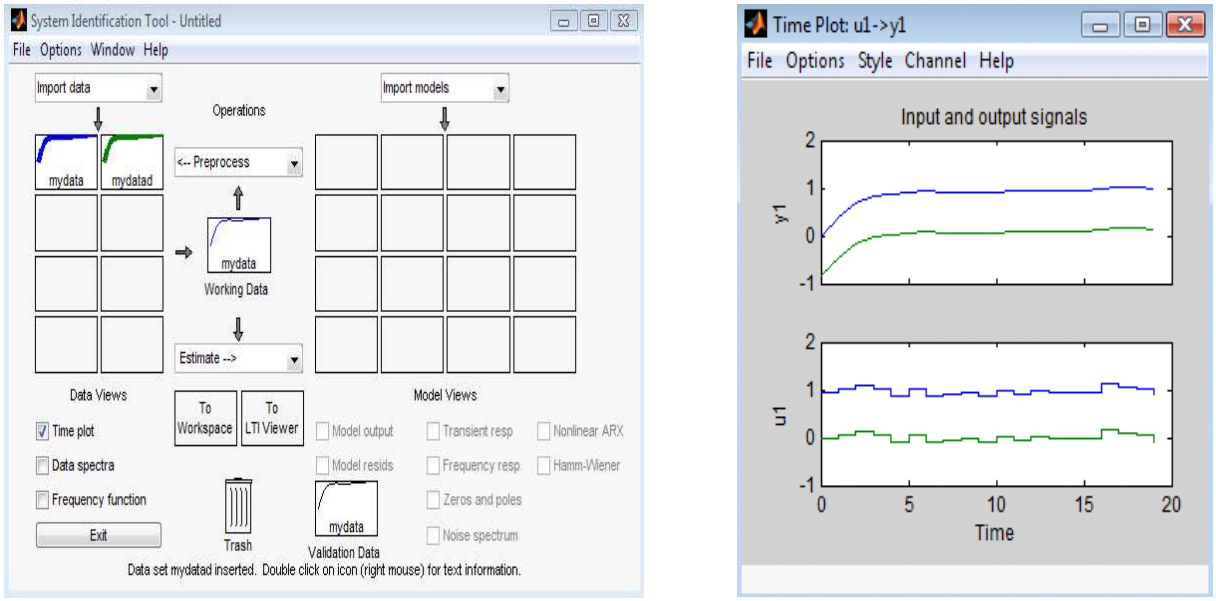
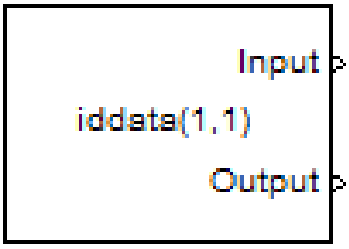


Figura 2.10 GUI Paso 9

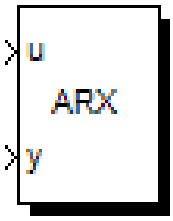
Figura 2.11 GUI Paso 10



Iddata Source1

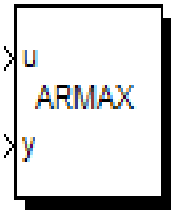
La identificación de parámetros puede realizarse con el Simulink; para lo que se debe crear una entrada y salida de datos en el workspace a través de un *data object*, como se explicó en el literal 2.4.1; este conjunto de datos guardado con un nombre específico, se lo lleva al Simulink a través del bloque *Iddata Source* que se lo encuentra en la librería de *System Identification Toolbox.*

La configuración de este bloque es muy simple, sólo se debe escribir en el recuadro *IDDATA object*, el nombre del data object creado en el workspace.



AutoRegressive with eXternal input model estimator

El bloque *AutoRegressive with external input model estimator,* permite ingresar el modelo a estimar, presenta una entrada *u* y una salida *y,* si el recuadro es *ARX* sólo necesitará ingresar los parámetros *na, nb, nk,* es decir, el orden del numerador, del denominador y el retardo que por default es igual a 1. Si el bloque es ARMAX, los parámetros necesarios son *na, nb, nc nk,* donde *nc* representa el número de coeficientes del polinomio C (ruido).



AutoRegressive Moving Average with eXternal input model estimator

A continuación se presenta un ejemplo, en donde el conjunto de datos de entrada y salida, es creado en el workspace del Matlab. El data object consta de 50 valores tanto de entrada *u* como de salida *y;* generados a partir de un sistema de primer orden representado mediante la siguiente función de transferencia:

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

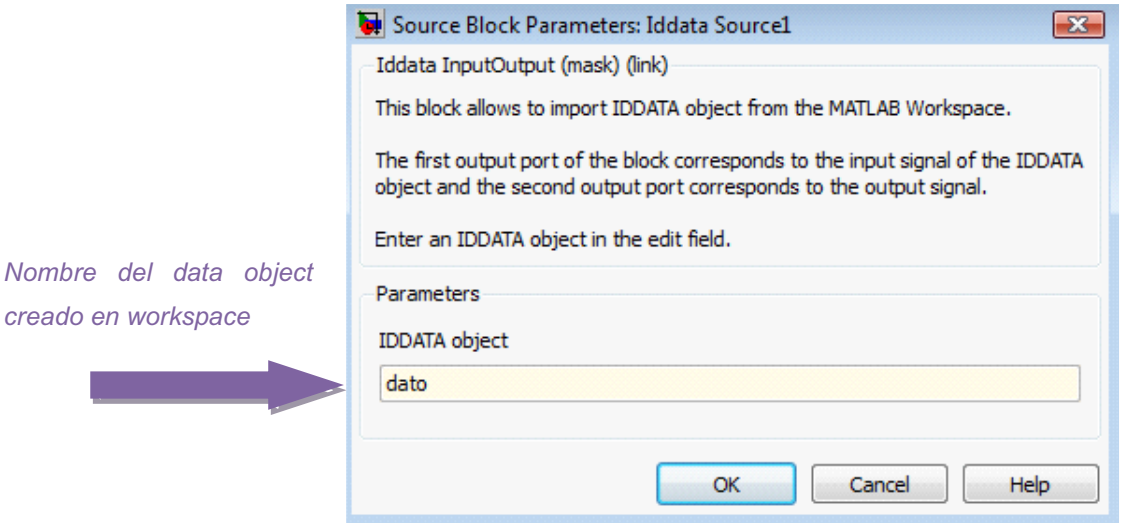


Figura 2.12. Configuración del Bloque de Adquisición de Datos

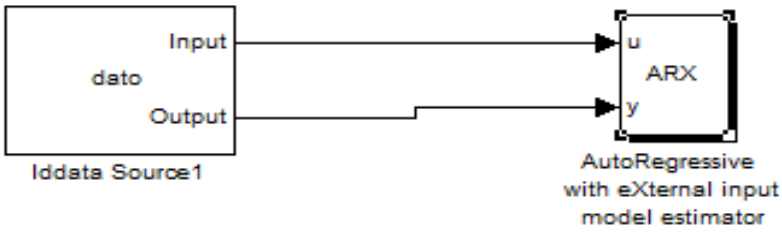


Figura 2.13 Identificación de Sistemas en Simulink usando el modelo ARX

En la figura 2.14 a), se puede observar como la salida estimada representada por el color azul, le sigue a la salida de datos creada en el workspace. En la figura 2.14 b), se cuantifica el error entre los valores estimados y reales, observándose que dicho error es muy pequeño, entre -0.05 y 0.05. El modelo utilizado para la identificación fue el modelo ARMA con los siguientes parámetros: *na*=1, *nb*=1, *nk*=1.

Para el siguiente ejemplo mostrado en la figura 2.15, se utiliza el bloque ARMAX, para identificar los parámetros del sistema.

a)

b)

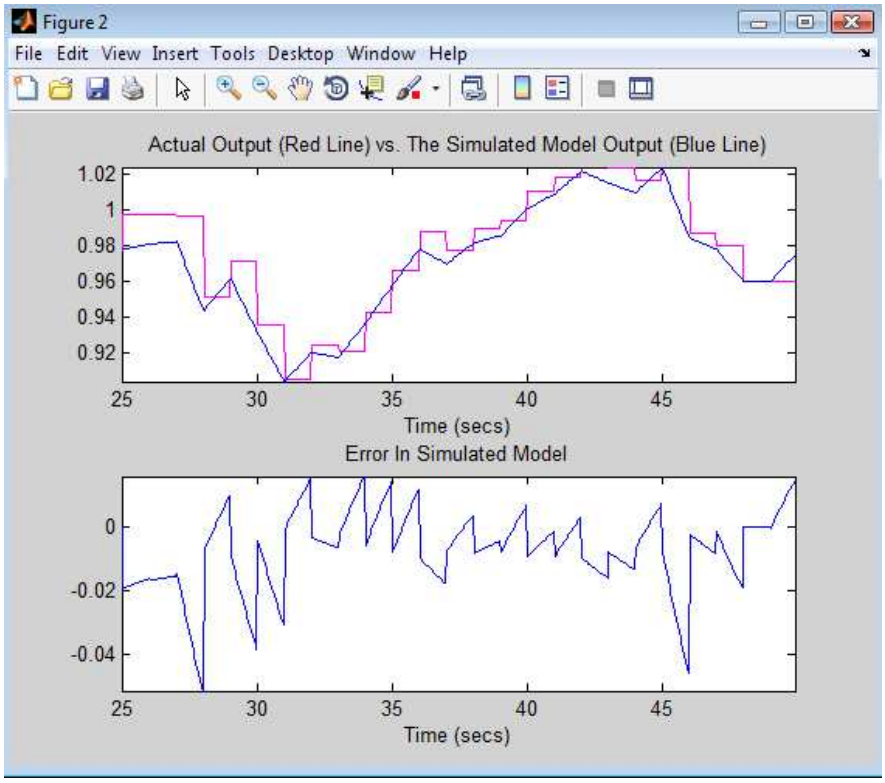


Figura.2.14 a) Salida de Datos Real vs. Salida de Datos Estimada
b) Error entre el Modelo Estimado y el Sistema Real

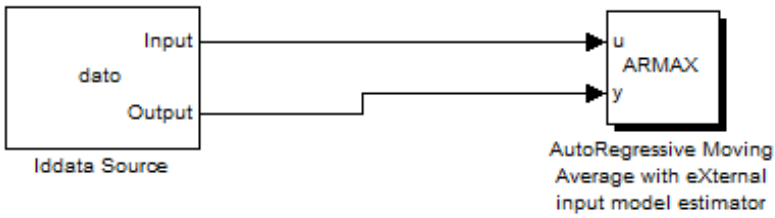


Figura 2.15 Identificación de Sistemas en Simulink usando el modelo ARMAX

A diferencia del ejemplo anterior, el error entre la salida de datos real y la salida de datos estimada, es mucho menor, ya que varía entre -0.04 y 0.03 aproximadamente;

es decir para el sistema de primer orden antes mencionado, el modelo *ARMAX* identifica los parámetros de una manera más aproximada a la real.

Los parámetros del modelo ARMAX utilizados fueron: *na*=1, *nb*=1, *nk*=1, *nc*=1
Una vez explicado los métodos posibles para realizar la identificación de parámetros, se procederá a realizar un ejemplo para todos los métodos anteriormente mencionados.

a)

b)

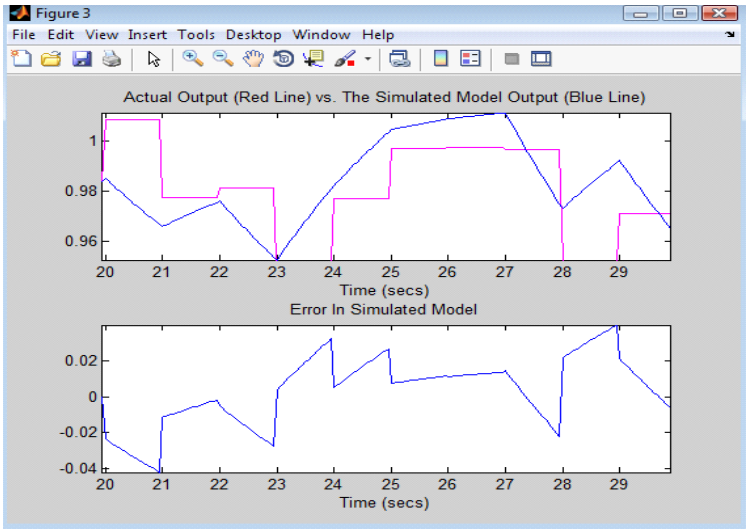


Figura 2.16 a) Salida de Datos Real vs. Salida de Datos Estimada b)Error entre el Modelo Estimado y el Sistema Real

Dado los datos de la tabla 2.3:

k	**u(k)**	**y(k)**
0	1	0
1	-1	1.4
2	-1	-0.42
3	1	-1.694
4	-1	0.2142
5	1	-1.2501

Tabla 2.3 Tabla de Datos

a) Usando Comandos del Matlab

Comandos en Matlab	Comentarios
dato=iddata(y,x,1) Time domain data set with 6 samples. Sampling interval: 1 Outputs Unit (if specified) y1 Inputs Unit (if specified) u1	Se crea un data object con los datos de entrada representados con la letra x, y datos de salida representados con la letra y.
th= arx([y x],[1 1 1]) Discrete-time IDPOLY model: A(q)y(t) = B(q)u(t) + e(t)	Se ingresa el modelo a estimar, con el orden de los parámetros
A(q) = 1 - 0.7 q^-1	Parámetros de *a* obtenidos
B(q) = 1.4 q^-1	
Estimated using ARX Loss function 2.48581e-009 and FPE 4.14301e-009 Sampling interval: 1	Parámetros de *b* obtenidos

Tabla 2.4 Identificación de Parámetros utilizando Comandos de Matlab

b) Usando Programación en archivo.m

Para un sistema de primer orden se realiza la Identificación con MCO:
na=1; nb=1; k=1

En MCO la única fórmula a utilizar es la siguiente:

$$\theta = [X^T * X]^{-1} * X^T * Y$$

Donde:

$$\boldsymbol{X} = \begin{bmatrix} 0 & 1 \\ 1.4 & -1 \\ -0.42 & -1 \\ -1.694 & 1 \\ 0.2142 & -1 \end{bmatrix} \quad ; \quad Y = \begin{bmatrix} 1.4 \\ -0.42 \\ -1.694 \\ 0.2142 \\ -1.2501 \end{bmatrix}$$

Entonces:

$$\theta = \begin{bmatrix} 0.7 \\ 1.4 \end{bmatrix}$$

El resultado indica que el parámetro a1=0.7 y el parámetro b1=1.4.

Para un sistema de primer orden se realiza la Identificación con MCR:

na=1; nb=1; k=1
Iteraciones = t =3
$\boldsymbol{\lambda}$ =0.98, se utilizará un factor de olvido para que la matriz de covarianza P, no se haga cero.

El conjunto de fórmulas a utilizar para Identificar mediante MCR son:

$$\hat{\theta}(k+1) = \hat{\theta}(k) + K(k+1).\varepsilon(k+1)$$

$$\varepsilon(k+1) = \left[y(k+1) - \varphi(k+1).\hat{\theta}(k)\right]$$

$$K(k+1) = \frac{P(k).\varphi^T(k+1)}{1 + \varphi(k+1).P(k).\varphi^T(k+1)}$$

$$P(k+1) = ([I - K(k+1).\varphi(k+1)]P(k))/\lambda$$

Valores Iniciales:

$$\hat{\theta}(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}; I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; P(0) = 1000 * I$$

t=0

$$\varepsilon(1) = \left[y(1) - \varphi(1).\hat{\theta}(0)\right] = 1.4 - [0 \;\; 1] * \begin{bmatrix} 0 \\ 0 \end{bmatrix} = 1.4$$

$$K(1) = \frac{P(0).\varphi^T(1)}{1 + \varphi(1).P(0).\varphi^T(1)} = \begin{bmatrix} 0 \\ 0.999 \end{bmatrix}$$

$$P(1) = ([I - K(1).\varphi(1)]P(0))/0.98 = \begin{bmatrix} 1020.4 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\hat{\theta}(1) = \hat{\theta}(0) + K(1).\varepsilon(1) = \begin{bmatrix} 0 \\ 1.3986 \end{bmatrix}$$

t=1

$$\varepsilon(2) = \left[y(2) - \varphi(2).\hat{\theta}(1)\right] = -0.42 - [1.4 \;\; -1] * \begin{bmatrix} 0 \\ 1.3986 \end{bmatrix} = 0.9786$$

$$K(2) = \frac{P(1).\varphi^T(2)}{1 + \varphi(2).P(1).\varphi^T(2)} = \begin{bmatrix} 0.7136 \\ -0.0005 \end{bmatrix}$$

$$P(2) = ([I - K(2).\varphi(2)]P(1))/0.98 = \begin{bmatrix} 1.0503 & 0.7422 \\ 0.7422 & 1.0397 \end{bmatrix}$$

$$\hat{\theta}(2) = \hat{\theta}(1) + K(2).\varepsilon(2) = \begin{bmatrix} 0.6983 \\ 1.3981 \end{bmatrix}$$

t=2

$$\varepsilon(3) = \left[y(3) - \varphi(3).\hat{\theta}(2)\right] = -1.694 - [-0.42 \;\; -1] * \begin{bmatrix} 0.6983 \\ 1.3981 \end{bmatrix} = -0.0026$$

$$K(3) = \frac{P(2).\varphi^T(3)}{1 + \varphi(3).P(2).\varphi^T(3)} = \begin{bmatrix} -0.4154 \\ -0.4744 \end{bmatrix}$$

$$P(3) = ([I - K(3).\varphi(3)]P(2))/0.98 = \begin{bmatrix} 0.5 & 0.1845 \\ 0.1845 & 0.4066 \end{bmatrix}$$

$$\hat{\theta}(3) = \hat{\theta}(2) + K(3).\varepsilon(3) = \begin{bmatrix} 0.6994 \\ 1.3993 \end{bmatrix}$$

En la figura 2.17 se observa la gráfica que muestra la convergencia de parámetros:

La figura 2.17 conjuntamente con las operaciones realizadas, muestran que el parámetro *a* converge a 0.6994 y el parámetro *b* converge a 1.3994.

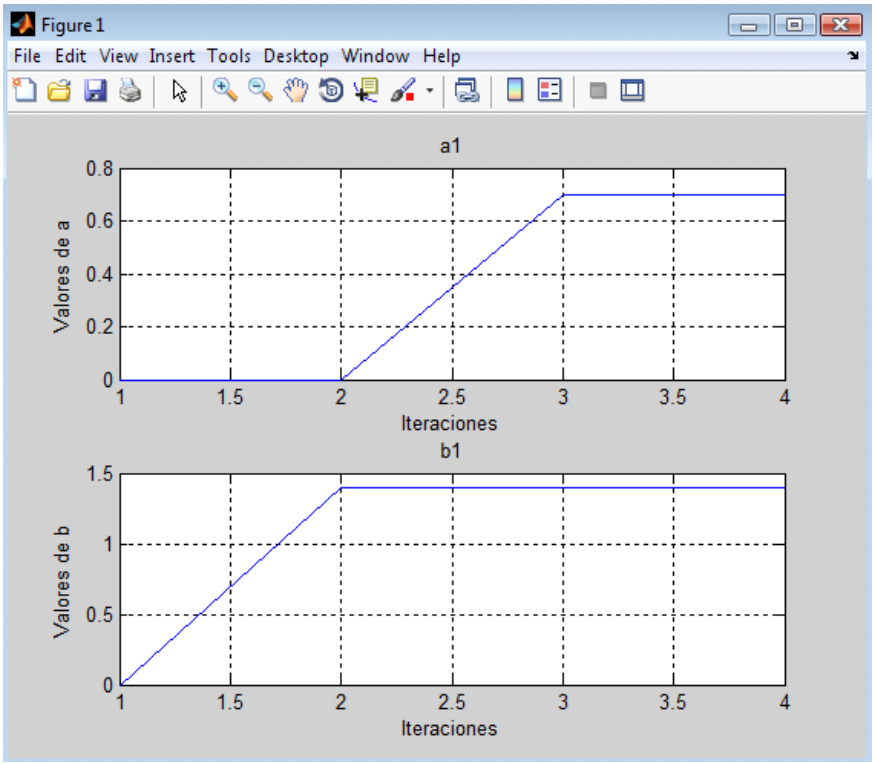


Figura 2.17. Convergencia de Parámetros

c) Usando el Toolbox de Identificación

Ahora para validar los datos obtenidos se utilizará el Toolbox de Identificación de Matlab, como se analizó en el numeral *2.4.3* llamado Identificación usando el System Identification Toolbox.

Como se observa en la figura 2.18, se importaron al toolbox de identificación los valores de entrada *u* y los valores de salida *y*. La estimación de parámetros se realizó para un modelo ARMA cuyos parámetros son: *na=1, nb=1, nk=1.*

A continuación se procede a importar el modelo llamado *ejemplo* al workspace de Matlab, y se obtuvieron los siguientes resultados:

A(q) = 1 - 0.7 q^-1
B(q) = 1.4 q^-1

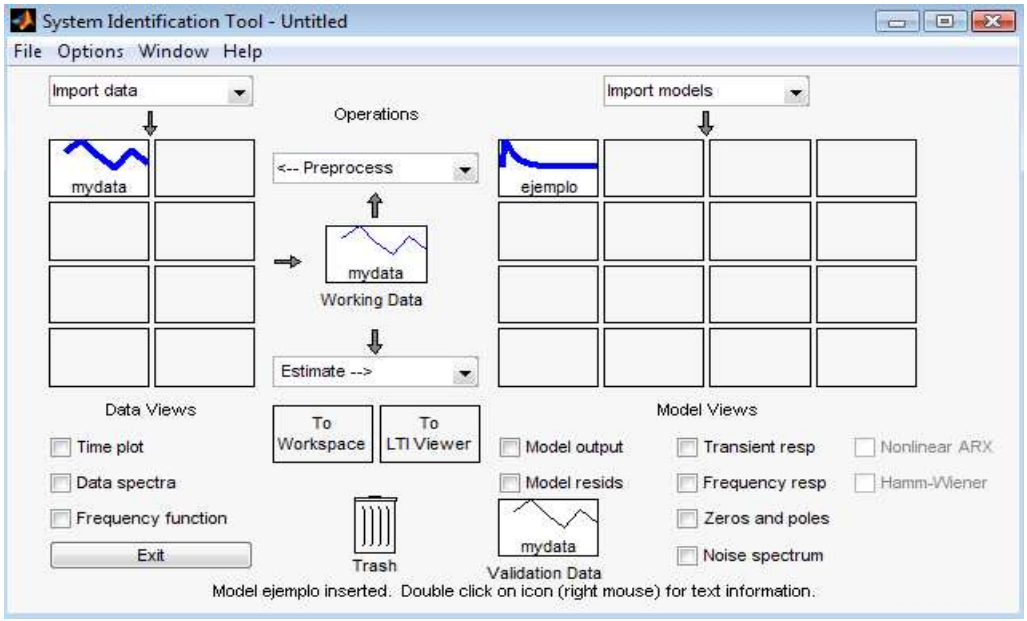


Figura 2.18 Identificación de Parámetros utilizando GUI (Carga de Datos)

Como se puede apreciar, los parámetros $a_0 = 1$ (por default), y $a_1 = 0.7$, $b_0 = 0$ (por default), y $b_1 = 1.4$, son casi los mismos a los obtenidos en el literal a) llamado Identificación de Sistemas Usando Comandos de Matlab.

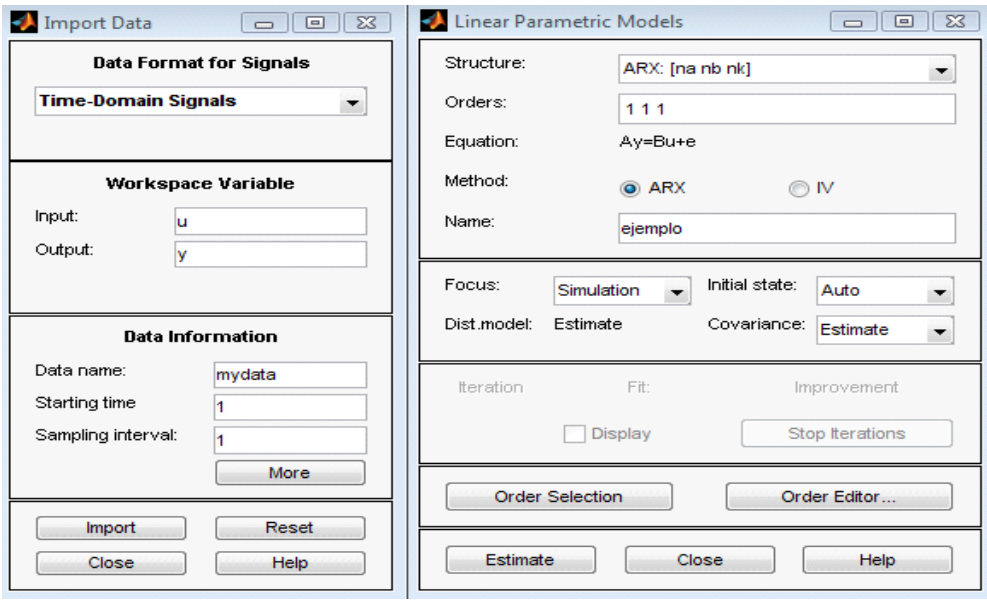
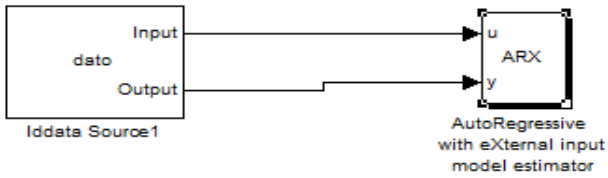


Figura 2.19 Identificación de Parámetros utilizando GUI (Configuración)

d) Usando Simulink



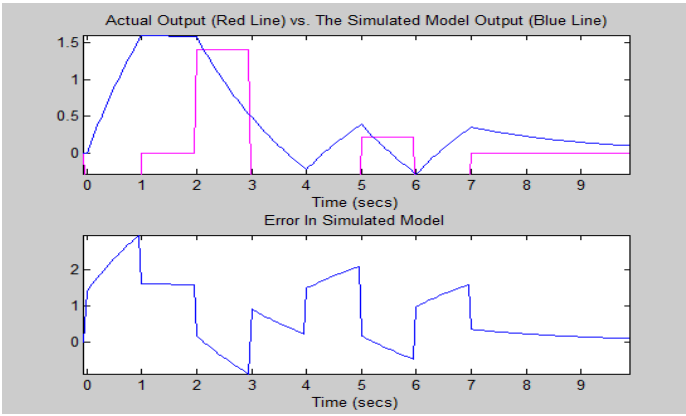


Figura 2.20. Identificación utilizando Simulink

En la figura 2.20 se puede observar como la salida simulada representada con color azul, trata de seguirle a la salida real representada con el color magenta.

CAPÍTULO III
ESTUDIO DEL CONTROL ADAPTATIVO

El Control Adaptativo aparece como una solución capaz de ajustarse a la naturaleza dinámica cambiante del proceso. Es aplicado a sistemas en los cuales los parámetros no varían muy rápido en el tiempo.

Es decir, en este tipo de control, los parámetros de la planta varían continuamente en el tiempo, por lo que es necesario tener un método de estimación que actualice los parámetros de una manera recursiva.

El control adaptativo es utilizado por algunas razones, entre las cuales se tiene:

- Variaciones en la dinámica del proceso
- Variaciones en las perturbaciones
- Ingeniería eficiente
- Fácil de utilizar

3.1. CLASES DE CONTROLADORES ADAPTATIVOS

Como se dijo anteriormente, existen dos clases de controladores adaptativos: los controladores adaptativos indirectos, en los que los parámetros de la planta se ajustan de acuerdo a los datos de entrada y salida; y los controladores adaptativos directos, en los cuales los parámetros del controlador se ajustan de acuerdo a los datos de entrada y salida.

Antes de utilizar un controlador adaptativo, resulta importante conocer cuál es el problema de control a ser afrontado, ya que simplemente podría resolverse dicho

problema con una realimentación de ganancia constante; por lo que se puede seguir el esquema de la figura 3.1.[4]

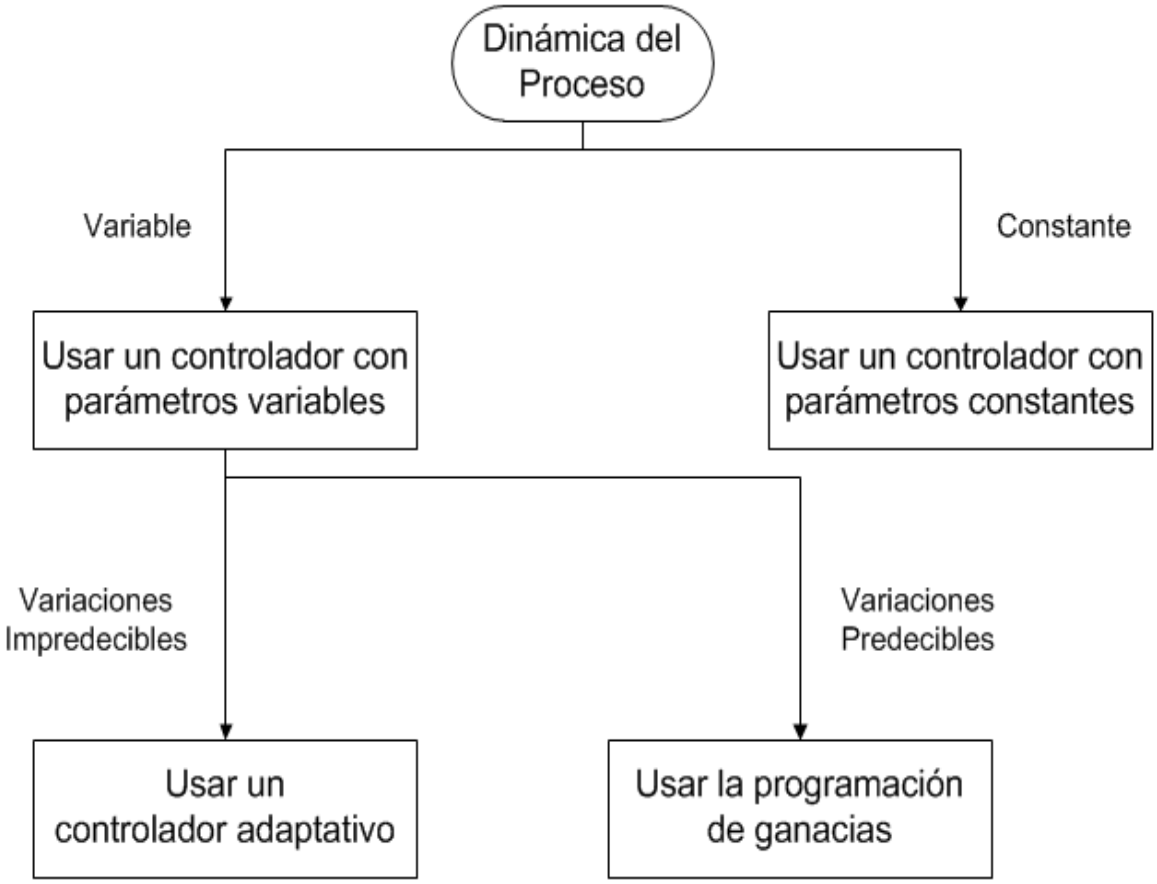


Figura 3.1. Procedimiento para escoger el tipo de controlador a utilizar

El mecanismo de adaptación en este tipo de control se lo puede incorporar mediante técnicas de ajuste tales como Programación de Ganancias, Model Reference Adaptative Control (MRAC) y Self-Tuning Regulator (STR).

[4] Dra. Suset Guerra. V Seminario Euro-Latinoamericano en Sistemas de Ingeniería y su Aplicación a la Producción Energética. Noviembre del 2009.

Programación de Ganancias

Para ciertos sistemas las variables auxiliares trabajan muy bien con las características dinámicas del proceso. Si estas variables son correctamente medidas pueden ser utilizadas para cambiar los parámetros del regulador.

El sistema con Programación de Ganancias se utiliza para variar las ganancias durante el proceso, por lo que se necesita conocer previamente el funcionamiento de la planta, además de una cuantificación de cómo cambiar los parámetros del controlador ante cambios en el proceso.

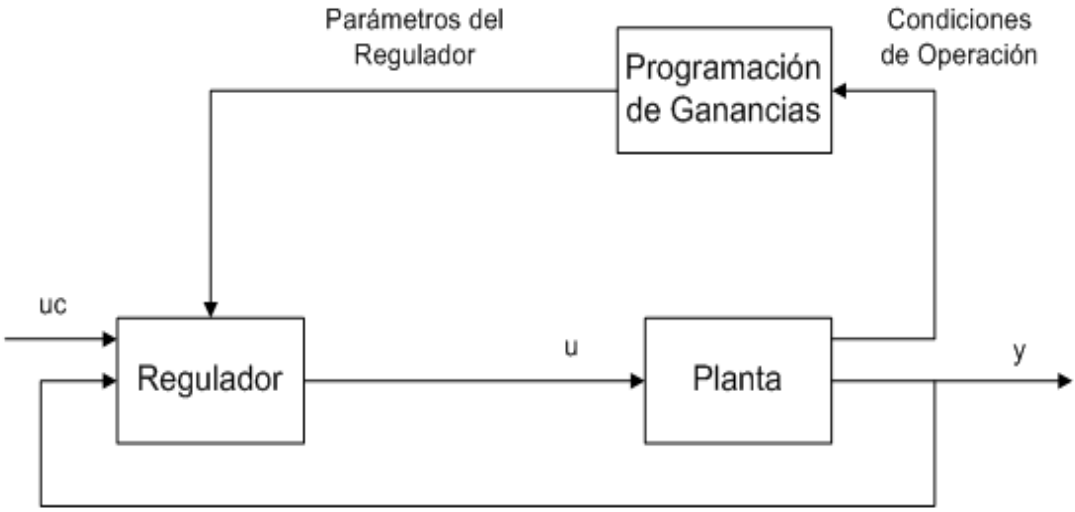


Figura 3.2. Esquema del Sistema con Programación de Ganancia

Para realizar el diseño de un controlador con el método de ganancia programada, se debe seguir los siguientes pasos:

1. Linealización de Actuadores No lineales
2. Programación de ganancias basadas en métodos auxiliares, estos métodos consisten en preparar tablas pre programadas en base a lo que se conoce

de la planta, esto proporcionará los valores que deben tener los parámetros del controlador en función de las variables de entrada y salida.

Control Adaptativo con Modelo de Referencia (MRAC)

Este modelo fue propuesto por H P. Whitaker, J. Yamron y A. Kezer del MIT, al estudiar los servomecanismos con ganancia variable en la publicación "Design of a Model Reference Adaptative System for Aircraft".[5]

Se basa en obtener una respuesta en lazo cerrado que sea idéntica a la dada por el modelo de referencia para la señal de entrada.

Como se puede observar en la figura 3.3, el MRAC está formado por un lazo interno con una realimentación ordinaria constituido por la planta y el regulador, cuyos parámetros son ajustados mediante un lazo externo en donde el error *e* entre la salida del modelo *ym* y la salida del proceso *y* es lo más pequeño posible.

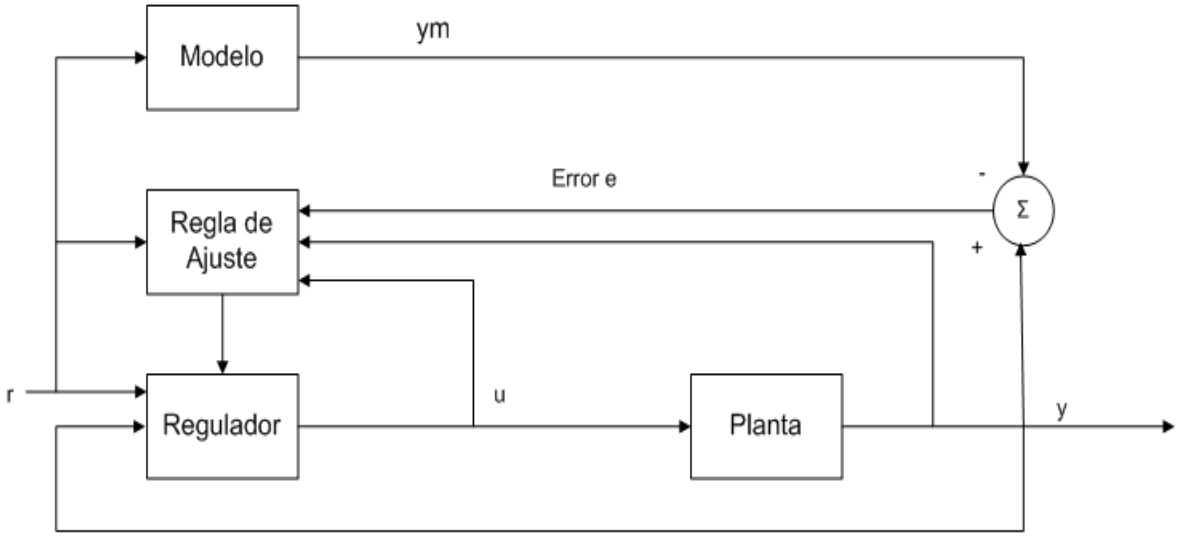


Figura 3.3. Esquema del MRAC

[5] http://automata.cps.unizar.es/Historia/Webs/teoria_moderna_de_control.htm

3.2. REGULADOR AUTOSINTONIZABLE (STR)

Fue planteado por Rudolph Emil Kalman en su artículo “Design of a self-optimizing control system”, en donde utilizaba el método de identificación de Mínimos Cuadrados Recursivo.[6]

El self-tuning regulator es un esquema en donde los parámetros de la planta son actualizados y los parámetros del regulador se obtienen de la solución del diseño del problema.

Para utilizar el Control Autosintonizado se debe conocer el modelo del sistema; las variaciones en el mismo se logran controlar simplemente acoplando un método de diseño de reguladores con un procedimiento de identificación en línea. Sin olvidar que tanto la parte de identificación como la de control deben de estar separadas.

El STR, de una manera más detallada, está formado por la planta en sí y un regulador con realimentación ordinaria; este regulador es ajustado por un estimador recursivo de los parámetros de la planta, cuyo resultado ayudará al cálculo de los valores adecuados de los coeficientes del regulador, este bloque de diseño es una solución on-line para un sistema cuyos parámetros son conocidos.

Este tipo de control adaptativo es llamado autosintonizable para indicar que los parámetros de su controlador tienen la propiedad de converger al controlador que fue diseñado si los parámetros del proceso fueran conocidos.

Lo que se muestra en la figura 3.4, no es más que la secuencia del Control Autosintonizable, cuyos elementos son:

[6] http://automata.cps.unizar.es/Historia/Webs/teoria_moderna_de_control.htm

Estimador: Da una estimación de los parámetros de la planta de manera recursiva.

Diseño del Controlador: Contiene el algoritmo para realizar el diseño del controlador.

Controlador: Es la implementación del controlador, cuyos parámetros son obtenidos del bloque de diseño.

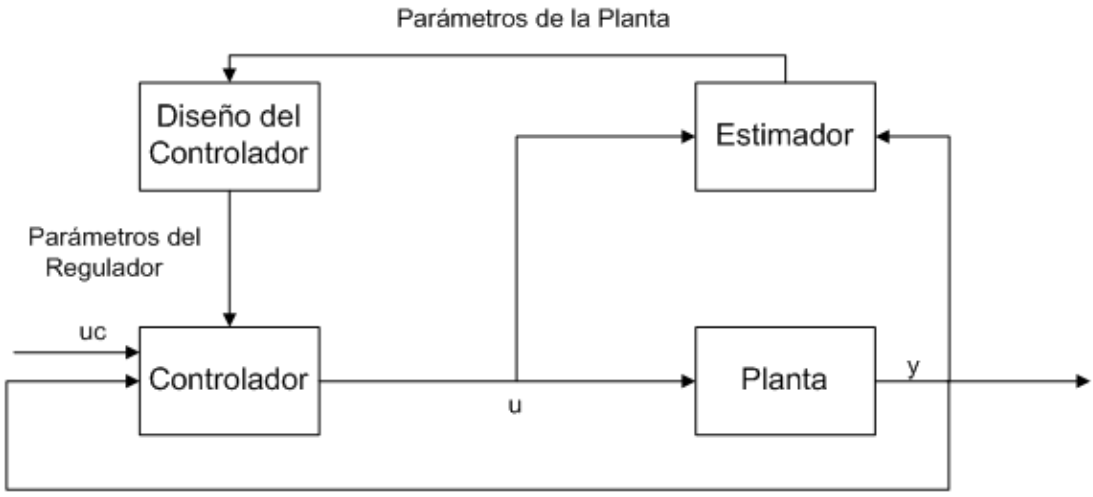


Figura 3.4. Esquema del Regulador Autosintonizable

El STR puede ser modelado con la ayuda de la Transformada Z, como se puede observar a continuación:

$$y(t) = \frac{z^{-k}B(z^{-1})}{1+A(z^{-1})}u(t) + \frac{c(z^{-1})}{1+A(z^{-1})}\xi(t) \qquad \textbf{(3.1)}$$

Donde, $u(t)$, $y(t)(t = 0, +1, +2, \dots)$ son las secuencias de tiempo asociadas con la salida del controlador y con las muestras de la salida del sistema, y $\boldsymbol{\xi(t)}$ es considerado ruido blanco, que no es más que una secuencia discreta no correlacionada.

Donde $\frac{\boldsymbol{C(z^{-1})}}{\boldsymbol{1+A(z^{-1})}}$; modela las perturbaciones.

El modelo de la planta está dado por:

$$\begin{cases} B(z^{-1}) = b_1 z^{-1} + \cdots + b_{n_b} z^{-n_b} \\ \\ A(z^{-1}) = a_1 z^{-1} + \cdots + a_{n_a} z^{-n_a} \end{cases} \quad \textbf{(3.2)}$$

La función de transferencia se caracteriza por el polinomio (3.2), el mismo que representa la dinámica del sistema a ser controlado y está relacionado al sistema de tiempo continuo por medio de la transformada Z. Cualquier retraso asociado con la dinámica de la planta, los cuales son enteros múltiplos del intervalo de muestreo T, están modelados por el término z^{-k}.

$$C(z^{-1}) = 1 + c_1 z^{-1} + c_2 z^{-2} + \cdots + c_{n_c} z^{-n_c} \quad \textbf{(3.3)}$$

La función de transferencia caracterizada por $A(z^{-1})$ y el polinomio $C(z)$ representan la influencia total de las perturbaciones aleatorias a la salida del sistema.

La variable z^{-1} puede ser interpretada como un operador de retraso, entonces la ecuación (3.1) puede ser escrita como una ecuación en diferencias, así:

$$y(t) = -\sum_{i=1}^{n_a} a_i y(t-i) + \sum_{j=1}^{n_b} b_j u(t-j-k) \ + \varepsilon(t) \quad \textbf{(3.4)}$$

Por conveniencia se escribe la perturbación como la ecuación (3.5).

$$\varepsilon(t) = \xi(t) + c_1 \xi(t-1) + \cdots + c_{n_c} \xi(t-n_c) \quad \textbf{(3.5)}$$

El objetivo del Control Autosintonizable es determinar los coeficientes $\{a_i, b_j\}$ de los polinomios $A(z^{-1}), B(z^{-1})$ los mismos que describen al sistema desconocido. En cada intervalo de muestreo estos coeficientes son re-estimados para luego pasar al bloque de diseño.

El controlador obtiene del bloque de diseño los coeficientes $\{f_i, g_i\}$, definidos según las ecuaciones (3.6) y (3.7).

$$u(t) = -\frac{G(z^{-1})}{1+F(z^{-1})} yr - ym \tag{3.6}$$

Donde:

$$\begin{cases} F(z^{-1}) = f_1 z^{-1} + \cdots + f_{n_f} z^{-n_f} \\ \\ G(z^{-1}) = g_0 + g_1 z^{-1} + \cdots + g_{n_g} z^{-n_g} \end{cases} \tag{3.7}$$

Las ecuaciones (3.6) y (3.7) representan un grupo de ecuaciones lineales que tienen una única solución siempre y cuando:

$$n_f = n_a - 1$$

$$n_g = n_b + k - 1$$

El método STR aplicado a un determinado intervalo de tiempo realiza las siguientes acciones:

1. Aplica el método de mínimos cuadrados recursivo para actualizar un modelo de la planta e identificar los parámetros de la misma, basado en el supuesto que las perturbaciones estocásticas son ruido blanco. Suponer que el modelo al tiempo t está dado por:

$$y(t) = -\tilde{A}(z^{-1})y(t) + \tilde{B}(z^{-1})u(t) + \xi'(t)$$

2. Utilizar los coeficientes estimados $\tilde{A}(z^{-1})$ y $\tilde{B}(z^{-1})$ para diseñar los coeficientes del regulador,

$$\left(1+\tilde{A}(z^{-1})\right)\left(1+F(z^{-1})\right)+\tilde{B}(z^{-1})G(z^{-1})=T_2(z^{-1})$$

Donde $T_2(z^{-1})$ define los polos deseados.

3. Aplicar una señal de control de entrada $u(t)$, determinada por:

$$u(t)=-\frac{G(z^{-1})}{1+F(z^{-1})}yr-ym$$

El esquema de la figura 3.5, resuelve el problema del control cuando se desea obtener la salida *y(t),* en presencia de disturbios estocásticos $\xi(t)$, lo más cercano posible al valor teórico de *yr(t).* Además, se debe notar que se incorpora un integrador con la finalidad de que el error de posición sea igual a cero.

3.3. CÁLCULO DE CONTROLADOR

En el lazo del Controlador Adaptativo con los parámetros identificados de la planta, se coloca un integrador con la finalidad de que el error de posición sea igual a cero.

Una vez obtenidos los polinomios A y B estimados de la planta, se procede al cálculo de los polinomios F y G de la siguiente manera:

El siguiente cálculo está hecho para un sistema de segundo orden, en donde se han calculado dos parámetros a y dos parámetros b.

$$W(z^{-1})=\frac{z^{-k}B(z^{-1})}{1+A(z^{-1})}=\frac{z^{-1}(b_1+z^{-1}b_2)}{1+a_1z^{-1}+a_2z^{-2}} \qquad \textbf{(3.8)}$$

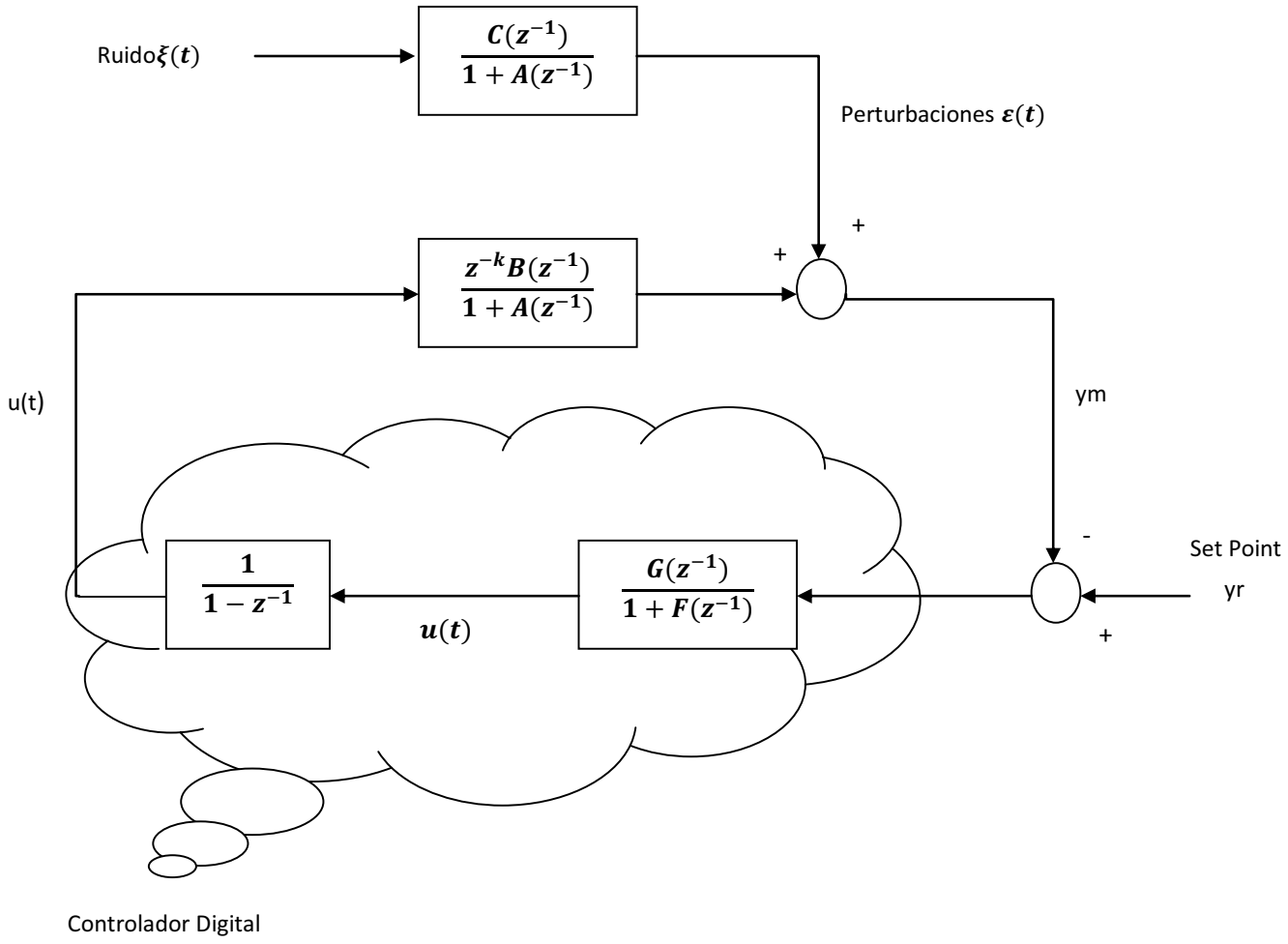


Figura 3.5 Esquema del STR

Entonces:

$$y(t) = \frac{W(z^{-1})}{1-z^{-1}}\, u(t) = W'(z^{-1})u(t) \qquad \textbf{(3.9)}$$

Donde:

$$W'(z^{-1}) = \frac{z^{-1}(b_1+z^{-1}b_2)}{1+\alpha_1 z^{-1}+\alpha_2 z^{-2}+\alpha_3 z^{-3}} \qquad \textbf{(3.10)}$$

Con:

$$\alpha_1 = a_1 - 1 \quad ; \; \alpha_2 = a_2 - a_1; \quad \alpha_3 = -a_2 \qquad \textbf{(3.11)}$$

Por lo que la ley de control obtenida a partir de la figura 3.6 viene dada por:

$$u(t) = \left\{\frac{g_0+z^{-1}g_1+z^{-2}g_2}{1+f_1z^{-1}}\right\}e(t) \qquad \textbf{(3.12)}$$

Además:

$$e(t) = y_r(t) - ym(t) \qquad \textbf{(3.13)}$$

Al combinar (3.12) y (3.9), se tiene la ecuación (3.14) en lazo cerrado:

$$y(t) = \frac{z^{-1}(g_0+z^{-1}g_1+z^{-2}g_2)(b_1+z^{-1}b_2)y_r(t)}{D(z^{-1})} \qquad \textbf{(3.14)}$$

Donde el denominador $\mathrm{D}(\mathrm{z}^{-1})$ es igual a:

$$D(z^{-1}) = (1+\alpha_1z^{-1}+\alpha_2z^{-2}+\alpha_3z^{-3})(1+f_1z^{-1}) + z^{-1}(b_1+z^{-1}b_2)(g_0+z^{-1}g_1+z^{-2}g_2)$$

(3.15)

Los polos deseados en lazo cerrado, corresponden a las raíces de la ecuación característica:

$$T(z^{-1}) = 1 + t_1z^{-1} + t_2z^{-2} + t_3z^{-3} \qquad \textbf{(3.16)}$$

que corresponde al método de asignación de polos.

Para obtener los coeficientes del controlador (f_1, g_0, g_1, g_2), se debe resolver el sistema de ecuaciones (3.17):

$$\begin{bmatrix} 1 & b_1 & 0 & 0 \\ \alpha_1 & b_2 & b_1 & 0 \\ \alpha_2 & 0 & b_2 & b_1 \\ \alpha_3 & 0 & 0 & b_2 \end{bmatrix}\begin{bmatrix} f_1 \\ g_0 \\ g_1 \\ g_2 \end{bmatrix} = \begin{bmatrix} t_1-\alpha_1 \\ t_2-\alpha_2 \\ t_3-\alpha_3 \\ 0 \end{bmatrix} \qquad \textbf{(3.17)}$$

Para obtener una solución única al sistema de ecuaciones dado, se deben dar las siguientes restricciones:

$$n_f = n_a - 1 = 2 - 1 = 1$$

$$n_g = n_b + k - 1 = 2 + 1 - 1 = 2$$

Debe notarse que en el sistema (3.17), se tiene tantas columnas de α como parámetros *f* existan, y tantas columnas de *b* como parámetros *g* hayan.

3.4. SIMULACIÓN

Al modelo de la figura 3.5 se le puede agregar un precompensador $S(z^{-1})$ el cual modela los cambios transitorios en la señal de referencia *yr(t),* antes de que ésta ingrese al sistema; lo cual se observa en la figura 3.6.

El precompensador representado por $S(z^{-1})$ muchas de las veces es simplemente representado por una ganancia.

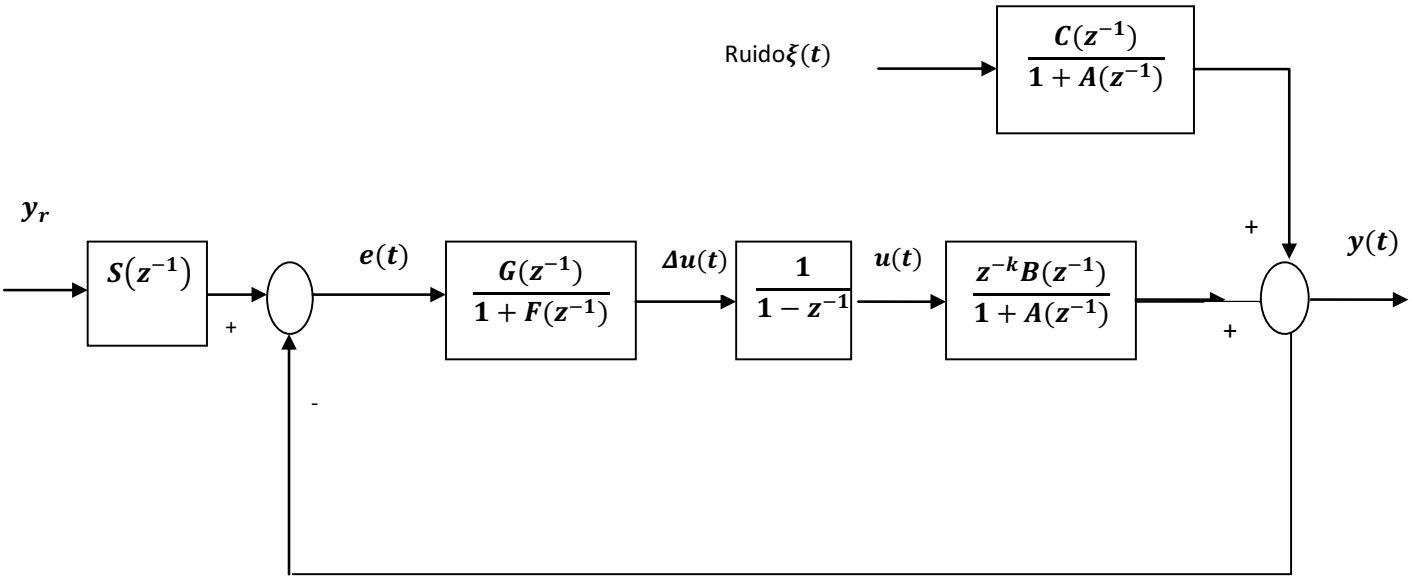


Figura 3.6 Lazo del Control Adaptativo con Integrador y Precompensador

Se debe aclarar que cuando la planta $\frac{z^{-k}B(z^{-1})}{1+A(z^{-1})}$ tiene integrador, no es necesario colocar en el lazo de control el integrador $\frac{1}{1-z^{-1}}$, reduciéndose el esquema de la figura 3.6 al esquema de la figura 3.7.

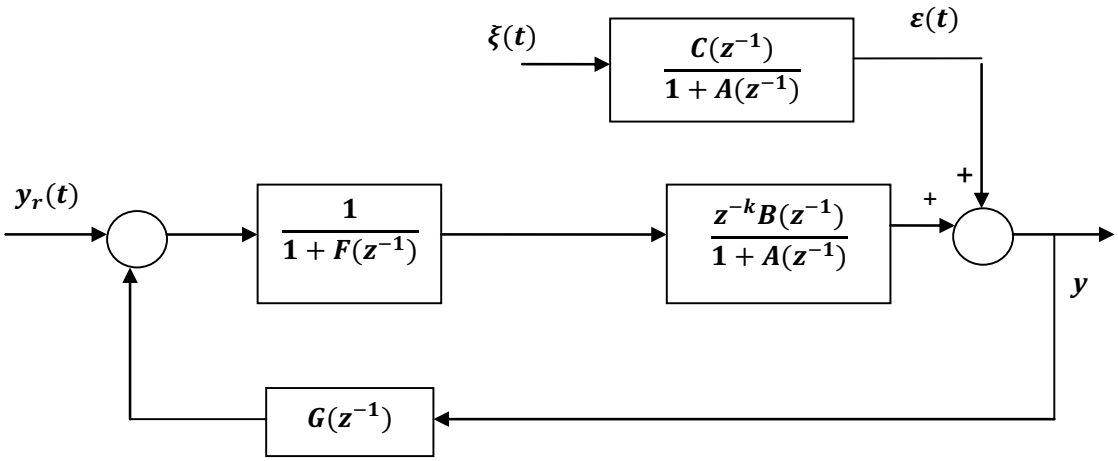


Figura 3.7. Lazo del Control Adaptativo sin Integrador

El objetivo del controlador adaptativo es la reducción de la influencia de las perturbaciones $\varepsilon(t)$, para lo cual se ha visto que existen varias maneras de obtener la acción de control como se puede observar en las figuras 3.5, 3.6, 3.7. Para la elaboración del presente proyecto se ha utilizado el lazo de control que se muestra en la figura 3.5.

CAPÍTULO IV
PLATAFORMA DE DESARROLLO PARA EL CONTROL ADAPTATIVO

4.1. DESCRIPCIÓN DE LA PLATAFORMA EN MATLAB

Para realizar el análisis de circuitos en tiempo real, se implementa dos tipos de circuitos, un circuito activo y un circuito pasivo.

Para poder analizar los circuitos en tiempo real se necesita de una tarjeta de adquisición de datos, para este caso específico se utilizó la NI-DAQ 6008, la cual se comanda por medio del programa computacional MATLAB versión 7.8 de 32 bits, que contiene el toolbox Data Acquisition.

Se debe aclarar que la tarjeta de adquisición de datos NI-DAQ 6008 presenta incompatibilidad con el sistema Windows VISTA de 64 bits, por lo que se procedió a instalar Windows XP con el que trabaja perfectamente. Una vez aclarados los puntos anteriores, se procede a explicar cómo programar algunos parámetros de la NI-DAQ.

La NI-DAQ 6008 tiene la ventaja de ser conexión USB lo que hace más fácil su uso, presenta 8 entradas análogas y 2 salidas análogas.

En las librerías de Simulink se encuentra una específica llamada Data Acquisition Toolbox, la misma que contiene 6 opciones para el uso de la tarjeta, estas opciones son:

- Analog Input
- Analog Output
- Digital Input

- Digital Output
- Analog Input (Single Sample)
- Analog Output (Single Sample)

4.2. ADQUISICIÓN DE DATOS

Para el presente análisis se utilizan Analog Input (Single Sample) y Analog Output (Single Sample), ya que se obtiene un solo dato de entrada y un solo dato a la salida.

En el recuadro de analog input se procede a dar un doble click y en el lugar de *Parámetos*, inmediatamente después de conectar la tarjeta de adquisición en *Device* aparece *nidaq Dev1*, es decir la tarjeta fue reconocida. En la ventana *Input type* se debe escoger si el valor a ingresar es diferencial o entrada simple, en este caso es *single ended*, ya que la tarjeta presenta 8 canales de entrada, se procede a habilitar el número de canales que se vayan a utilizar. Los demás parámetros se dejan como vinieron por default.

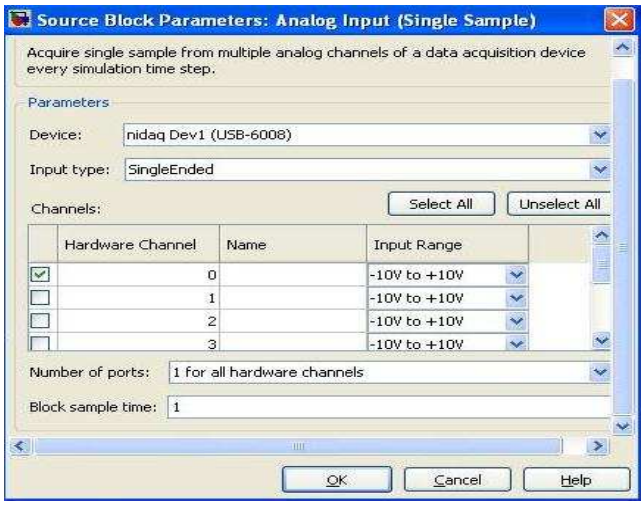


Figura 4.1 Ventana de Canales de Entrada de la Ni-DAQ 6008

4.3. SALIDA DE DATOS

En el recuadro de analog output así mismo se da un doble click y al igual que en el recuadro anterior se configura *Device* y *Channels*, ya que NI-DAQ tiene dos canales de salida.

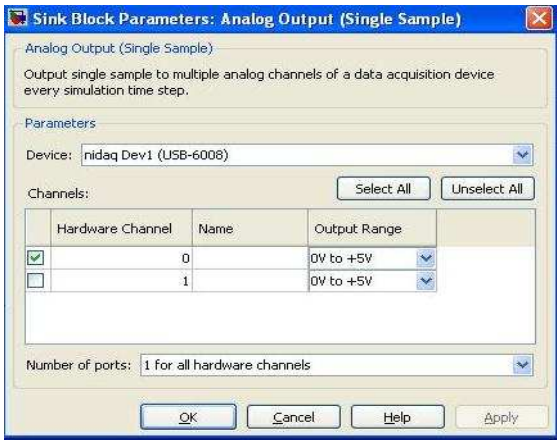


Figura 4.2 Ventana de Canales de Salida de la Ni-DAQ 6008

Una vez configurada la tarjeta USB 6008, se puede realizar las conexiones necesarias para obtener los resultados buscados, es decir, realizar el Control en Tiempo Real de los circuitos Activo y Pasivo.

Como primera prueba se realizó lo siguiente:

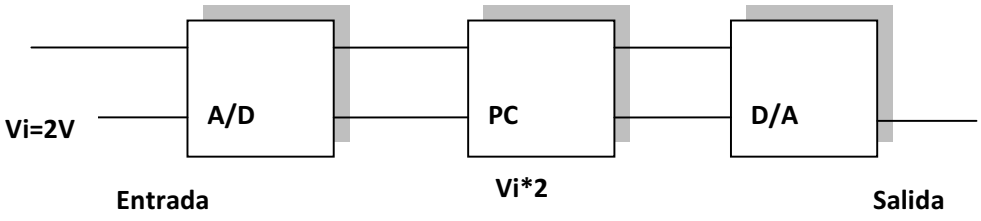


Figura 4.3 Esquema de Entrada y Salida de Datos

La figura 4.3 muestra que se procedió a colocar un voltaje de entrada igual a 2V en la USB 6008; en la PC se colocó una ganancia de 2 y a la salida de NI-DAQ se obtuvieron 4V, esta prueba se la realizó para comprobar que ya se podía obtener datos de la tarjeta y sacarlos por la misma.

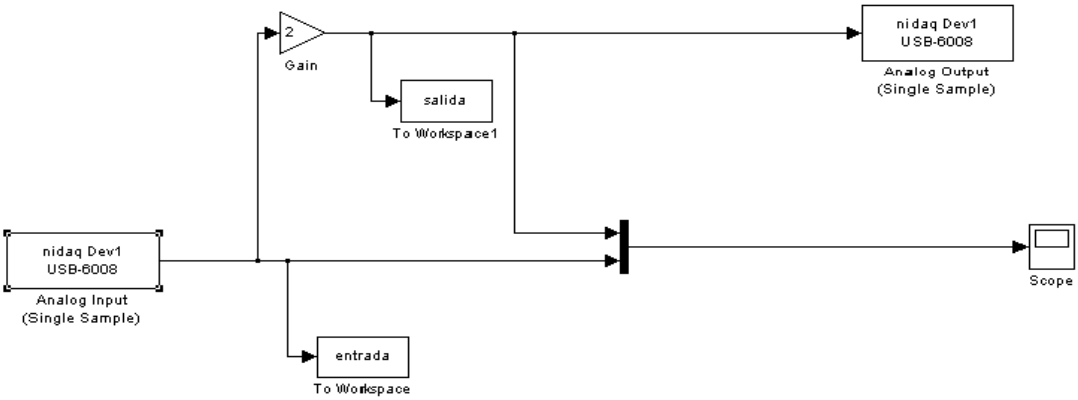


Figura 4.4 Esquema de Entrada y Salida de Datos en Simulink

Como se dijo anteriormente, en la PC se colocó una ganancia de 2, esto se lo hizo en Simulink, como se puede observar en la figura 4.5, además se colocó un osciloscopio para poder observar tanto la entrada de 2V como la salida de 4V.

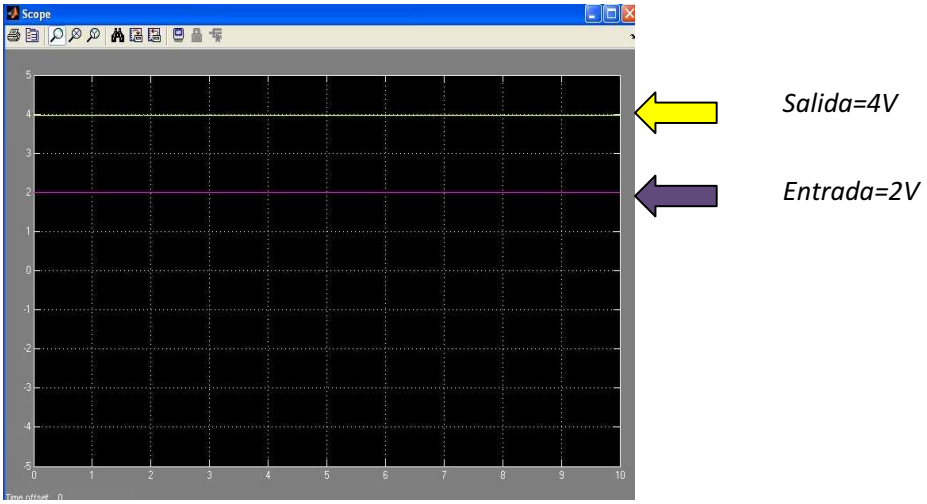


Figura 4.5 Gráfica de Entrada y Salida de Datos

4.4. CONTROL EN TIEMPO REAL

Una vez que se puede obtener y entregar datos a la tarjeta de adquisición, se procedió a realizar pruebas con circuitos tanto activos como pasivos.

CIRCUITO PASIVO:

El circuito es un doble filtro RC. ($\mathbf{R = 5k\Omega;\ C = 1000uF}$)

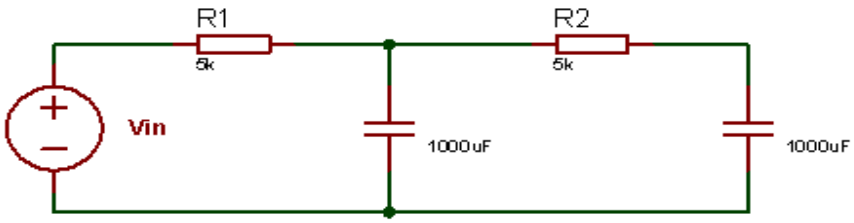


Figura 4.6. Circuito RC

La razón de escoger los valores de R y C anteriormente mencionados es porque con éstos valores se tendrá una respuesta del sistema lento con una constante de tiempo igual a 5 seg.

$$\mathbf{T = RC = 5k\Omega * 1000uF = 5seg}$$

Para obtener la función de transferencia del circuito de la figura 4.7, se plantean las ecuaciones matriciales de nodo:

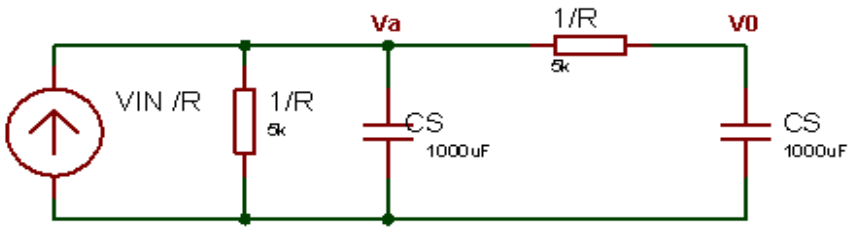


Figura 4.7. Circuito Equivalente

$$\begin{bmatrix} \left(\frac{2}{R}+Cs\right) & -\frac{1}{R} \\ -\frac{1}{R} & \left(\frac{1}{R}+Cs\right) \end{bmatrix} \begin{bmatrix} Va \\ Vo \end{bmatrix} = \begin{bmatrix} \frac{Vin}{R} \\ 0 \end{bmatrix}$$

$$Vo = \frac{\begin{vmatrix} \left(\frac{2}{R}+Cs\right) & \frac{Vin}{R} \\ -\frac{1}{R} & 0 \end{vmatrix}}{\begin{vmatrix} \left(\frac{2}{R}+Cs\right) & -\frac{1}{R} \\ -\frac{1}{R} & \left(\frac{1}{R}+Cs\right) \end{vmatrix}} = \frac{\frac{Vin}{R^2}}{\left(\frac{2}{R}+Cs\right)\left(\frac{1}{R}+Cs\right)-\frac{1}{R^2}} = \frac{Vin}{(2+CsR)(CsR+1)-1}$$

$$\frac{Vo}{Vin} = \frac{1}{(CsR)^2+3CsR+1} = \frac{1}{R^2C^2s^2+3RCs+1} = \frac{1}{25s^2+15s+1}$$

$$G(s) = \frac{1}{25s^2+15s+1}\ ; H(s) = 1 \qquad \text{LazoAbierto}$$

$$G(s) = \frac{1}{25s^2+15s+2} \qquad \text{LazoCerrado}$$

En la figura 4.8 se puede observar la respuesta de simulación en lazo abierto del sistema basado en un circuito RC, observando un tiempo de establecimiento de 53.3 seg, por lo que en el siguiente capítulo se procederá a realizar el diseño de un Controlador Adaptativo tanto en Simulación como en Tiempo Real.

Al igual que con el circuito RC, se analizará la respuesta del sistema basado en amplificadores operacionales.

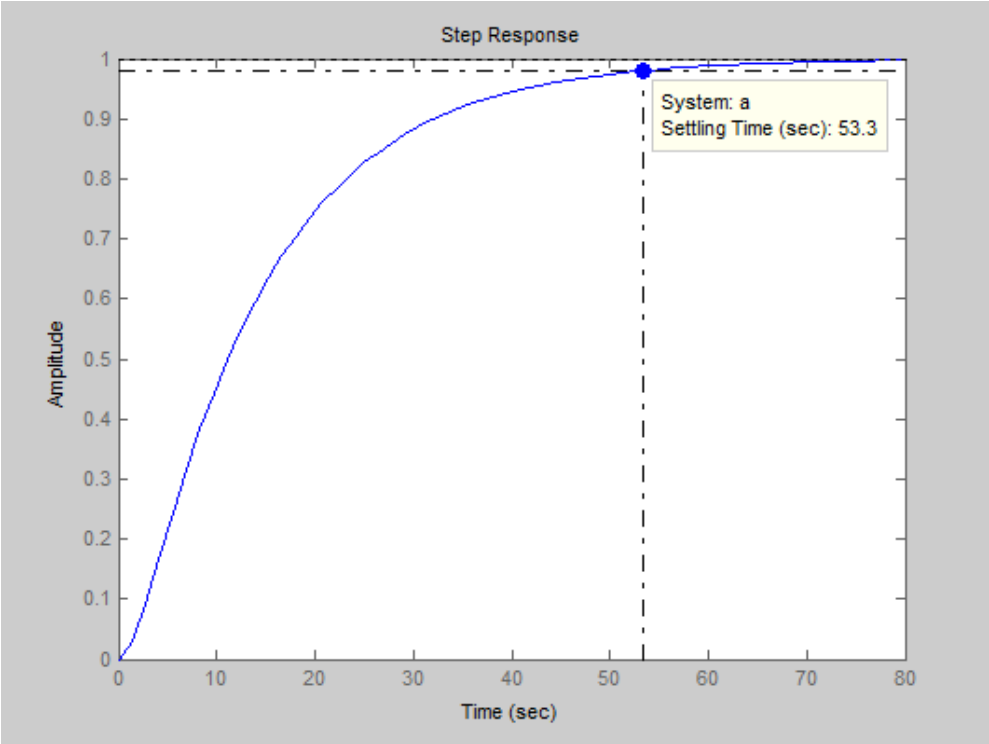


Figura 4.8 Análisis de G(s)en Lazo Abierto

<u>CIRCUITO ACTIVO:</u>

Dada la función de transferencia:

$$\mathrm{Gp} = \frac{0.2}{s^2 + 0.7s + 0.1}\ ; \mathrm{H(s)} = 1 \qquad \mathrm{LazoAbierto}$$

$$\mathrm{Gp} = \frac{0.2}{s^2 + 0.7s + 0.3} \qquad \mathrm{Lazo\ Cerrado}$$

$$(s^2 + 0.7s + 0.1)y = 0.2u$$

$$\ddot{y} + 0.7\dot{y} + 0.1y = 0.2u$$

$$\ddot{y} = -0.7\dot{y} - 0.1y + 0.2u$$

$$\ddot{y} = -0.35\dot{y} - 0.05y + 0.1u$$

Este cambio se debe a que se le ha cambiado la constante de tiempo $\tau = 0.5$ con la finalidad de que el sistema sea más lento.

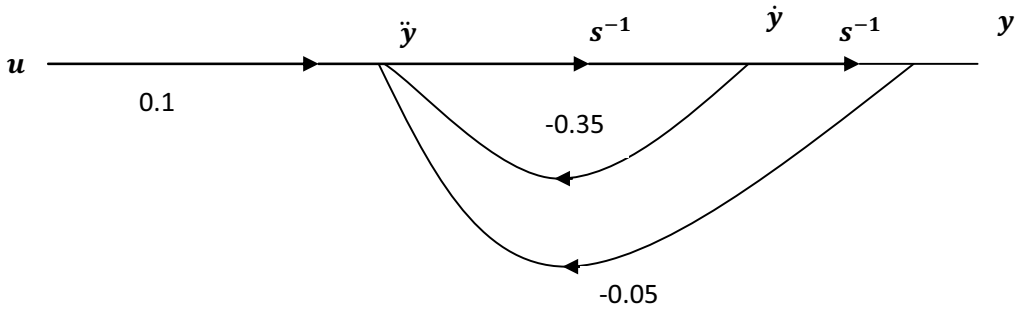


Figura 4.9 Diagrama de Flujo

Análisis del sistema:

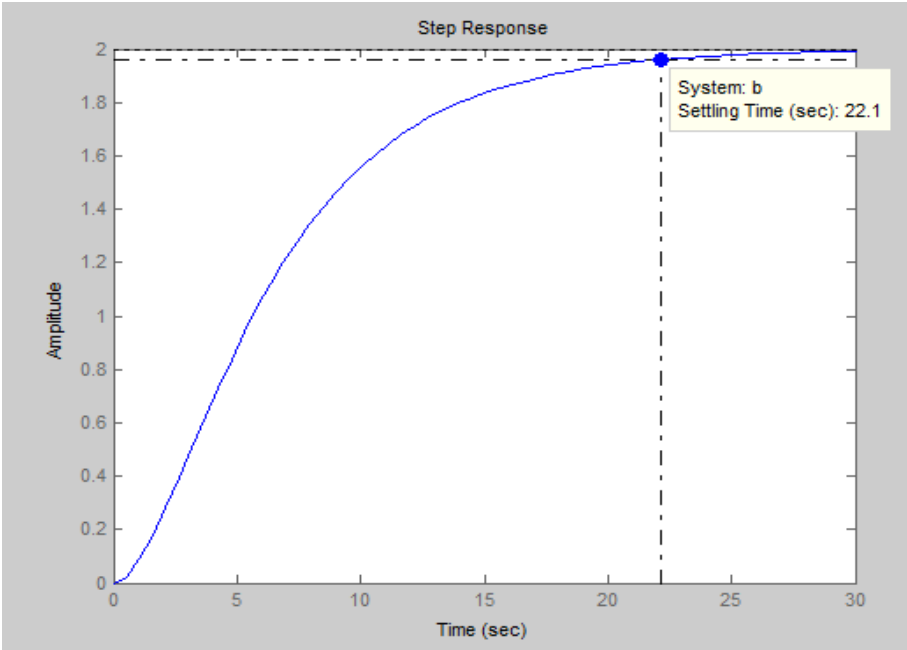


Figura 4.10 Análisis de Gp en Lazo Abierto

Como se puede apreciar en la figura 4.10, la respuesta del sistema basado en amplificadores operacionales es lenta, con un tiempo de establecimiento igual a 22 seg. Asimismo, en el capítulo V se procederá a diseñar un Controlador Adaptativo Self-tuning para mejorar la respuesta del sistema.

Se utiliza la ecuación diferencial para implementar la función de transferencia Gp usando amplificadores operacionales, como se observa en la figura 4.11.

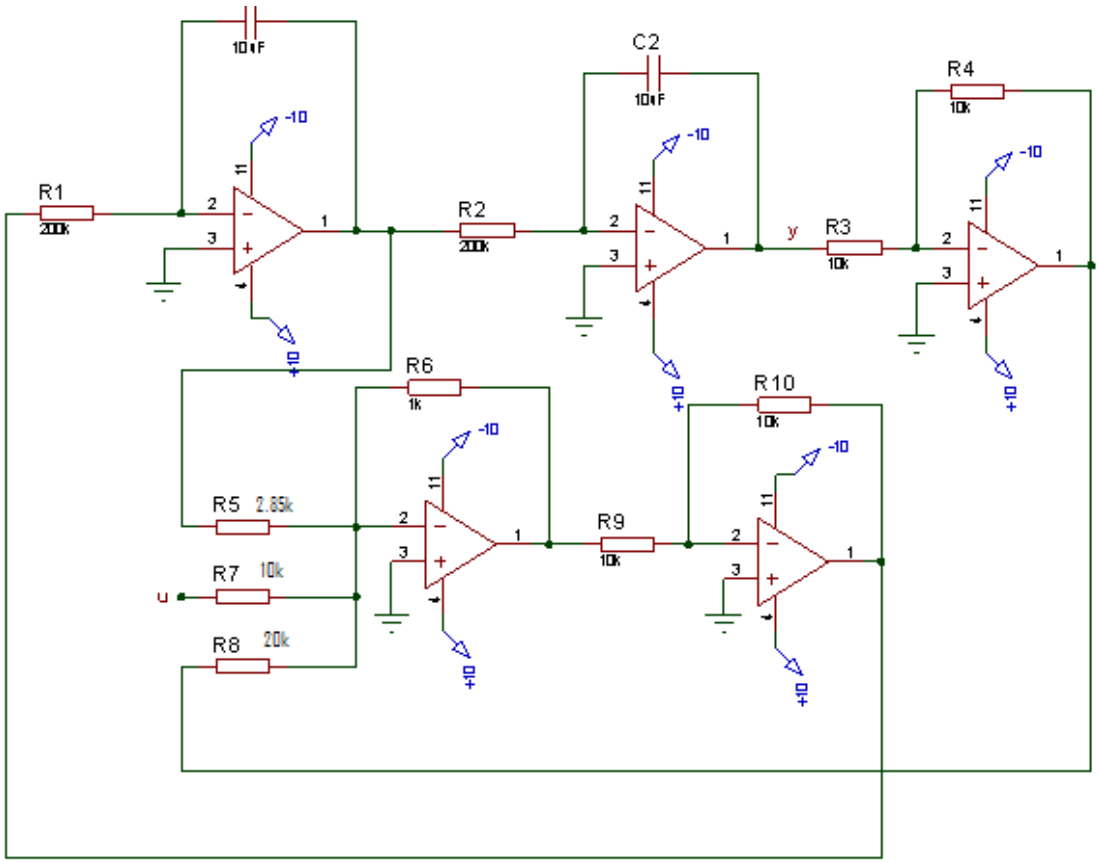


Figura 4.11. Diagrama del Circuito Activo

CAPÍTULO V
PRUEBAS Y RESULTADOS

En el presente capítulo se procederá a realizar pruebas tanto en Simulación como en Tiempo Real, asimismo se realizará pruebas de Identificación y pruebas del diseño del Controlador Adaptativo Self-tuning.

5.1. RESULTADOS DE SIMULACIÓN

Resultados de Identificación

Dada la función de transferencia:

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

Se generan datos a partir de $G(s)$, en esta ocasión se genera 40 datos utilizando la rutina de generación mostrada en la figura 5.1.

Figura 5.1 Ventana para Generación de Datos implementada en Guide

Los datos generados están tabulados en la tabla 5.1.

X	Y
0.2422	0
1.7184	0.0517
0.7867	0.4253
0.3312	0.6257
1.2616	0.5859
1.5161	0.7074
1.9556	0.9309
-1.9979	1.2080
1.4618	0.5983
0.4503	0.5420
1.9598	0.5941
0.1107	0.8764
-0.0819	0.8148
1.2054	0.5634
-1.0886	0.6356
-0.0076	0.3175
1.6034	0.1184
0.2986	0.4151
1.3807	0.5013
0.9546	0.6768
0.3439	0.8018
-1.0131	0.7227
0.6657	0.3177
-1.6661	0.2641
0.5038	-0.1179
0.6438	-0.1296
0.9190	0.0861
1.5630	0.3197
1.9292	0.6451
1.0761	1.0096
0.3258	1.1164
1.7133	0.9498
0.3204	1.0538
-1.9321	0.9559

-1.5166	0.2834
1.4508	-0.3135
-0.0628	-0.0642
1.3794	0.0712
-1.1624	0.3462
0.2092	0.1217

Tabla 5.1 Datos Generados a partir de la Función de Transferencia G(s)

Al ser éste un sistema de segundo orden, se debe encontrar a_1, a_2, b_1, b_2, se procede a hacer el cálculo de dichos parámetros por medio de MCO, y MCR sólo para comparar los resultados.

Con el método de lo mínimos cuadrados ordinario, se obtiene la matriz THETA_MCO, formada por los cuatro parámetros a identificar:

$$\text{THETA_MCO} = \begin{bmatrix} 0.7555 \\ -0.0498 \\ 0.2133 \\ 0.0810 \end{bmatrix}$$

Se debe aclarar que los valores de a obtenidos en la matriz THETA_MCO para su posterior uso deben de ser cambiados el signo.

Es decir $a1 = -0.7555; a2 = 0.0498; b1 = 0.2133; b2 = 0.0819.$

Con el método de mínimos cuadrados recursivo, como su nombre lo dice, se realiza el cálculo de manera iterativa, para este caso específico se ha escogido 38 iteraciones, mostrando a continuación los valores obtenidos en a última iteración.

$$\text{THETA_MCR} = \begin{bmatrix} 0.7541 \\ -0.0487 \\ 0.2133 \\ 0.0813 \end{bmatrix}$$

Es decir $a1 = -0.7541; a2 = 0.0487; b1 = 0.2133; b2 = 0.0813.$

La gráfica de convergencia de parámetros se muestra en la figura 5.2.

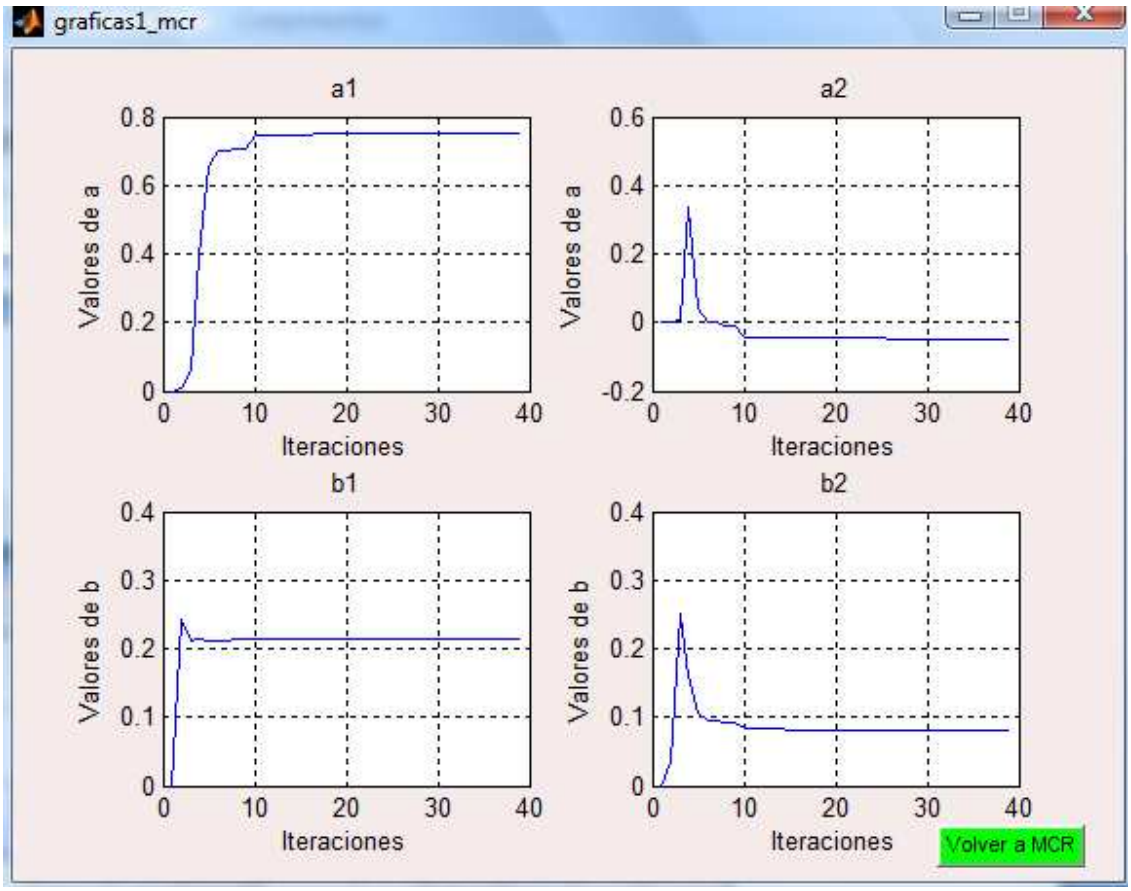


Figura 5.2 Convergencia de Parámetros

Se aprecia que los parámetros a y b convergen.

Para comparar la identificación realizada con el modelo real del sistema, se realiza la gráfica de dichas dos salidas, como se puede observar en la figura 5.3.

La salida identificada representada con el color azul es muy parecida a la salida real, por lo que queda confirmado que la identificación paramétrica es correcta.

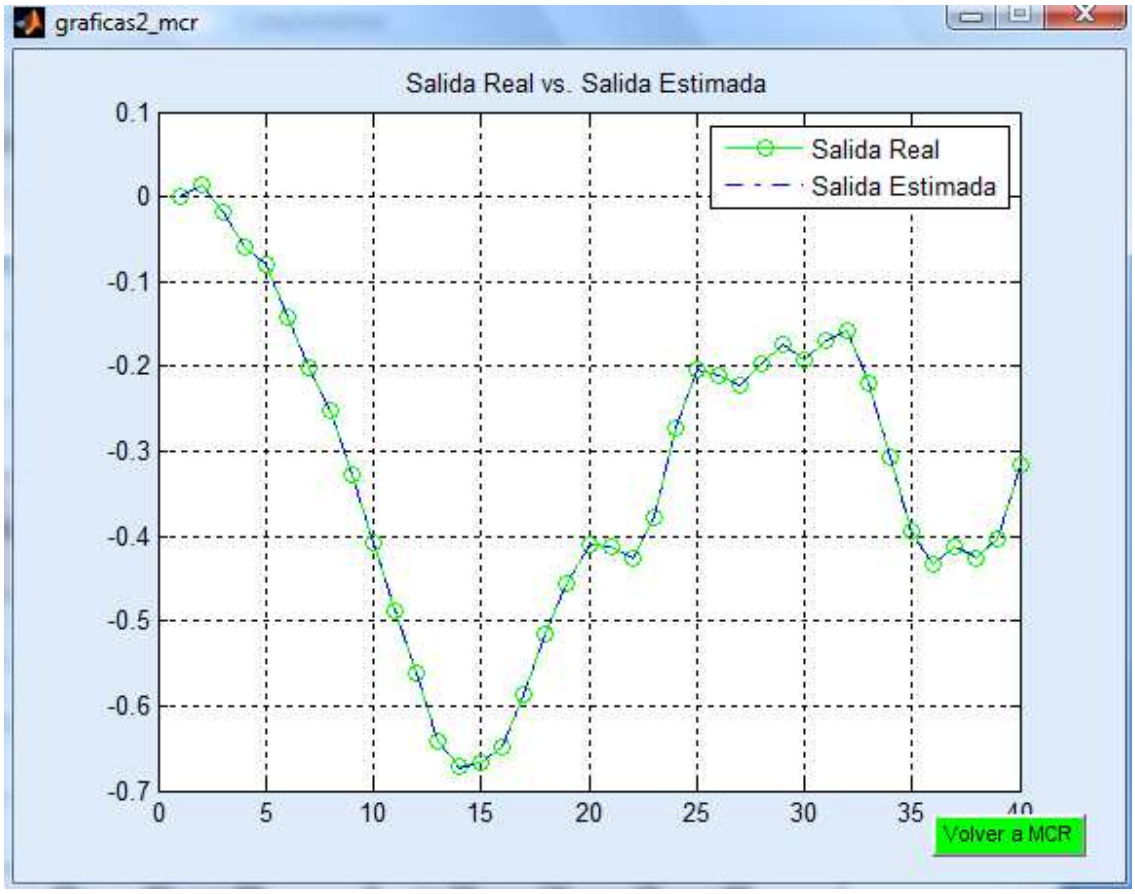


Figura 5.3 Gráfica de la Salida Real vs. Salida Estimada

A continuación se procederá a realizar otro ejercicio de Identificación, a partir de la siguiente función de transferencia.

$$\mathrm{Gp} = \frac{4}{s^2 + s}\ ; \mathrm{H}(s) = 1$$

Se generan datos a partir de Gp, en esta ocasión se genera 30 datos, como se observa en la figura 5.4.

Figura 5.4 Ventana de Generación de Datos para la Función de Transferencia Gp

Los datos generados se observa en la tabla 5.2

X	Y
1.2589	0
1.6232	1.8525
-1.4921	6.2531
1.6535	7.3923
0.5294	8.6676
-1.6098	11.6637
-0.8860	10.9569
0.1875	7.6917
1.8300	5.8301

1.8596	8.0365
-1.3695	13.5190
1.8824	15.4864
1.8287	17.5328
-0.0585	22.9666
1.2011	26.8128
-1.4325	29.9339
-0.3130	30.2444
1.6629	28.3847
1.1688	29.8173
1.8380	33.8226
0.6230	39.2367
-1.8572	44.0887
1.3965	43.8002
1.7360	43.7869
0.7149	47.8135
1.0310	52.1827
0.9725	56.0639
-0.4311	60.0136
0.6219	61.8614
-1.3153	63.0019

Tabla 5.2 Datos Generados para Gp

Así mismo, se necesita el cálculo de los parámetros a_1, a_2, b_1, b_2.
Con el método de mínimos cuadrados ordinario se obtiene:

$$\text{THETA_MCO} = \begin{bmatrix} 1.3679 \\ -0.3679 \\ 1.4715 \\ 1.0570 \end{bmatrix}$$

Es decir $a1 = -1.3679;\ a2 = 0.3679; b1 = 1.4715; b2 = 1.0570$

Con el método de mínimos cuadrados recursivo, después de 28 iteraciones, se obtiene:

$$\text{THETA_MCR} = \begin{bmatrix} 1.3679 \\ -0.3679 \\ 1.4715 \\ 1.0570 \end{bmatrix}$$

Es decir $a1 = -1.3679;\ a2 = 0.3679; b1 = 1.4715; b2 = 1.0570$

Cuya gráfica de convergencia de parámetros se observa en la figura 5.5.

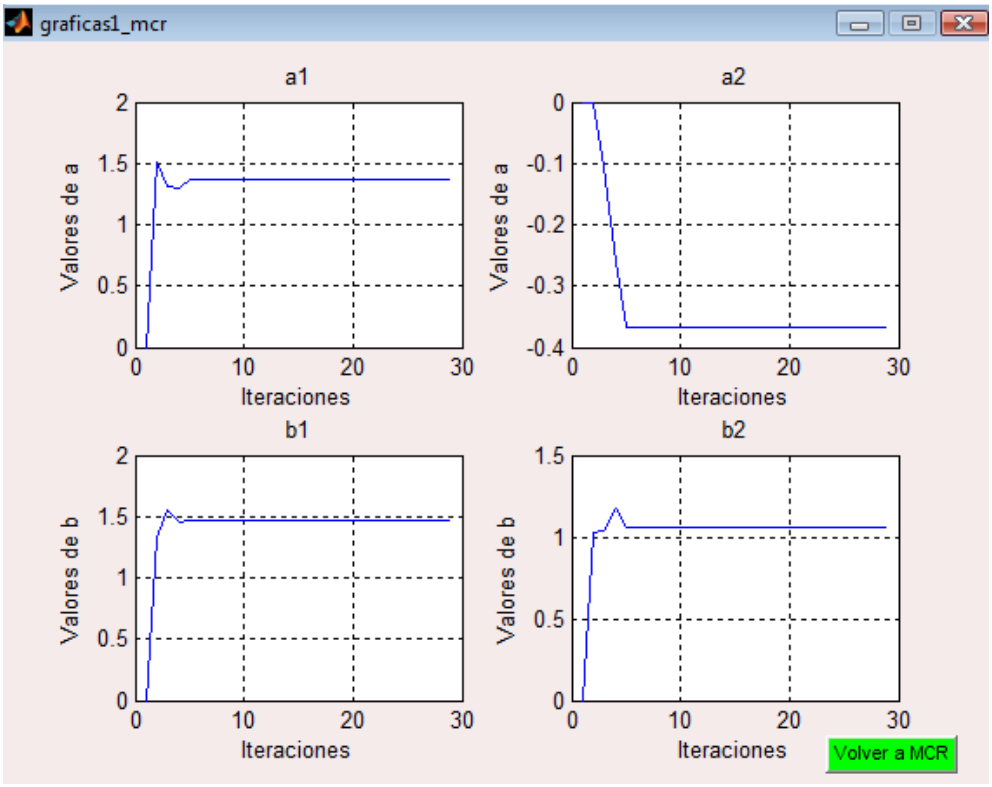


Figura 5.5. Gráfica de Convergencia de Parámetros

De igual forma que el ejercicio anterior para comparar la identificación realizada con el modelo real del sistema, se realiza la gráfica de dichas dos salidas, como se puede observar en la figura 5.6.

Figura 5.6 Gráfica de la Salida Real vs. Salida Estimada

En la figura 5.6 se observa como la salida creada a partir de los parámetros identificados y representada por el color azul, le sigue a la salida real representada por el color verde.

Como último ejercicio de identificación se realizará un ejercicio con la presencia de un parámetro c, esta vez se realizará el siguiente ejercicio utilizando comandos de Matlab.

A partir de la siguiente función de transferencia:

$$G = \frac{1}{25s^2 + 15s + 1}$$

Discretizada con un periodo de muestreo igual a 1 seg.

$$G(z) = \frac{0.01648z + 0.0135}{z^2 - 1.519z + 0.5488}$$

Esta vez, se debe ingresar un parámetro c para identificarlo. En este ejercicio se utilizará c=0.6.

Entonces los parámetros reales de G son:

$$a1 = -1.519; a2 = 0.5488; b1 = 0.01648; b2 = 0.0135; c1 = 0.6$$

A diferencia de la Identificación sin ruido, esta vez para generar los datos de salida se debe sumar un y1 y y2 representados:

$$yt = \frac{b1z^{-1} + b2z^{-2}}{1 + a1z^{-1} + a2z^{-2}} u(t) + \frac{1 + c1z^{-1}}{1 + a1z^{-1} + a2z^{-2}} e(t)$$

Donde:

$$y1 = \frac{b1z^{-1} + b2z^{-2}}{1 + a1z^{-1} + a2z^{-2}}\ u(t)$$

$$y2 = \frac{1 + c1z^{-1}}{1 + a1z^{-1} + a2z^{-2}}\ e(t)$$

$u(t) = Es\ representado\ por\ una\ entrada\ paso$

$e(t) = Es\ ruido\ blanco, generado\ a\ través\ del\ comando\ randn$

Utilizando el comando iddata y armax, aprendidos en el Capítulo II, numeral 2.4.1. se tiene:

dato=iddata(yt,u,1);

th=armax([yt x],[2 2 1 1]) ≡ th=armax([dato],[2 2 1 1])

Se obtiene en el Workspace de Matlab

A(q) = 1 - 1.486 q^-1 + 0.5146 q^-2

B(q) = 0.024 q^-1 + 0.005234 q^-2

C(q) = 1 + 0.5378 q^-1

Entonces los parámetros identificados de la planta son:

$$a1 = -1.486; a2 = 0.5146;\ b1 = 0.024; b2 = 0.005334; c1 = 0.5378$$

Se puede concluir que los parámetros identificados son parecidos a los parámetros reales.

Con la rutina hecha en archivo.m se puede observar la gráfica 5.7 en la cual se aprecia como el Error de Predicción representado por el color celeste, le sigue al ruido representado por el color rojo.

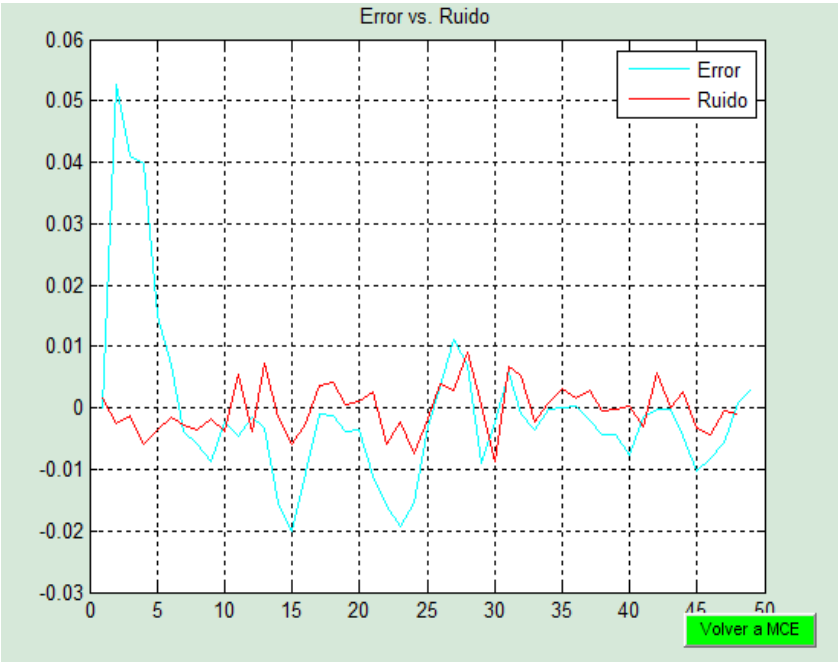


Figura 5.7. Gráfica de Error de Predicción vs. Ruido Generado

Resultados de Diseño de Controlador Adaptativo Self-tuning

Se realiza el análisis de la función de transferencia:

$$G(s) = \frac{8}{s^2 + s + 9.25}$$

Discretizándola con T=0.2 seg.

$$G(z) = \frac{0.1453z + 0.1359}{z^2 - 1.494z + 0.8187}$$

En la figura 5.8 se observa la respuesta de G(z) en lazo abierto frente a una entrada paso.

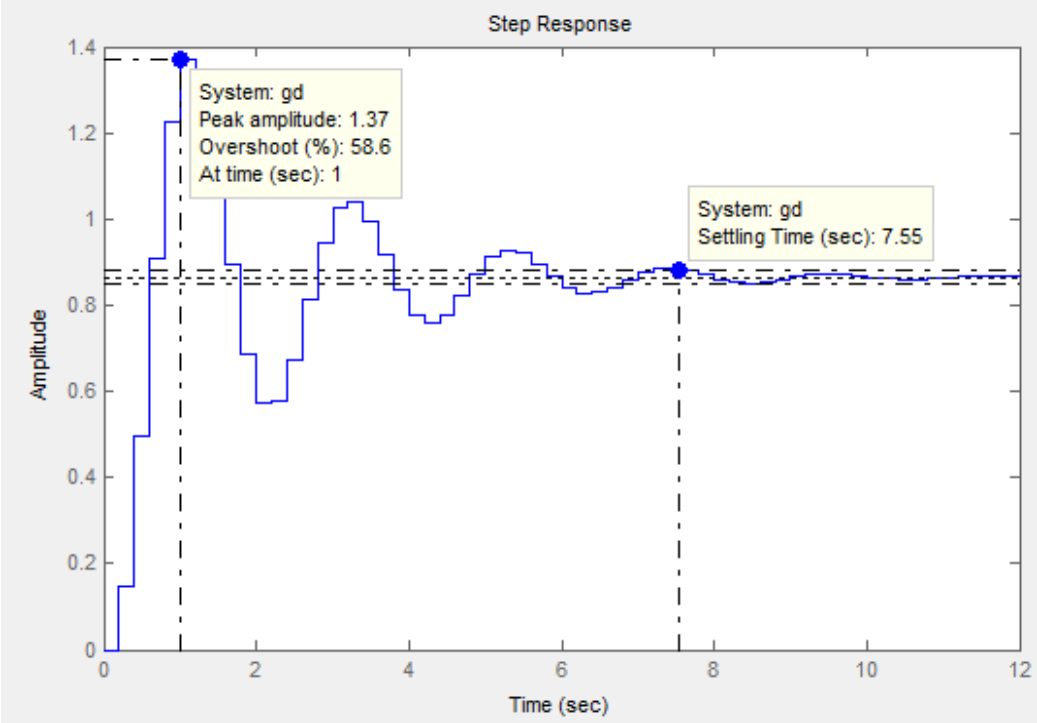


Figura 5.8. Respuesta de G(z) en Lazo Abierto frente a una Entrada Paso

La figura 5.8 muestra un tiempo de establecimiento igual a 7.55 seg y un sobrepico igual al 58.6%.

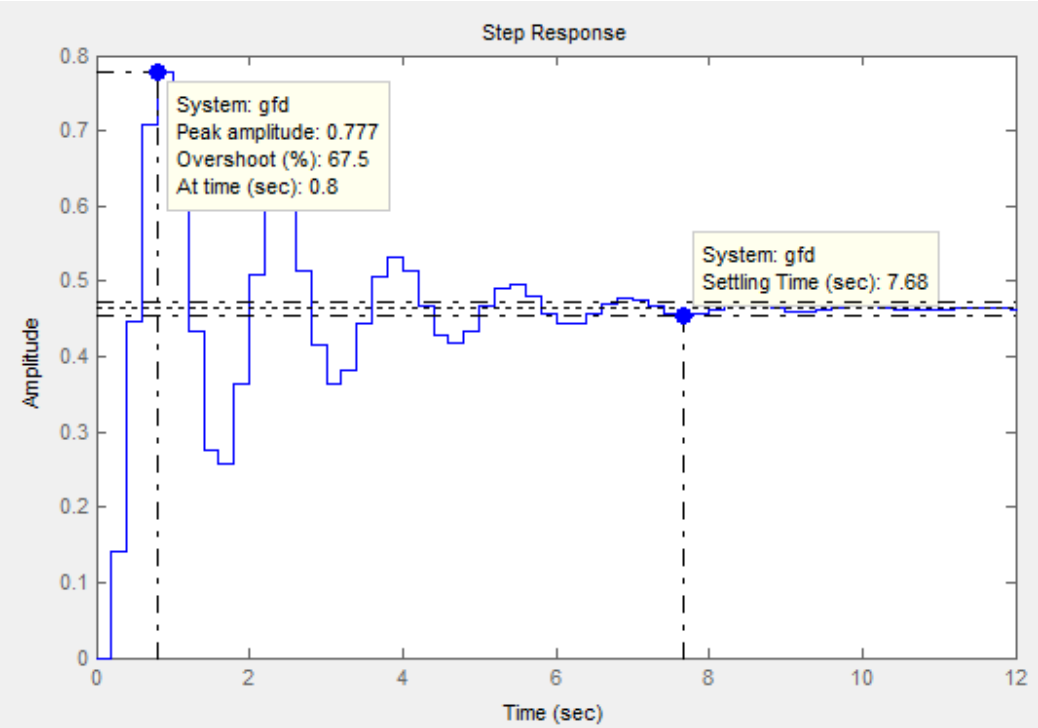


Figura 5.9. Respuesta de G(z) en Lazo Cerrado frente a una Entrada Paso

La figura 5.9 muestra la respuesta en lazo cerrado de la planta G(z) en donde se aprecia una planta con un tiempo de establecimiento de 7.68 seg y un sobrepico de 67.5%, por lo que se realiza el diseño de un Controlador Adaptativo Self-tuning con el método de Asignación de Polos para mejorar esta respuesta.

Al ser el método a utilizar el método de Asignación de Polos, se diseña un Controlador con 2 Polos Reales =$(z-0.8)(z-0.9)$.

$$T(z^{-1}) = 1 - 1.7z^{-1} + 0.72z^{-2}$$

Al ser un sistema de segundo orden se debe calcular: $f1, g0, g1, g2$.

Para obtener los valores de los Polinomios F y G, se debe resolver el sistema matricial:

$$\begin{bmatrix} 1 & 0.1453 & 0 & 0 \\ -2.4935 & 0.1359 & 0.1453 & 0 \\ 2.3121 & 0 & 0.1359 & 0.1453 \\ -0.8186 & 0 & 0 & 0.1359 \end{bmatrix} * \begin{bmatrix} f1 \\ g0 \\ g1 \\ g2 \end{bmatrix} = \begin{bmatrix} -1.7 + 2.4935 \\ 0.72 - 2.3121 \\ 0 + 0.81862 \\ 0 \end{bmatrix}$$

$$f1 = 0.4694; g0 = 2.2303; g1 = -4.9870; g2 = 2.8279$$

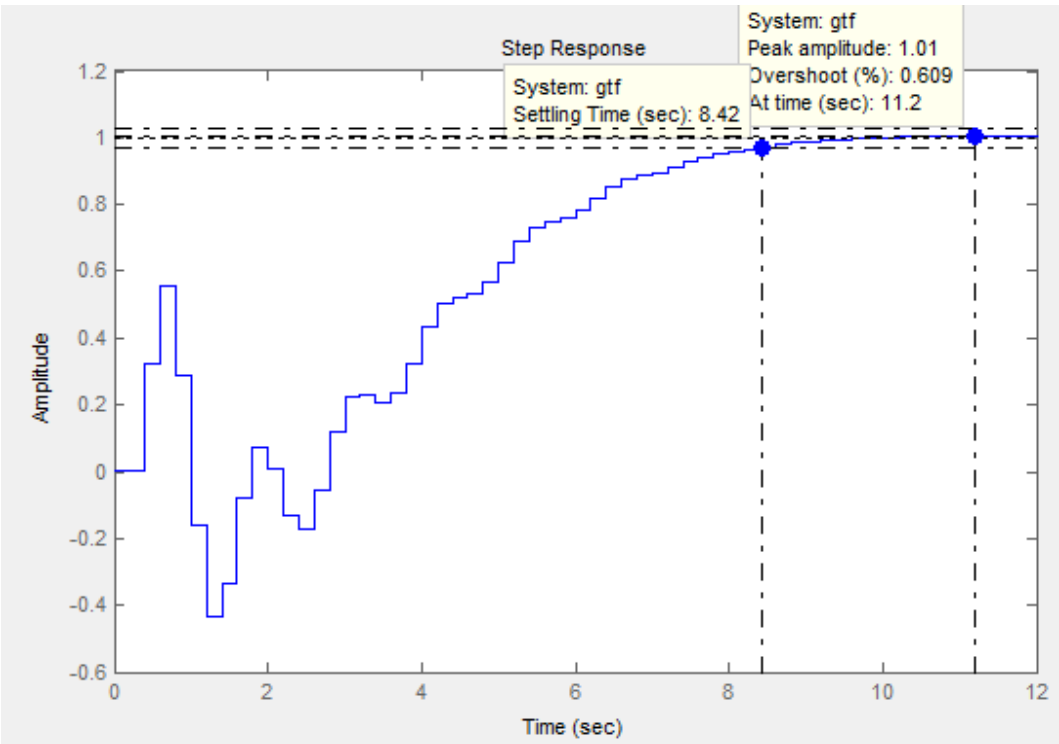


Figura 5.10. Respuesta de la Planta con Controlador y Polos Reales

$$(\mathbf{z} - \mathbf{0.8})(\mathbf{z} - \mathbf{0.9})$$

Como se puede apreciar en la figura 5.10. la respuesta del sistema con Controlador Adaptativo es más rápida que el sistema original, ya que el sobrepico disminuyó a 0.6% aunque se sacrificó el tiempo de establecimiento al incrementarlo a 8.42 seg.

A continuación se procede a realizar el diseño del Controlador Adaptativo con tres Polos Reales=$(z - 0.7)(z - 0.8)(z - 0.9)$

$$T(z^{-1}) = 1 - 2.4z^{-1} + 1.91z^{-2} - 0.504z^{-3}$$

Para obtener los valores de los Polinomios F y G, se debe resolver el sistema matricial:

$$\begin{bmatrix} 1 & 0.1453 & 0 & 0 \\ -2.4935 & 0.1359 & 0.1453 & 0 \\ 2.3121 & 0 & 0.1359 & 0.1453 \\ -0.8186 & 0 & 0 & 0.1359 \end{bmatrix} * \begin{bmatrix} f1 \\ g0 \\ g1 \\ g2 \end{bmatrix} = \begin{bmatrix} -2.4 + 2.4935 \\ 1.91 - 2.3121 \\ -0.504 + 0.8186 \\ 0 \end{bmatrix}$$

$$f1 = 0.1208; g0 = -0.1878; g1 = -0.5186; g2 = 0.7278$$

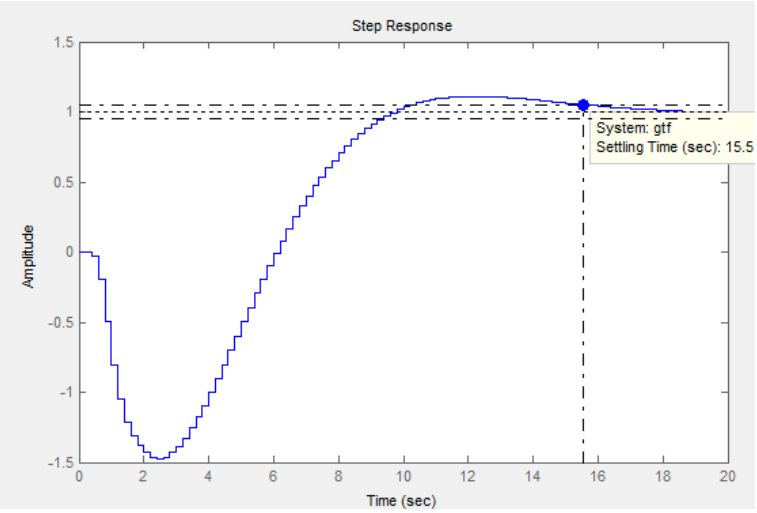


Figura 5.11. Respuesta de la Planta con Controlador y Polos Reales

$$(\mathbf{z} - \mathbf{0.7})(\mathbf{z} - \mathbf{0.8})(\mathbf{z} - \mathbf{0.9})$$

La figura 5.11 muestra un tiempo de establecimiento de 15.5 seg, es decir es una respuesta más lenta que la respuesta obtenida con el controlador diseñado con 2 polos reales.

5.2. RESULTADOS DE TIEMPO REAL

Para las pruebas en tiempo real se implementaron dos circuitos representados en el Capítulo IV por las figuras 4.6 y 4.11.

La figura 5.12 muestra la respuesta del circuito RC en lazo abierto. Es un sistema lento con un tiempo de establecimiento de 48 seg.

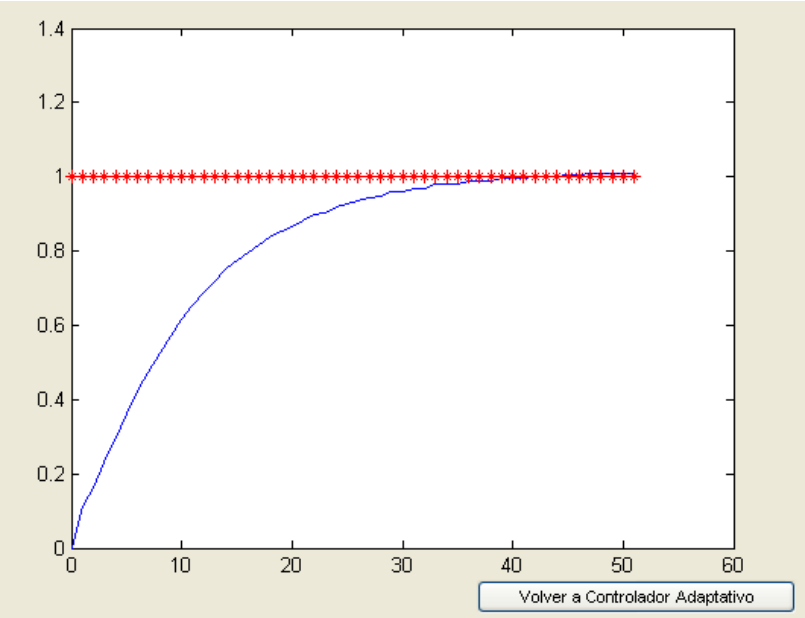


Figura 5.12. Respuesta de la Planta RC en Lazo Abierto

La gráfica de la figura 5.13 muestra la respuesta de la planta RC en lazo cerrado, graficada en Simulink. El tiempo de simulación de la figura 5.13 es de 10000 que son aproximadamente 92 segundos, es decir la planta RC en lazo cerrado se establece en aproximadamente 28 seg al valor de 0.532 V.

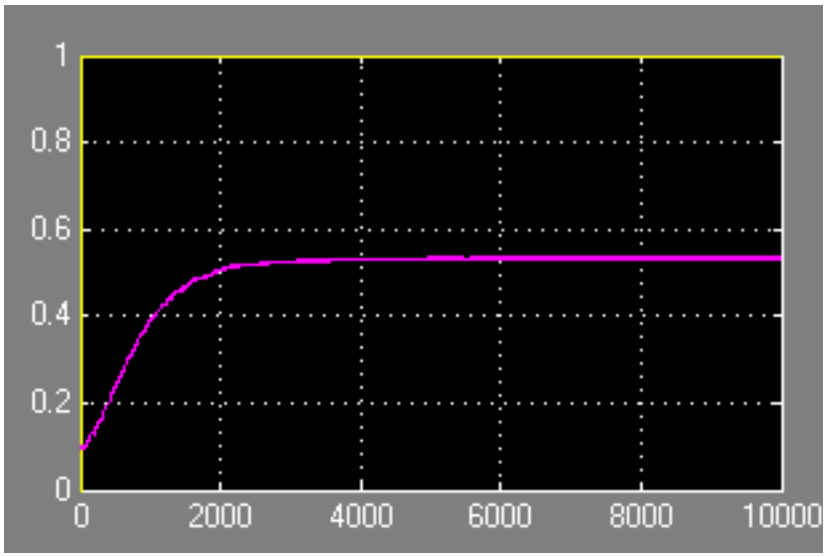


Figura 5.13. Respuesta de la Planta RC en Lazo Cerrado

En Tiempo Real se procedió a diseñar un Controlador Adaptativo Self-tuning con el método de Asignación de Polos.

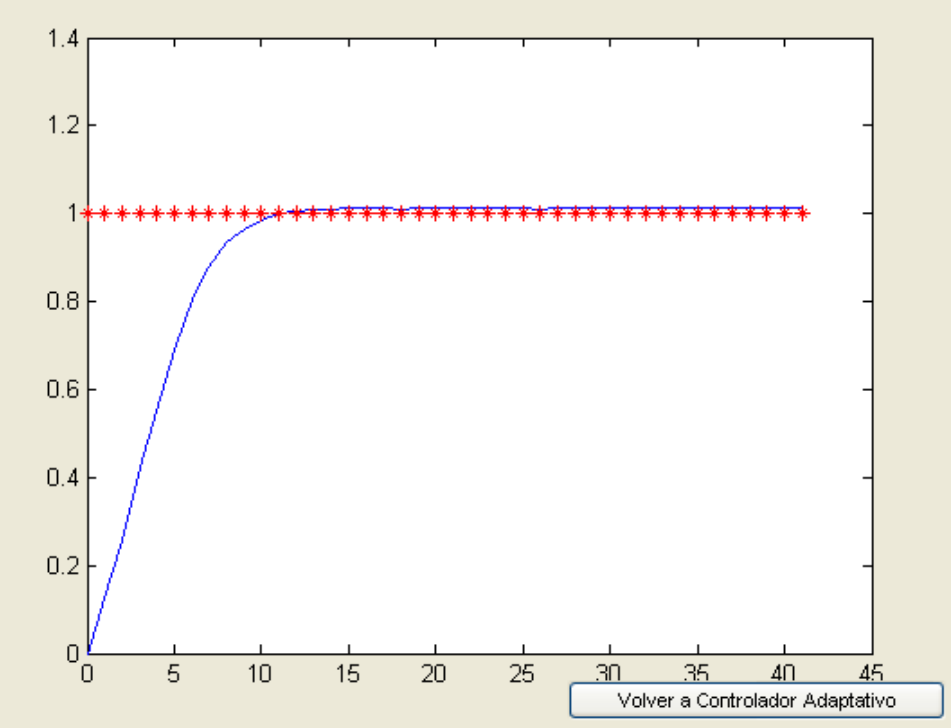


Figura 5.14. Respuesta de la Planta RC con Controlador Adaptativo con Polos Reales (z-0.1)(z-0.2)

La respuesta de la figura 5.14 muestra la salida de la planta RC con un Controlador Adaptativo diseñado con ubicación de dos polos reales (z-0.1)(z-0.2), siendo esta respuesta más rápida que la original ya que presenta un tiempo de establecimiento de 16 seg.

A continuación se diseña un Controlador Adaptativo con dos Polos Reales ubicados en (z-0.5)(z-0.6), y su respuesta se aprecia en la figura 5.15, una respuesta rápida pero con error de posición debido a que no se cuenta con un integrador físico, ya que si se colocara dicho integrador, los parámetros de la planta cambiarían.

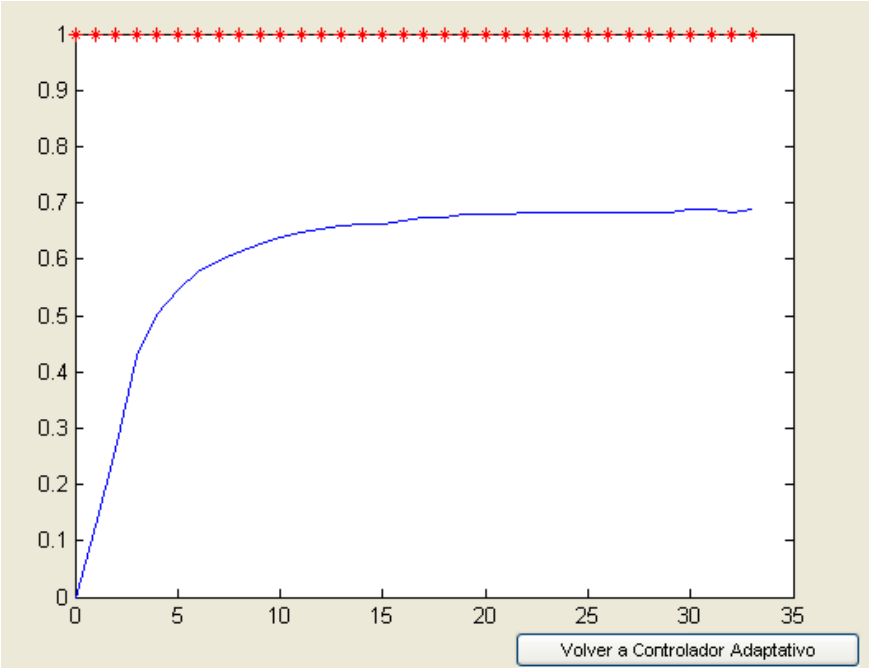


Figura 5.15. Respuesta de la Planta RC con Controlador Adaptativo con Polos Reales (z-0.5)(z-0.6)

Si colocamos más polos y dichos polos más pequeños, claramente se ve que el error de posición en el sistema disminuye, como se observa en las figuras 5.16, y 5.17.

A continuación se realizará el diseño de un controlador con polos imaginarios únicamente con la finalidad de probar que mediante la ubicación de polos se obtiene una respuesta deseada del sistema; ya que para este caso específicamente no es necesario colocar un controlador que me entregue una respuesta oscilatoria del sistema.

En la figura 5.18 se puede observar la respuesta del sistema RC con Controlador Adaptativo con Polos Complejos Conjugados (z-(-0.4+0.5j))(z-(0.4-0.5j)).

La respuesta de la figura 5.18 es una respuesta un poco oscilatoria, debido a la ubicación del los polos para el diseño del Controlador.

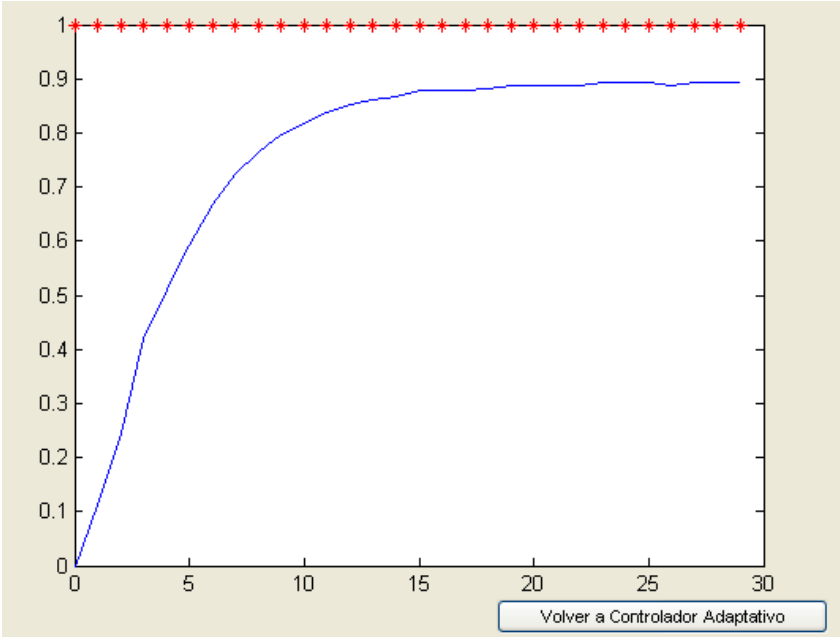


Figura 5.16. Respuesta de la Planta RC con Controlador Adaptativo con Polos Reales (z-0.3)(z-0.4)

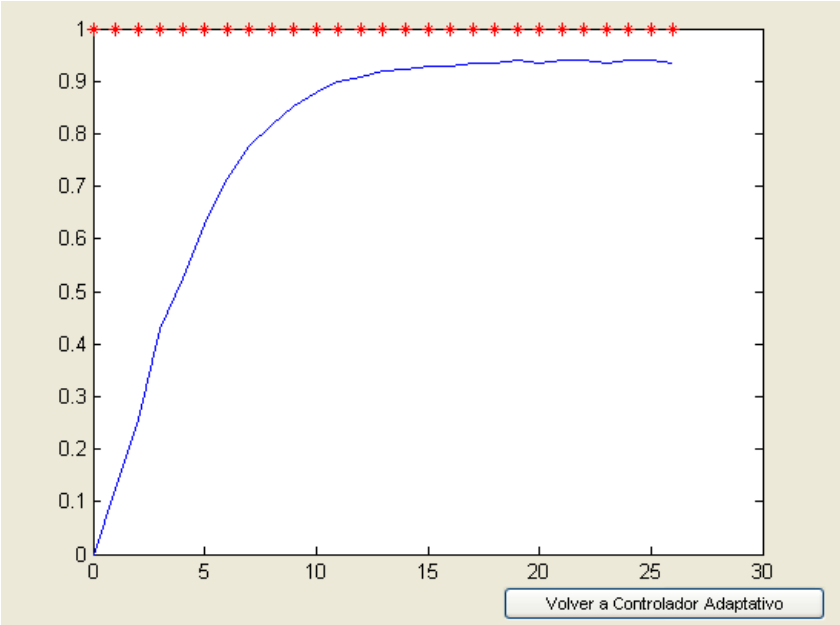


Figura 5.17. Respuesta de la Planta RC con Controlador Adaptativo con Polos Reales (z-0.1)(z-0.2)(z-0.3)

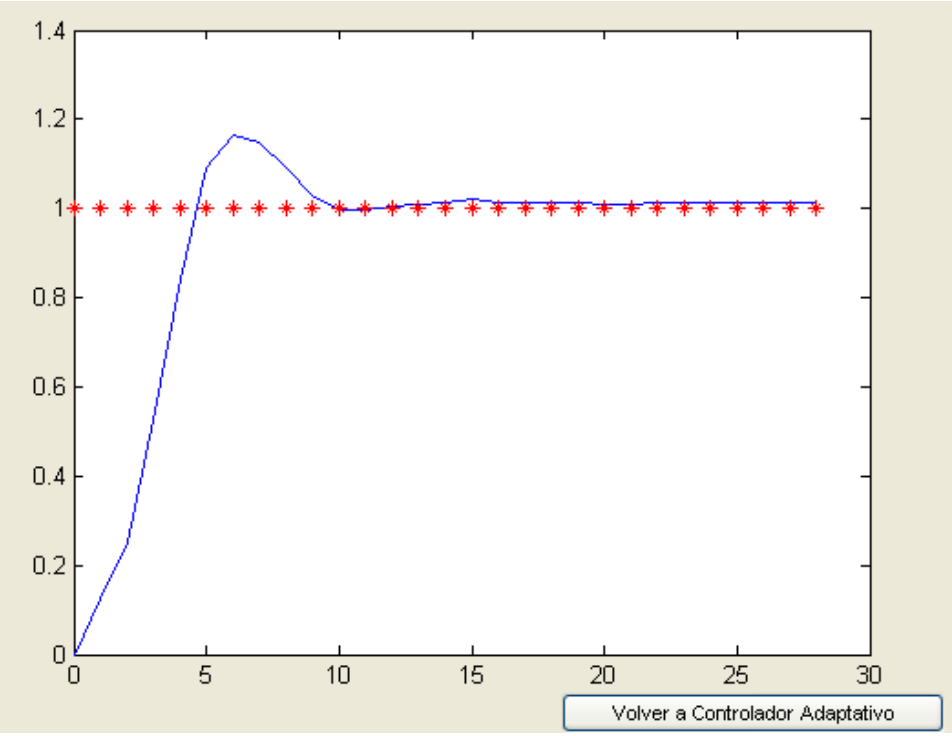


Figura 5.18. Respuesta de la Planta RC con Controlador Adaptativo con Polos Complejo Conjugados (z-(-0.4+0.5j))(z-(0.4-0.5j))

Como última prueba para el sistema RC se procedió a realizar un cambio en la planta una vez que ésta se haya estabilizado. Dicho cambio, se lo hizo al variar una de las resistencias de 5 kΩ del sistema por una resistencia de 10 kΩ.

El cambio se lo puede apreciar en la figura 5.19 aproximadamente a los 21 seg como una variación en la salida, esto se debe a que el algoritmo debe identificar los nuevos parámetros de la planta, los cuales convergen después de pocas iteraciones.

Otra prueba en tiempo real, se la realizó con el circuito de amplificadores operacionales, su respuesta en lazo abierto esta vez se la observó en Simulink en la figura 5.20.

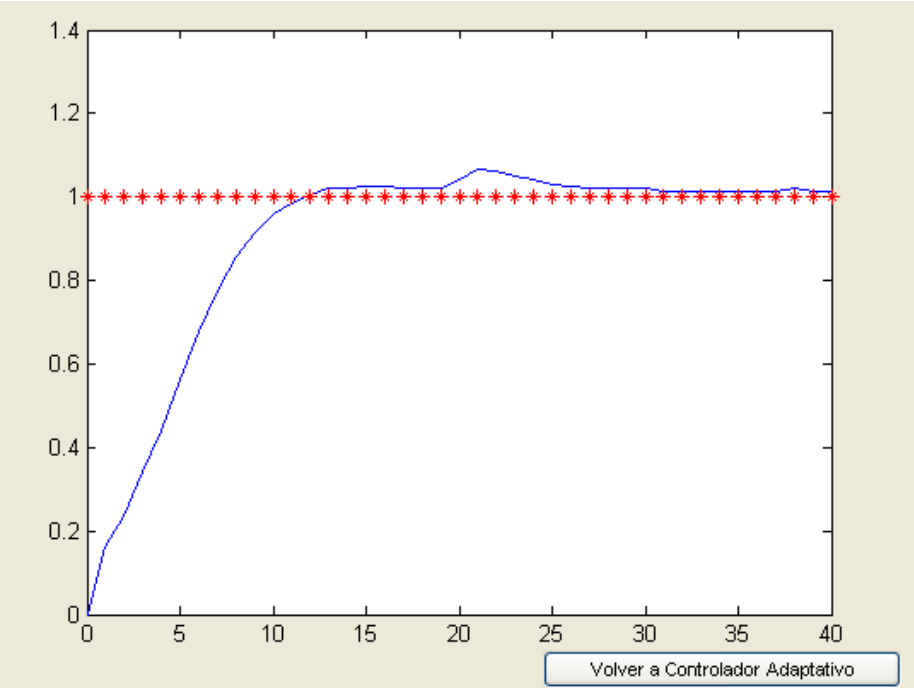


Figura 5.19. Respuesta de la Planta RC con Controlador Adaptativo con Polos Reales (z-0.1)(z-0.2) y cambio en la Planta.

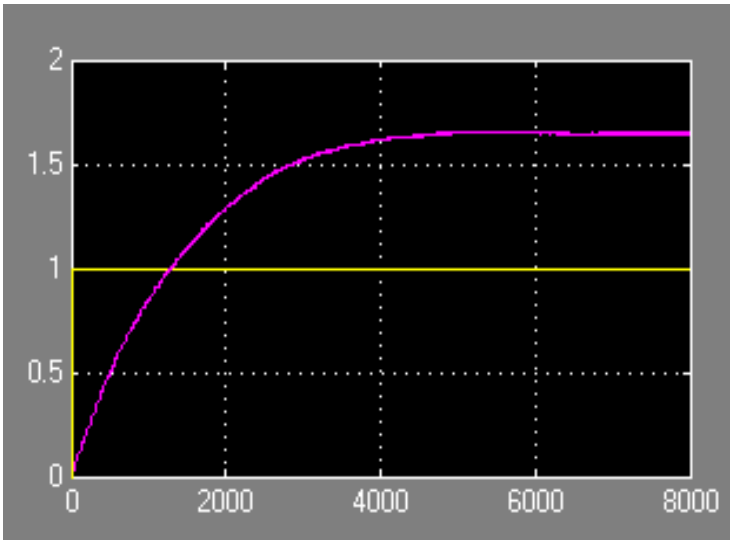


Figura 5.20. Respuesta de la Planta de Amplificadores Operacionales en Lazo Abierto

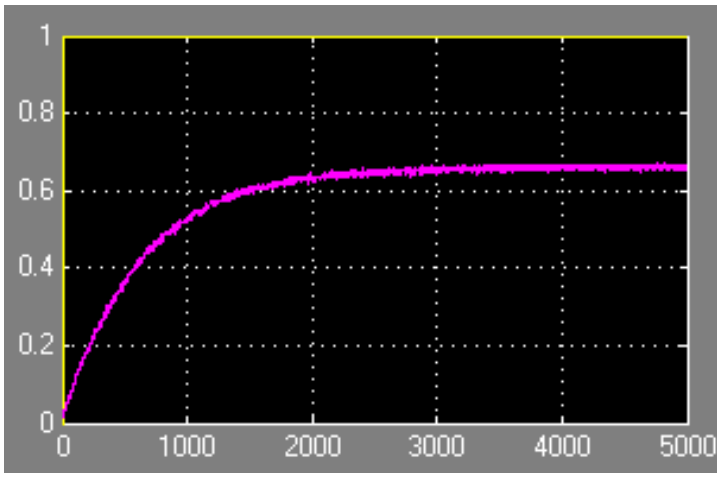


Figura 5.21. Respuesta de la Planta de Amplificadores Operacionales en Lazo Cerrado

La respuesta de la figura 5.21 muestra la salida del circuito de amplificadores operacionales en lazo cerrado, muestra una planta de respuesta lenta que tiene un tiempo de establecimiento de aproximadamente 33 seg. y con error de posición ya que se estabiliza a 0.6V.El tiempo de simulación para la figura 5.21 es de 5000 que aproximadamente es igual a 46 seg.

Se diseñó un Controlador Adaptativo variando la ubicación de los polos.

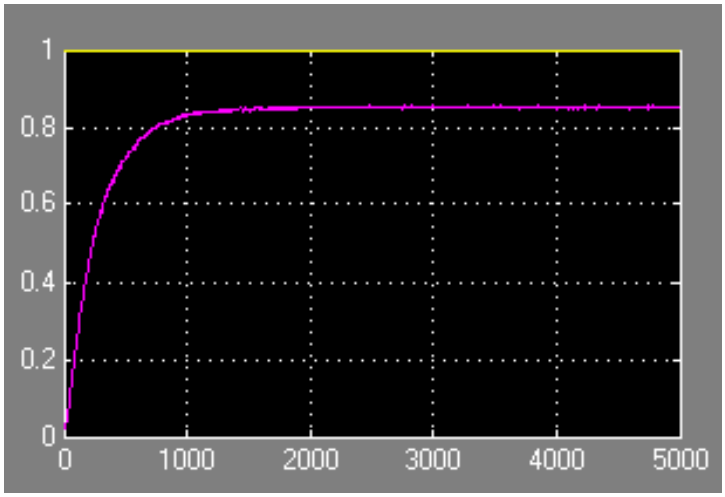


Figura 5.22. Respuesta de la Planta de Amplificadores Operacionales con Controlador Adaptativo con Polos (z-0.1)(z-0.2)

La respuesta de la figura 5.22 muestra una respuesta más rápida ya que el sistema presenta un tiempo de establecimiento de aproximadamente igual a 11 seg.

En la figura 5.23 se observa la Ley de Control aplicada al sistema original para obtener la respuesta de la figura 5.22.

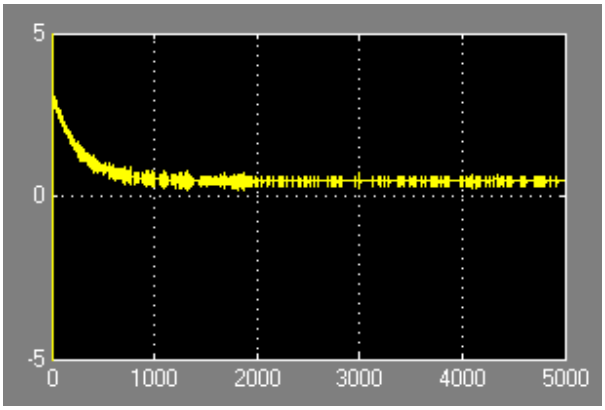


Figura 5.23. Ley de Control

A continuación, en la figura 5.24 se muestra la respuesta del sistema con un Controlador Adaptativo con tres polos reales (z-0.1)(z-0.2)(z-0.3), y su ley de control en la figura 5.25.

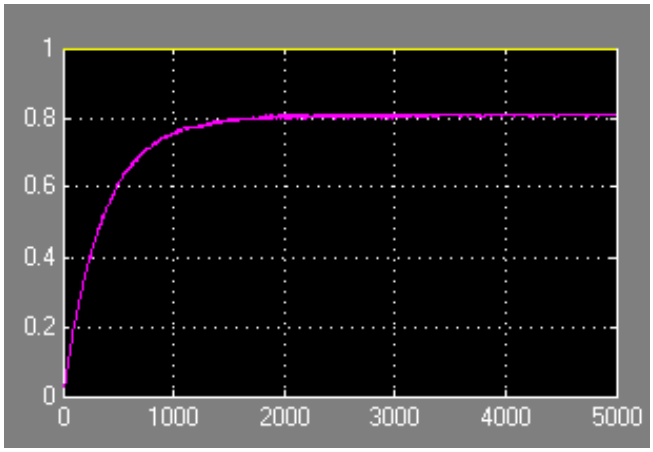


Figura 5.24. Respuesta de la Planta de Amplificadores Operacionales con Controlador Adaptativo con Polos (z-0.1)(z-0.2)(z-0.3)

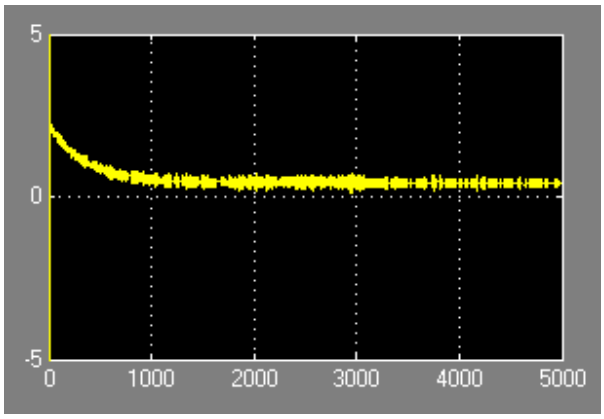


Figura 5.25. Ley de Control

CAPÍTULO VI
CONCLUSIONES

6.1. CONCLUSIONES

- Para el proceso de identificación en el presente proyecto se ha utilizado el método de mínimos cuadrados recursivo debido a que la estimación de parámetros se la realiza de manera recursiva, es decir, existe una actualización de parámetros después de cada periodo de muestreo, para actualizar la ley de control.

- La herramienta computacional Matlab presenta varias alternativas para la ejecución de este Proyecto de Titulación, las cuales se las ha explicado durante el desarrollo del mismo, el estudiante puede escoger la opción que mejor le parezca, e incluso puede combinar los procesos de cálculo, es decir, para la identificación de parámetros puede utilizar el propio toolbox de Matlab y para realizar el cálculo del Controlador Adaptativo utilizará la programación adecuada realizada en archivo extensión .m.

- Se debe tener en cuenta que tanto para la simulación como para tiempo real, en este proyecto específico no se ha utilizado Simulink, debido a que no contiene una librería para Control Adaptativo, es decir, el diseño del Controlador Adaptativo Self-Tuning por el método de Asignación de Polos, se lo realizó mediante programación en archivos. m, además al no poder combinar los procesos realizados en Workspace como en Simulink, todo el proceso desde Identificación de Parámetros hasta Diseño del Controlador se realizó en programación.

- Se ha implementado una plataforma para realizar control en Tiempo Real utilizando adquisición y salida de datos, identificación y Control Self-tunimg, que entrega resultados satisfactorios.

- El método de Identificación de mínimos cuadrados es suficiente para aplicaciones de Control Digital, ya que incluso entrega resultados satisfactorios aún con la presencia de ruido blanco, aproximación que se puede realizar en la mayoría de procesos industriales.

- Si es que el ruido es correlacionado, es decir, el modelo de las perturbaciones contiene un filtro, se puede utilizar el método de mínimos cuadrados extendido, pues como se pudo apreciar en los resultados, el error de predicción le sigue al ruido, con lo que además de identificar la planta se identifica la dinámica de la perturbación.

- En cuanto al algoritmo de control, éste entrega resultados satisfactorios para el caso de regulación, sin embargo se puede extender al caso de seguimiento añadiendo un integrador.

- La bondad del algoritmo se puede apreciar cuando se realiza una modificación de la planta, y tanto la identificación como el control permiten detectar aquella variación y realizar la regulación para regresar a la salida deseada.

6.2. RECOMENDACIONES

- Se recomienda utilizar para la adquisición de datos, con la tarjeta NI-DAQ 6008, las entradas diferenciales y no las entradas simples, ya que estas últimas presentan un diferencia de voltaje igual a 1.4 V, es decir, si se desea medir a través de la DAQ 5V proporcionados por una fuente externa,

no se leerán 5V sino por el contrario se obtendrán 3.6V debido a esta diferencia de voltaje existente en las entradas simples de la NI-DAQ.

- En ocasiones después de trabajar duramente con el programa computacional Matlab, éste suele presentar errores como que existen bloques de Simulink que están siendo utilizados para otras aplicaciones, cuando éste no es el caso; otros, en los que durante la ejecución de programas con extensión .m aparecen cuadros de error que llevan otros errores incomprensibles, en estos casos, lo que se recomienda es cerrar Matlab, esperar unos momentos y abrirlo nuevamente. Durante la realización de diversas simulaciones en ésta Tesis, se presentaron problemas como los mencionados, entonces se procedió a cerrar Matlab, e incluso a apagar y encender nuevamente el computador para seguir con los trabajos requeridos, y sin realizar cambio alguno, los programas funcionaron correctamente.

REFERENCIAS BIBLIOGRÁFICAS

Libros:

- IEE CONTROL ENGINEERING SERIES 21; BENNETT,S; LINKENS D.A. Computer Control of Industrial Processes. Peter Peregrinus Ltd. New York.1982.

- FRANKLIN, Gene F.; POWELL, J. David. Digital Control of Dynamic Systems. Addison Wesley Publishing Company. Primera Edición. 1980.

- ASTROM, Johan Karl.; WITTENMARK, Bjorn. Adapatative Control. Addison-Wesley Publishing Company. Primera Edición. 1989.

- IEE CONTROL ENGINEERING SERIES 15; HARRIS, C.J.; BILLINGS, S.A.; Self-Tuning and Adaptative Control. Peter Peregrinus Ltd. Primera Edición.1981.

- LASAC`95. Second Latin American Seminar on Advanced Control, Fourth Seminar on System Identification, Parameter Estimation and Adaptative Control. Chile. Septiembre 1995.

Direcciones Electrónicas:

- http://www.monografias.com/trabajos22/aplicaciones-matlab/aplicaciones-matlab.shtml
- http://www.dea.icai.upco.es/ramon/Control_Avanzado/Adaptativo/Estima_Adapta.pdf
- http://roboticslab.uc3m.es/publications/SGarridoPhD.pdf
- http://www.edutecne.utn.edu.ar/tutoriales/identificacion-sistemas-segundo-orden.pdf

- http://www.epsem.upc.edu/~esaii/assign/ident/Tema%201.pdf
- http://ingenieria-matematica.eafit.edu.co/recursos/glosario/r.html
- http://automata.cps.unizar.es/Historia/Webs/teoria_moderna_de_control.htm
- http://ingenieria-matematica.eafit.edu.co/recursos/historia/1940_1960.html
- http://www.dea.icai.upco.es/ramon/Control_Avanzado/Material2.html#adaptativo
- http://instruments.hao.ucar.edu/spinor/htdocs/NI-DAQ.pdf
- http://www.slideshare.net/ptah_enki/asignacin-de-polos-y-predictor-de-smith
- http://math.bu.edu/misc/DOCSERVER/raw/ident.pdf
- http://www.mathworks.com/access/helpdesk/help/toolbox/ident/exampleindex.html
- http://www.profesaulosuna.com/data/files/TELECOMUNICACIONES/AM/CIRCUITOS%20AM/ruido.pdf
- http://www.monografias.com/trabajos-pdf3/glosario-terminos-control/glosario-terminos-control.pdf
- http://www.uv.es/gcamps/materialea/ch06.pdf
- http://books.google.com.ec/books. MORENO, Luis. “Ingeniería de Control: modelado, análisis y control de sistemas dinámicos”

ANEXOS

A.1 Manual de la Tarjeta Ni- DAQ 6008

Ni-Daq 6008

- 8 analog inputs at 12 or 14 bits, up to 48 kS/s
- 2 analog outputs at 12 bits, software-timed
- 12 TTL/CMOS digital I/O lines
- One 32-bit, 5 MHz counter
- Digital triggering
- Bus-powered

Requirements and Compatibility

OS Information	Driver Information	Software Compatibility
Windows 2000/XP Mac OS X Windows 7 Windows Mobile Windows Vista x64 Windows Vista x86 Windows CE	NI-DAQmx Base NI-DAQmx	ANSI C/C++ LabVIEW LabVIEW SignalExpress Measurement Studio LabWindows/CVI

Product	USB-6008
Analog Inputs	8 single-ended/4 differential
Input Resolution	12
Max Sampling Rate (kS/s)	10
Analog Outputs	2
Output Resolution	12
Output Rate (Hz)	150
Digital I/O Lines	12
32-Bit Counter	1
Triggering	Digital

A.2 Manual de Usuario del Control Adaptativo

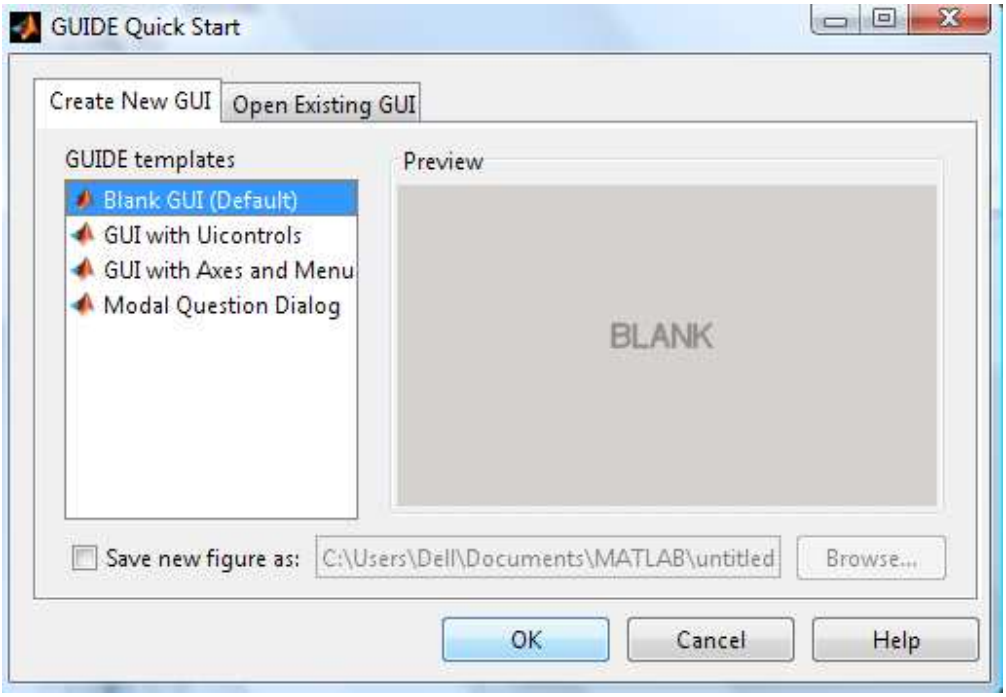
Para mayor comodidad al momento de realizar tanto la identificación de los parámetros de la planta, como para el cálculo del controlador, el usuario contará con una pantalla gráfica que lo ayudará a realizar las acciones deseadas.

A continuación se dará una breve explicación de cómo acceder a un GUI existente y como crear nuevos GUIs. Para acceder a dicha interfaz gráfica, el usuario debe abrir primero el programa computacional Matlab.

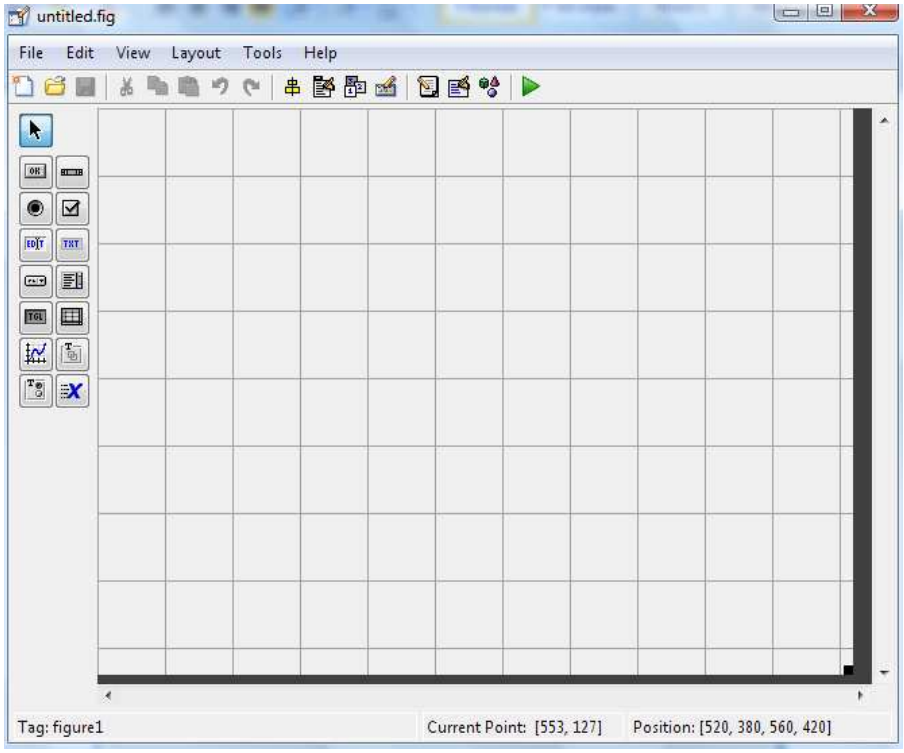
Luego deberá escribir en el workspacede Matlab el comando *guide*.

>> guide

Se desplegará en la pantalla la ventana del GUI (graphicaluser interface), para crear un nuevo GUI en blanco.



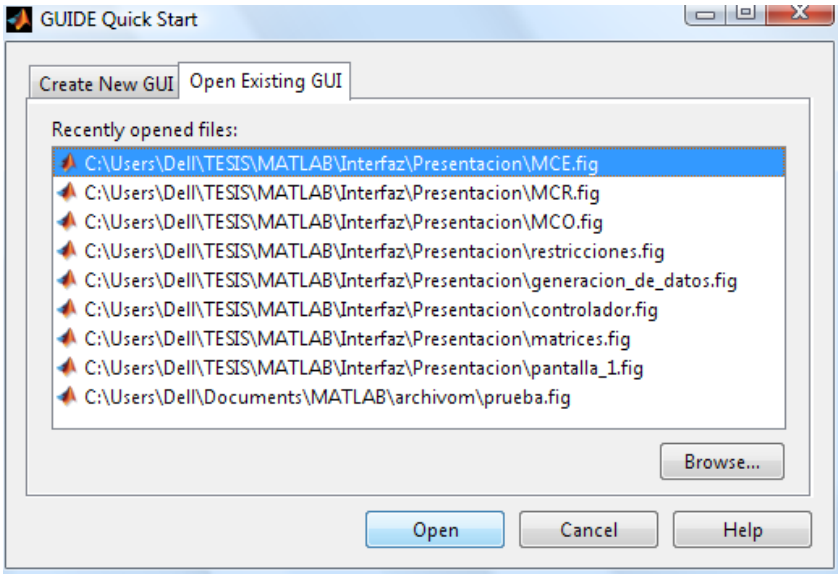
Al abrir el GUI en blanco se obtendrá la siguiente pantalla:



Donde, para realizar cualquier acción primero se lo debe guardar, y se obtendrán dos archivos, uno de ellos con extensión *.fig* que representa la pantalla en donde se van a realizar todos los cambios que el usuario va a manejar; y otro archivo con extensión *.m* donde se encuentra toda la programación correspondiente a cada pantalla.

Para poder ejecutar el programa se debe hacer click sobre el ícono de RunFigure .

Ahora bien, si se desea abrir un GUI ya existente, se lo debe seleccionar de la siguiente pantalla como se puede ver a continuación:



Una vez que el usuario conoce como abrir el GUIDE, se procederá a explicar los pasos a seguir para realizar el proceso de identificación y el diseño del controlador.

La primera pantalla a desplegarse será la que muestre dos opciones, las cuales son: Simulación o Control en Tiempo Real, así:

Al escoger la opción de Simulación el usuario tendrá la opción de realizar por él mismo Identificación y Diseño del Controlador Adaptativo Self-tuning, como se ve a continuación:

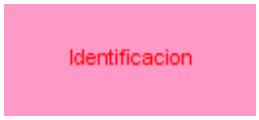
GENERACIÓN DE DATOS

Para generar datos, el usuario deberá ingresar tanto el numerador como el denominador de una función de transferencia, y se logrará discretizarla al presionar el botón de Discretizar, obteniendo tanto un numerador y denominador discretos. Además, ingresará cuantos valores desea generar, una vez ingresados todos estos datos, se debe presionar el botón de Generar Valores y se observará en el workspace de Matlab dos columnas de datos generados, tanto de entrada (x) como de salida (y).

IDENTIFICACIÓN DE PARÁMETROS

Al presionar el botón de Restricciones, inmediatamente se desplegará una ventana en la cual el usuario deberá ingresar las condiciones necesarias para realizar la identificación de parámetros.

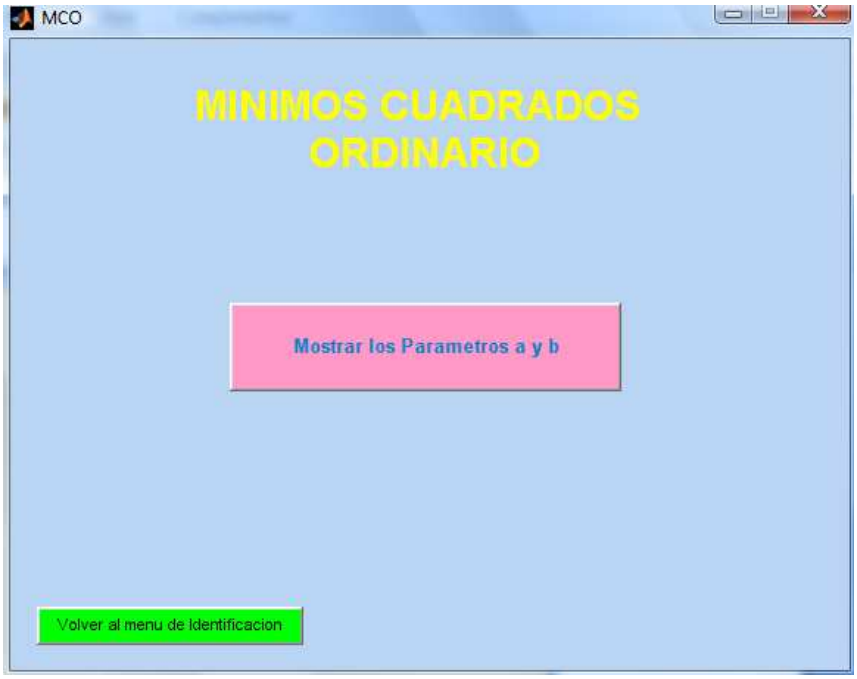
Como se puede observar, la ventana de restricciones es aquella en donde se ingresan los datos necesarios para el proceso de identificación, es decir los órdenes de *a,b,c*, y *k*, se debe tener en cuenta que el número de iteraciones posibles a realizar durante la recursión se ve restringida por la cantidad de valores a generar y por el orden de a, por lo que al presionar el botón de Calculo de Iteraciones posibles se obtendrá el número de repeticiones que el usuario podría realizar durante la recursión. Entonces, en el recuadro en blanco de Iteraciones, el usuario ingresará cuantas veces desea realizar la recursividad.

Cuando todas las restricciones hayan sido ingresadas, se procederá a realizar la Identificación de parámetros sólo presionando el botón de 

La Identificación de parámetros se la puede realizar mediante el método de los Mínimos Cuadrados, ya sean estos, Ordinarios, Recursivo o Extendido.
Para realizar una validación de resultados, se puede obtener los parámetros de *a,b,c,*y, *k,* mediante los tres métodos de identificación.

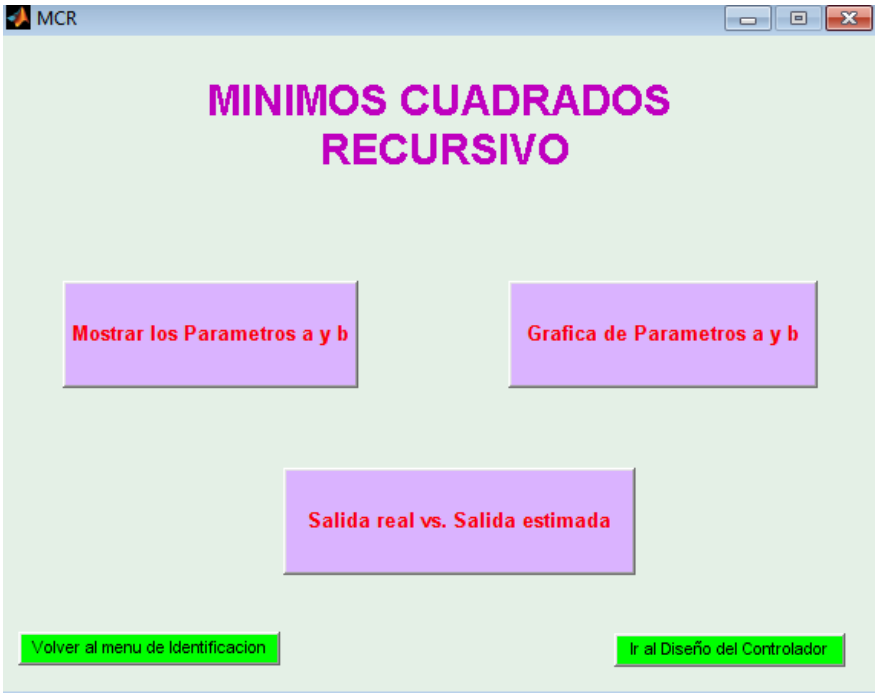
Al escoger la opción de Mínimos Cuadrados Ordinario se observará en el workspace de Matlab la matriz THETA_MCO, la misma que contiene los parámetros *a,b* identificados.

Si se escoge el método de identificación de Mínimos Cuadrados Recursivo, se podrá observar además de la matriz THETA_MCR formada por los parámetros a,b, la gráfica de la convergencia de los mismos y otra gráfica en la cual se puede observar la salida del sistema real vs. La salida estimada del sistema.



Existe un tercer método para realizar identificación, el cual es por medio de los Mínimos Cuadrados Extendido, en donde se podrá observar la matriz THETA_MCE formada por los parámetros *a,b,c,* la gráfica de convergencia de dichos parámetros, la salida real vs. Salida estimada, y una última opción que permite observar como los residuales calculados siguen al ruido.

Todas las ventanas que muestran los diferentes métodos de identificación, presentan en la esquina inferior izquierda la opción de volver al menú de identificación con la finalidad de observar las diferentes matrices THETA con sus respectivos parámetros y poder validarlos. La ventana de MCR presenta en la esquina inferior derecha la opción de ir al diseño del controlador.



CÁLCULO DEL CONTROLADOR

Una vez obtenidos los parámetros de la planta por medio del método de identificación, se procede al cálculo de los polinomios F, G, que son aquellos que contienen la información necesaria para el controlador.

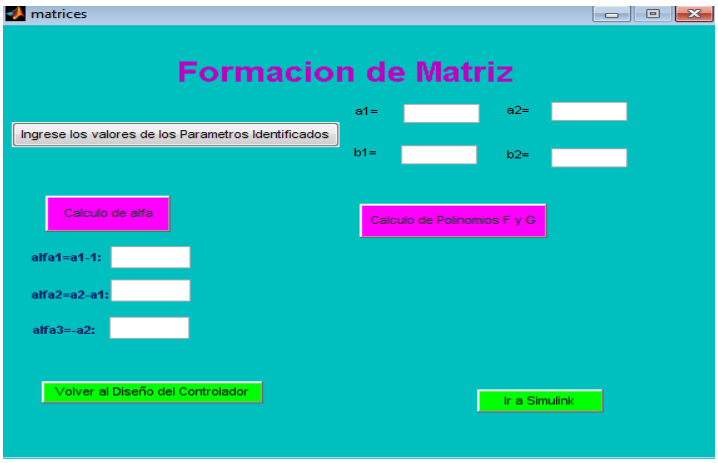
Por esta razón al presionar el botón de Ir al Diseño del Controlador, se desplegará una nueva ventana, en donde se deberán ingresar los polos deseados con los cuales se va a realizar el diseño del controlador.

Se tiene la opción de ingresar polos reales o polos imaginarios en la plano Z, además de ingresar dos o tres polos.

Una vez ingresados los polos, se decide que circuito se desea controlar, un circuito sin presencia de ruido o con ruido.

En cualquiera de los dos casos, se procede a adquirir desde el workspace los parámetros que fueron identificados, el cálculo de los alfas y por último el cálculo de los polinomios F y G.

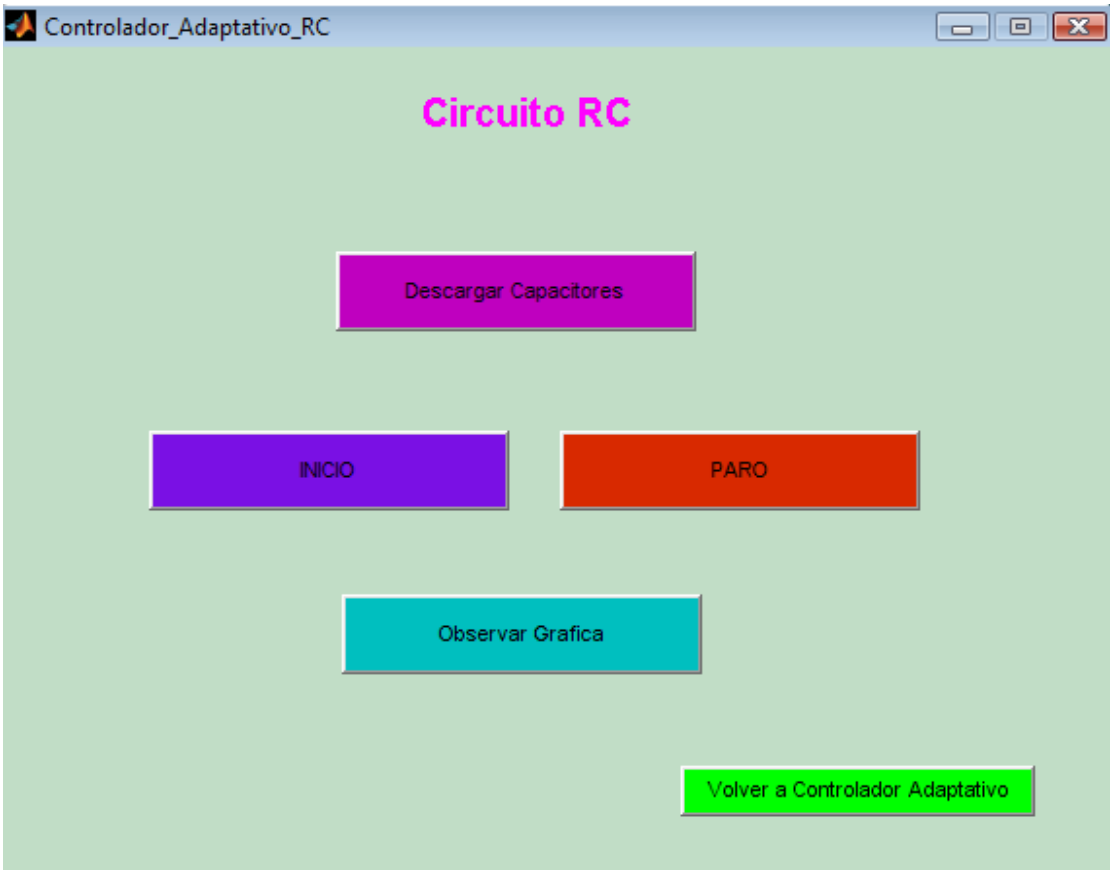
Además se tiene la opción de observar la respuesta de los sistemas en Simulink.



Si se escoge la opción de Tiempo Real, se desplegará la pantalla para ingresar los datos necesarios para realizar el diseño del Controlador Adaptativo por medio de Asignación de Polos.

Se tiene la opción de controlar el sistema RC o el sistema basado en amplificadores operacionales.

En cualquiera de los dos casos, se desplegará una ventana en la que el usuario correrá el programa y lo parará, además de observar la respuesta del mismo.


Controlador_Adaptativo_RC
Circuito RC
Descargar Capacitores
INICIO
PARO
Observar Grafica
Volver a Controlador Adaptativo