



# IDENTIFICACIÓN DE SISTEMAS USANDO UNA HERRAMIENTA DE SOFTWARE LIBRE CON ACCESO A INTERNET

**Autor:** Br. María Elena Noriega Rivas.

**Tutor:** Prof. Pablo Lischinsky.

Proyecto de Grado presentado ante la ilustre UNIVERSIDAD DE LOS ANDES como  
requisito final para optar al Título de INGENIERO DE SISTEMAS.

Mérida, Venezuela

Mayo 2008

## RESUMEN

La identificación de sistemas trata de la estimación de modelos de sistemas dinámicos a partir de los datos observados.

Esta tesis dirige su esfuerzo principalmente hacia la facilitación de la resolución de los problemas de identificación. El objetivo de este trabajo es la construcción de una herramienta de *software* libre que permita la identificación de sistemas, es decir modelos matemáticos de sistemas dinámicos, a partir de datos experimentales de entrada/salida, vía Internet. Se utilizará como prototipo experimental el péndulo, este sistema se identificará en diferentes puntos de operación utilizando modelos lineales discretos de tipo ARX.

Mediante la interfaz gráfica realizada en lenguaje de programación web PHP, conectada a Octave (programa de software libre de métodos numéricos), utilizando programas (*scripts*) similares al del *Systems Identification Toolbox* de Matlab, se aplicó el método de mínimos cuadrados para la estimación de parámetros de los modelos. El usuario tendrá la facilidad de ver las experiencias e identificar modelos a partir de dichas experiencias, en su computador usando un acceso Internet. Esto significa que no necesita tener instalado algún software para efectuar la identificación de sistemas.

<b>1. GENERALIDADES</b>	1
1.1. Introducción	2
Antecedentes	3
1.2. Objetivos	3
Objetivo General	3
Objetivos Específicos	3
1.3. Esquema de Diseño	3
<b>2. IDENTIFICACIÓN DE SISTEMAS</b>	4
2.1. Concepto de Sistema	4
Modelo de un Sistema	4
2.2. Clasificación de Modelos	6
2.3. Identificación de Sistemas	7
El Proceso de Identificación	8
Métodos de Identificación	10
Estructura de los Modelos Lineales Discretos	10
2.4. Métodos para el Ajuste de Parámetros	14
Método de Estimación por Mínimos Cuadrados (MC)	14
Propiedades del Método de Mínimos Cuadrados	17
Criterios para la validación de la Estructura de los Modelos	17
<b>3. IDENTIFICACIÓN DE UN MODELO ARX UTILIZANDO MATLAB Y</b>	
<b>OCTAVE</b>	20
3.1. Uso de Matlab en Identificación de Sistemas	20
3.2. Uso de Octave en Identificación de Sistemas	22
3.3. El Pendubot	23
3.4. El Péndulo	24
3.5. Identificación de un Sistema Discreto	27
3.6. Experiencia con un Punto de Operación $q_{op} = 0 [rad]$	28

Con una entrada PRBS generada a partir de un valor de diferencial amplitud DA=0.05 [Nm] y un diferencial de tiempo DT=0,15 [seg]	28
Con una entrada PRBS generada a partir de un valor de diferencial amplitud DA=0.2 [Nm] y un diferencial de tiempo DT=0,15 [seg]	31
Con una entrada PRBS generada a partir de un valor de diferencial amplitud DA=0.1 [Nm] y un diferencial de tiempo DT=0,30 [seg]	33
Con una entrada PRBS generada a partir de un valor de diferencial amplitud DA=0.2 [Nm] y un diferencial de tiempo DT=0,30 [seg]	36
3.7. Experiencia con un punto de operación $q_{op} = \frac{\pi}{6} [rad]$	39
Con una entrada PRBS generada a partir de un valor de diferencial amplitud DA=0.05 [Nm] y un diferencial de tiempo DT=0,15 [seg]	39
Con una entrada PRBS generada a partir de un valor de diferencial amplitud DA=0.1 [Nm] y un diferencial de tiempo DT=0,15 [seg]	42
Con una entrada PRBS generada a partir de un valor de diferencial amplitud DA=0.1 [Nm] y un diferencial de tiempo DT=0,30 [seg]	45
Con una entrada PRBS generada a partir de un valor de diferencial amplitud DA=0.2 [Nm] y un diferencial de tiempo DT=0,30 [seg]	47
<b>4. INTERFAZ WEB CON PHP Y OCTAVE</b>	<b>51</b>
<b>5. CONCLUSIONES Y RECOMENDACIONES</b>	<b>75</b>
<b>6. BIBLIOGRAFÍA</b>	<b>77</b>

**APÉNDICE A: PROGRAMAS REALIZADOS EN OCTAVE .....79**

1. Ejemplo1.m	.....79
2. Ejemplo2.m	.....81
3. Ejemplo3.m	.....83
4. arx2g.m	.....84

**APÉNDICE B: PROGRAMAS REALIZADOS EN PHP .....95**

1. inicio.php	.....95
2. menu.php	.....97
3. experiencia.php	.....98
4. ejemplo.php	.....102

# CAPITULO 1

## GENERALIDADES

### 1.1. Introducción

El desarrollo de modelos matemáticos de sistemas dinámicos controlados a partir de datos experimentales de entrada/salida conocido como identificación de sistemas, es un paso fundamental en el proceso de desarrollo de sistemas de control.

La identificación de sistemas de modelos discretos lineales de una entrada y una salida consiste en una estimación de parámetros por medio de la técnica de mínimos cuadrados, entre otras, para lo cual se requiere de un *software* que permita realizar los cálculos necesarios. En el área de control se utiliza comúnmente la herramienta Matlab y una de sus librerías llamada *Systems Identification Toolbox*, el cual es un *software* costoso.

Es por ello que proponemos desarrollar algunos programas de identificación de sistemas mediante el método de mínimos cuadrados usando una herramienta de *software* libre de cálculo numérico llamado Octave. Posteriormente exploraremos la facilidad de conexión Web de este software, para que pueda ser fácilmente utilizada por este medio.

## Antecedentes

Para realizar identificación de sistemas se utilizan generalmente herramientas computacionales, entre ellas la herramienta Matlab, un paquete comercial, es decir, su uso incurre en costos; es por esto que en este trabajo se utilizará Octave [14] para identificar modelos matemáticos de sistemas dinámicos controlados a partir de datos experimentales de entrada/salida.

En muchos casos el usuario no dispone de este software en su computador, pero una alternativa es que los usuarios puedan conectarse vía Internet para realizar sus experimentos. Esta idea ha sido utilizada en universidades en todo el mundo. Alguna de estas aplicaciones está descrita en el artículo [11].

En la Escuela de Ingeniería de Sistemas se han realizados poco trabajos basados en la enseñanza a distancia, algunos de ellos son los trabajos realizados sobre Laboratorios Virtuales remotos [12], y cálculo numérico vía Web con PHP y GNU Octave [10]. En otro trabajo se ha utilizado la herramienta GNU Octave para realizar simulaciones de sistemas de control no lineales [9].

Por otro lado, se ha desarrollado una serie de prácticas de laboratorio utilizando el sistema de desarrollo a tiempo real dSPACE y el primer enlace del proceso Pendubot que resulta en un péndulo controlado (robot manipulado de un grado de libertad) [8].

En la Web se encuentran muchos laboratorios virtuales que utilizan la herramienta Matlab para realizar aplicaciones de diferentes tipos; sin embargo, no puede decirse lo mismo de la herramienta Octave.

## 1.2. Objetivos

### Objetivo General

Construir una herramienta de software libre vía Internet que permita la identificación de sistemas, es decir, modelos matemáticos de sistemas dinámicos controlados a partir de datos experimentales de entrada/salida.

### Objetivos Específicos

- Identificar el péndulo (Pendubot) del Laboratorio de Sistemas de Control.
- Diseñar y realizar las experiencias entrada/salida en los diferentes puntos de operación del péndulo y registrar los datos resultantes.
- Elaborar algunos programas (*scripts*) similares al del *Systems Identification Toolbox* de Matlab utilizando Octave (programa de *software* libre de métodos numéricos), entre ellos mínimos cuadrados para la estimación de parámetros de los modelos de tipo ARX.
- Permitir que el usuario pueda cargar los archivos de datos experimentales e identificar los modelos a partir de ellos en su computador, empleando los programas previamente desarrollados usando un acceso a Internet.
- Permitir que el usuario pueda ejecutar los *scripts* elaborados en Octave, para identificar el sistemas sin necesidad de tener un software instalado en su computador.

## 1.3. Esquema De Diseño

La aplicación funciona en un servidor Web, el usuario de la interfaz Web se conecta al servidor mediante un enlace http. El servidor procesa la petición de entrada e interpreta las órdenes que se envían a través del formulario.

Los datos capturados en el formulario se envían a un fichero tipo *script*. El programa Octave se encarga de procesar el cálculo realizando la estimación de los parámetros usando el método de mínimos cuadrados y retorna el resultado. También se le permitirá al usuario descargar los *scripts* y ejecutar, si lo desea, los comandos en Octave instalados en su computador.



## CAPITULO 2

### IDENTIFICACIÓN DE SISTEMAS

#### 2.1. Concepto de Sistema

Es un conjunto organizado de elementos o partes interactuantes e interdependientes, que se relacionan formando un todo unitario y complejo.

Cabe aclarar que los elementos o partes que componen al sistema, no se refieren al campo físico (objetos), sino más bien al funcional. De este modo los elementos o partes pasan a ser funciones básicas realizadas por el sistema.

Las señales observables de interés para el observador se denominan salidas del sistema, mientras que las señales que pueden ser manipuladas libremente son las entradas del mismo. El resto de señales que influyen en la evolución de las salidas pero no pueden ser manipuladas por el observador se denominan perturbaciones [13], (Figura 2.1).

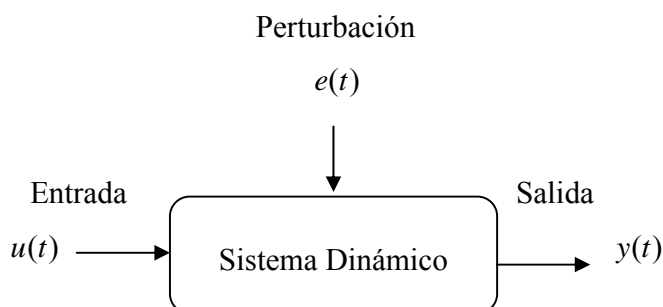


Figura 2.1. Sistema dinámico con entrada  $u(t)$ , perturbación  $e(t)$  y salida  $y(t)$ .

#### Modelo de un Sistema

La primera operación que se debe realizar cuando se aborda un problema de control es la obtención de un modelo matemático del proceso que se pretende controlar. El modelo debe tratar de reproducir lo mejor posible las respuestas de un proceso  $y(t)$  ante una entrada  $u(t)$ .

Para poder aplicar la teoría de control se deben obtener modelos matemáticos (modelos en variables de estado, en funciones de transferencia, etc.) que básicamente se generan de dos formas:

- El modelado físico, es el procedimiento a seguir para construir modelos matemáticos utilizando las leyes físicas fundamentales (de conservación de masa, energía, etc.) para reproducir el comportamiento del sistema. No todos los sistemas pueden ser modelados a través de las leyes que lo rigen ya sea por su complejidad o porque las leyes físicas que lo gobiernan se desconocen.
- La identificación del sistema, que consiste en construir modelos matemáticos a través de datos experimentales o de registros de las señales de entrada y salida (ver figura 2.1). La identificación de sistemas también es conocida como modelado empírico ya que es fundamentalmente experimental.

El campo de aplicación del modelado teórico es muy restricto; en muchos casos la estructura del modelo obtenido a partir del conocimiento físico posee un conjunto de parámetros desconocidos, que sólo se pueden determinar experimentando sobre el sistema real. De ahí la necesidad de recurrir a los métodos de *identificación de sistemas*.

Los modelos obtenidos mediante técnicas de identificación tienen las siguientes desventajas:

1. Su rango de validez suele ser limitado (sólo son aplicables a un determinado punto de operación, un determinado tipo de entrada o un proceso concreto), es decir es válido solamente para el sistema estudiado.
2. La identificación solo puede ser aplicada a sistemas existentes. En muchos casos los parámetros identificados no tienen relación con magnitudes físicas.

## 2.2 Clasificación de los Modelos

Dependiendo de los principios considerados, se puede hacer una distinción entre los siguientes modelos: [1]

- **Modelos Mentales** son un conjunto de ideas, imágenes que solo existen en la mente del hombre, involucran todas las relaciones con el medio que lo rodea. Estos modelos carecen de formalismo matemático.
- **Modelos Matemáticos**, corresponden a una ecuación matemática o un conjunto de ellas en base a las cuales se puede conocer el comportamiento del sistema. En otras palabras son construcciones abstractas y simplificadas para predecir su comportamiento.

Los modelos pueden ser clasificados de otras maneras, como por ejemplo:

- **Modelos Análogos**, los cuales son basados en analogías entre procesos en diferentes áreas. Por ejemplo un oscilador mecánico y uno eléctrico pueden ser descritos por la misma ecuación diferencial lineal de segundo-orden, pero los coeficientes tendrán diferentes interpretaciones físicas.
- **Modelos Físicos**, que son en su mayoría trabajos a escala, las unidades tienen las mismas características esenciales que los modelos de los procesos.

Los modelos matemáticos pueden ser derivados de dos maneras:

- **Modelando**, se refiere a la derivación de los modelos básicos de las leyes de la física, económicos, etc.
- **Identificación**, se refiere a la determinación de modelos dinámicos utilizando o basándose en datos experimentales.

Los modelos matemáticos de sistemas dinámicos pueden ser clasificados de diferentes maneras. Tales modelos describen cómo el efecto de una señal de entrada puede influenciar en el comportamiento del sistema en tiempos posteriores.

Una manera de clasificar los modelos dinámicos es la siguiente:

- **Modelos una entrada, una salida (SISO) – modelos multivariantes.** Los modelos SISO se refieren a procesos donde la descripción viene dada por la influencia de una entrada y una salida. Cuando se involucran más variables resulta un modelo multivariable.
- **Modelos lineales – modelos no lineales.** Un modelo es lineal si la salida depende linealmente de la entrada y de los posibles ruidos; en otro caso es no lineal. Se representan por Ecuaciones Diferenciales Ordinarias (EDO), lineales (EDOL) o no lineales (EDONL).
- **Modelos invariantes en el tiempo – modelos variantes en el tiempo.** Un modelo es invariante en el tiempo cuando las propiedades del sistema modelado se consideran constantes en el tiempo. De lo contrario, los modelos tienen parámetros que cambian con el tiempo.
- **Modelos paramétricos – modelos no paramétricos.** Un modelo paramétrico es descrito por un conjunto de parámetros, los modelos no paramétricos no se encuentran sujetos a ninguna forma funcional, emplean formas funcionales flexibles que aproximen la función objetivo, algunos ejemplos pueden consistir en una función o un gráfico.
- **Modelos Causales y No Causales.** Un sistema es causal si depende sólo de las condiciones presentes y pasadas, pero no de las futuras, es decir, hay una relación de causalidad.
- **Modelos Estocásticos y Determinísticos.** En ocasiones existen variables que afectan el sistema, pero no es posible predecir el valor que éstas puedan tomar; una de las alternativas para hacer frente a estos casos consiste en considerar que esa variable es aleatoria, y buscar técnicas basadas en la teoría de probabilidades para analizar el sistema. Un modelo que incluya variables aleatorias es un modelo estocástico, mientras que modelos exentos de aleatoriedad se denominan modelos determinísticos. Para un modelo determinísticos la salida puede ser exactamente calculada tan pronto como se conozca la señal de entrada. Los modelos estocásticos contienen términos aleatorios que hacen imposible este cálculo exacto.
- **Modelos en tiempo continuo – discreto.** Los sistemas continuos describen las relaciones entre señales continuas, y se caracterizan mediante ecuaciones diferenciales. Los sistemas discretos trabajan con señales muestreadas o medidas, y son descritos mediante ecuaciones en diferencias.

### 2.3. Identificación de Sistemas

La identificación de sistemas consiste en construir modelos matemáticos de sistemas dinámicos basándose en las entradas y salidas observadas. Se ajusta el modelo a las observaciones realizadas sobre el sistema.

El modelo obtenido a partir de datos experimentales de entrada y salida aquí derivado se conoce como **Modelo de Caja Negra ó Modelo Entrada – Salida**, no se centra en estudiar el interior del sistema sino en su comportamiento respecto al entorno.

Identificar un sistema consiste encontrar un conjunto de *reglas* y *parámetros* asociados que describan un modelo aceptable para el proceso que en él se está llevando a cabo.

#### El Proceso de Identificación

En términos generales, el proceso de identificación comprende los siguientes pasos:

1. **Obtención de datos entrada - salida.** Se debe excitar el sistema mediante la aplicación de una señal de entrada y registrar la evolución de sus entradas y salidas durante un intervalo de tiempo.

Los aspectos más importantes que deben considerarse para la elección de la entrada se pueden resumir en [2]:

- Las señales binarias frecuentemente son convenientes para identificar sistemas lineales.
- Elegir el rango de frecuencia de entrada de manera que tenga la mayor parte de su energía en las bandas de frecuencia que son importantes para el sistema, donde particione los puntos del diagrama de Bode.
- Es una buena idea generar primero la secuencia de entrada en una manera off-line y examinar sus características antes de aplicarlo al sistema.

2. **Tratamiento previo de los datos capturados.** Es necesario observar y reparar los datos erróneos o innecesarios, eliminando tendencias a altas frecuencias.

3. **Elección de la estructura del modelo.** Se define el tipo de modelo a utilizar: lineal o no lineal, discreto o continuo, es decir, hay que determinar la estructura deseada. Para ello se debe tener un conocimiento previo del sistema (conocer la leyes físicas que rigen el proceso).

4. **Obtención de los parámetros del modelo.** Se escoge el método de estimación de parámetros a utilizar y se estiman los parámetros que mejor se ajustan al modelo real (respuesta obtenida experimentalmente a través de los datos entrada-salida).

5. **Validación del modelo.** El último paso consiste en comparar el sistema con el comportamiento del modelo y evaluar su diferencia. Se debe elegir un criterio para valorar la calidad del modelo. Los pasos del proceso de identificación se observan en el organigrama de la figura 2.2 [2].

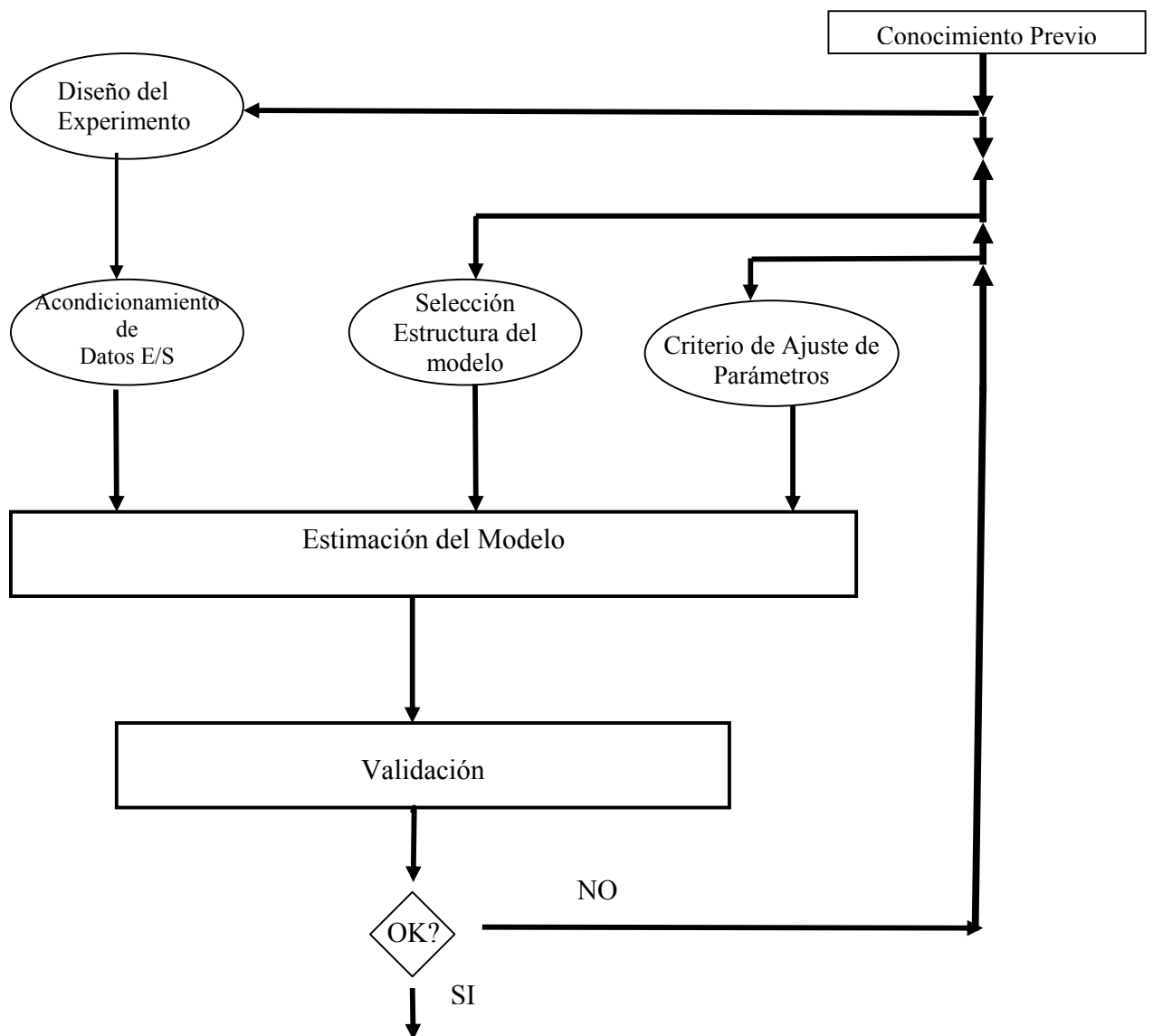


Figura 2.2 El proceso de identificación

## Métodos de Identificación

Los métodos de identificación se pueden clasificar, de acuerdo al algoritmo utilizado, en algoritmos recursivos y no recursivos. Los algoritmos recursivos hacen una llamada a la función desde la misma función, se utiliza como punto de partida un conjunto de datos que previamente fueron estimados. En el caso de algoritmos no recursivos se calcula en un solo paso el valor de los parámetros, partiendo de una secuencia de datos almacenados.

Según el modelo obtenido se puede clasificar en:

- a. *Métodos no paramétricos*, permiten obtener modelos no paramétricos del sistema bajo estudio. Algunos de estos métodos son: análisis transitorios, análisis de la respuesta en frecuencia, análisis de la correlación, análisis espectral y análisis de Fourier.
- b. *Métodos paramétricos*, obtiene modelos paramétricos.

Dependiendo de la aplicación se clasifican en:

1. *Métodos de identificación off-line (a posteriori)*, los datos adquiridos del sistema en estudio son almacenados, luego son transferidos a un computador para su evaluación y procesamiento.
2. *Métodos de identificación on-line*, los datos son directamente procesados en tiempo real.

Existen diversos métodos matemáticos para ajustar los parámetros a un conjunto de datos de entrada-salida. Algunos de los más utilizados son el método de mínimos cuadrados el cual es utilizado en este trabajo siendo explicado posteriormente y el método de las variables instrumentales.

## Estructura de los modelos lineales discretos

Los métodos de estimación paramétricos están muy relacionados con el modelo utilizado. La forma general de representar la estructura de un modelo discreto es, según [Ljung94]:

$$y(t) = G(q) u(t) + H(q) e(t) \quad (2.3)$$

Donde  $q$  es el operador de retardo y  $e(t)$  es un ruido blanco, lo que significa que es una señal aleatoria idénticamente distribuida, no se encuentra correlacionada a lo largo del tiempo y por lo tanto tiene covarianza cero. Su densidad espectral es constante a lo largo de todo el rango de

frecuencia y carece de memoria.  $G(q)$  y  $H(q)$  son funciones racionales del operador  $q$ , siendo filtros de orden finito que modelizan la parte determinista y la parte estocástica respectivamente.

Tanto  $G(q, \theta)$  como  $H(q, \theta)$  son funciones de transferencia con cocientes de polinomio del tipo:

$$G(q, \theta) = \frac{B(q)}{F(q)} = \frac{b_1 q^{-nk} + b_2 q^{-nk-1} + \dots + b_{nb} q^{-nk-nb+1}}{1 + f_1 q^{-1} + \dots + f_{nf} q^{-nf}} \quad (2.3.1)$$

donde,  $nk$  es el número de retardos en la entrada de periodos de muestreo.

$$H(q, \theta) = \frac{C(q)}{D(q)} = \frac{1 + c_1 q^{-1} + \dots + c_{nc} q^{-nc}}{1 + d_1 q^{-1} + \dots + d_{nd} q^{-nd}} \quad (2.3.2)$$

A partir del modelo (2.3) se puede definir una familia de modelos lineales discretos.

#### Modelos de error de salida, OE:

Los polinomios de la ecuación (2.3.2) se convierten en la unidad, es decir:  $C(q)=D(q)=1$ ,

Resultando: 
$$y(t) = \frac{B(q)}{F(q)} u(t) + e(t) \quad (2.3.3)$$

La figura 2.3 representa el modelo OE:

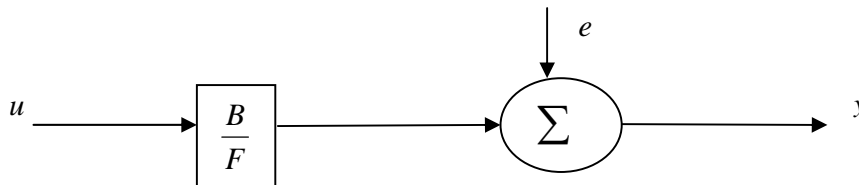


Figura 2.3 Modelo de Error de Salida OE.



**Modelo Box-Jenkins, BJ:**

Los polinomios de la ecuación (2.3.1) y (2.3.2), se hacen no nulos, obteniéndose la siguiente ecuación:

$$y(t) = \frac{B(q)}{F(q)} u(t) + \frac{C(q)}{D(q)} e(t) \quad (2.3.4)$$

La siguiente figura representa el modelo Box- Jenkins:

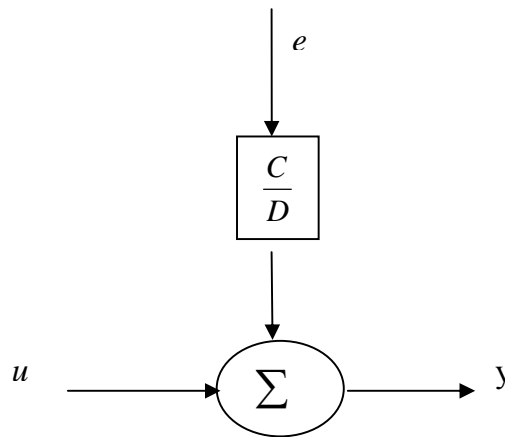


Figura 2.4. Modelo Box-Jenkins.

**Modelos Autoregresivos de Media Móvil y Variables Exógenas, ARMAX:**

En este caso los polinomios  $F(q)$  y  $D(q)$  de la ecuación (2.3.1) y (2.3.2) son iguales a  $A(q)$ , esto quiere decir  $F(q)=D(q)=A(q)$ ,

donde:

$$A(q) = 1 + a_1 q^{-1} + \dots + a_{na} q^{-na} \quad (2.3.5)$$

y  $na$  es el orden del polinomio  $A$ .

Lo que resulta:

$$A(q)y(t) = B(q)u(t) + C(q)e(t) \quad (2.3.6)$$

La figura siguiente representa el modelo ARMAX

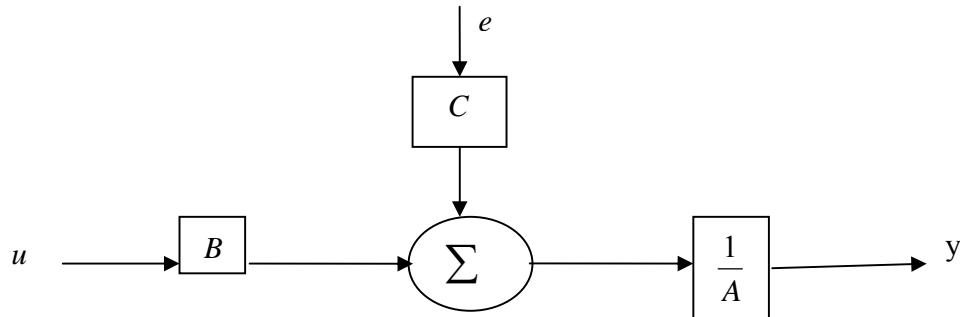


Figura 2.5. Modelo ARMAX.

### Modelos Autoregresivos con Variables Exógenas, ARX:

Los polinomios  $F(q)$ ,  $D(q)$ ,  $A(q)$  y  $C(q)$  de la ecuación (2.3.1) y (2.3.2) toman el valor de la unidad, esto quiere decir:  $F(q)=D(q)=A(q)$  y  $C(q) = 1$ .

resultando: 
$$A(q)y(t) = B(q)u(t) + e(t) \quad (2.3.7)$$

El modelo ARX esta representado en la siguiente figura:

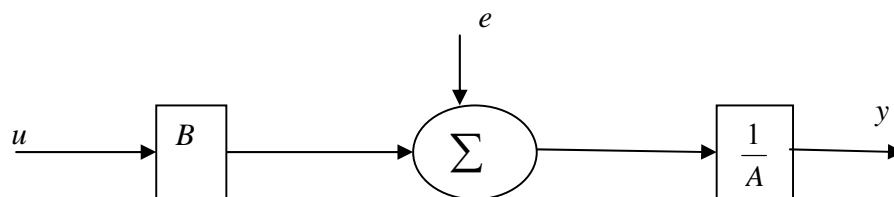


Figura 2.6. Modelo ARX.

En la tabla siguiente se resumen las distintas familias de modelos lineales discretos:

Tabla 1. Resumen de las Diferentes Estructuras de Modelos Paramétricos

Tipo de Modelo	Condición	Estructura resultante
Modelo ARX	$F(q)=D(q)=C(q) = 1$	$A(q)y(t) = B(q)u(t) + e(t)$
Modelo Output Error (OE)	$C(q)=D(q)=1$	$F(q)y(t) = B(q)u(t) + e(t)$
Modelo ARMAX	$F(q)=D(q)=1$	$A(q)y(t) = B(q)u(t) + C(q)e(t)$
Modelo Box-Jenkins (BJ)	$A(q) = 1$	$y(t) = \frac{B(q)}{F(q)}u(t) + \frac{C(q)}{D(q)}e(t)$

## 2.4. Métodos para el Ajuste de Parámetros

Una vez elegida la estructura del modelo (tanto el tipo - ARX, ARMAX, BJ, OE...- como los órdenes de cada polinomio), es necesario determinar el valor de los parámetros que se ajustan a los datos experimentales de entrada - salida. Esta etapa del proceso de identificación se ve facilitada por la existencia de herramientas de *software* que proporcionan diferentes algoritmos para el ajuste de parámetros. Una de estas herramientas es el *Toolbox de Identificación* de Matlab [16].

### Método de Estimación por Mínimos Cuadrados (MC)

La regresión lineal es simplemente un tipo de modelo paramétrico. La correspondiente estructura del modelo puede ser escrita como:

$$y(t) = \varphi^T(t) \theta \quad (2.4)$$

donde  $y(t)$  es una cantidad medida (o salida del sistema),  $\varphi(t)$  es un vector de  $n$  cantidades conocidas y  $\theta$  es un vector de  $n$  parámetros desconocidos. Los elementos del vector  $\varphi(t)$  son frecuentemente llamados variables de regresión o regresor, mientras que  $y(t)$  es llamada variable regresora [1].

El problema es encontrar el estimador  $\hat{\theta}$  del vector de parámetros  $\theta$ . La ecuación (2.4) se reduce a un sistema de ecuaciones lineales, esto puede ser escrito matricialmente como:

$$Y = \phi\theta. \quad (2.4.1)$$

Si la matriz es no singular, el sistema de ecuaciones lineales puede ser fácilmente resuelto para  $\theta$ .

Ahora se introduce la ecuación del error

$$\varepsilon(t) = y(t) - \varphi^T(t)\theta, \quad (2.4.2)$$

Siendo  $\varepsilon$  un vector definido como:

$$\varepsilon = \begin{pmatrix} \varepsilon(1) \\ \vdots \\ \varepsilon(N) \end{pmatrix} \quad \text{donde } N \text{ es el número de datos.}$$

La estimación de  $\theta$  por mínimos cuadrados es definida como el vector  $\hat{\theta}$  que minimiza la función de pérdida

$$V(\hat{\theta}) = \frac{1}{2} \sum_{t=1}^N \varepsilon^T(t) = \frac{1}{2} \varepsilon^T \varepsilon = \frac{1}{2} \|\varepsilon\|^2 \quad (2.4.3)$$

Donde  $\|\cdot\|$  denota la norma del vector. La solución al problema de optimización visto arriba está dado por el siguiente lema:

Considere la función de pérdida dada en (2.4.3), la cual tiene un único punto mínimo dado por:

$$\hat{\theta} = (\phi^T \phi)^{-1} \phi^T Y \quad (2.4.4)$$

Para su comprensión se explicará el método de mínimos cuadrados utilizando una estructura de modelo ARX (2.3.7). Estos modelos se pueden representar por la ecuación:

$$y(t) = \varphi^T(t)\theta + v(t) \quad (2.4.5)$$

siendo:  $y(t)$  es el valor de la salida en el instante  $t$ ,  $\varphi(t)$  es el vector regresor o información, y  $\theta$  es el vector de parámetros.

$$\varphi(t) = [-y(t-1) \dots -y(t-na) \ u(t-nk-1) \dots u(t-nb-nk)]^T \quad (2.4.6)$$

$$\theta = [a_1 \dots a_{na} \ b_1 \dots b_{nb}]^T$$

El problema a resolver consiste en estimar el vector de parámetros,  $\hat{\theta}$ , partiendo de las  $N$  observaciones realizadas:  $y(1), u(1), \dots, y(N), u(N)$ .

De la ecuación (2.4.5) se deduce un conjunto de ecuaciones lineales:

$$\begin{aligned} y(1) &= \phi^T(1)\hat{\theta} \\ y(2) &= \phi^T(2)\hat{\theta} \\ &\vdots \\ y(N) &= \phi^T(N)\hat{\theta} \end{aligned} \quad (2.4.7)$$

que pueden ser escritas en forma matricial según:

$$Y = \Phi \hat{\theta}$$

siendo:  $Y$  un vector de dimensión  $N$ ,  $Y = [y(1) y(2) \dots y(N)]^T$  y  $\Phi = [\phi(1) \phi(2) \dots \phi(N)]^T$ .

El error de modelización o residuo se define como:

$$\varepsilon = Y - \Phi \hat{\theta} \quad (2.4.8)$$

con  $\varepsilon = (\varepsilon(1) \dots \varepsilon(N))^T$ .

La estimación por mínimos cuadrados, consiste escoger  $\hat{\theta}$  que minimice la función residuo,  $V(\hat{\theta})$ , definida por la ecuación (2.4.3).

Sustituyendo la ecuación (2.4.8) en (2.4.3), se obtiene que la función a minimizar es:

$$\min_{\hat{\theta}} V(\theta) = V(\hat{\theta}) = \frac{1}{2} [Y^T Y - Y^T \Phi \hat{\theta} - \hat{\theta}^T \Phi^T Y + \hat{\theta}^T \Phi^T \Phi \hat{\theta}]. \quad (2.4.9)$$

La derivada de la función (2.4.9) respecto a los parámetros debe ser nula:

$$0 = \frac{dV(\theta)}{d\theta} = -Y^T \Phi + \hat{\theta}^T (\Phi^T \Phi),$$

Deduciéndose que el valor de los parámetros que minimiza  $V(\hat{\theta})$  es:

$$\hat{\theta} = (\phi^T \phi)^{-1} \phi^T Y \quad (2.4.10)$$

La ecuación anterior se puede escribir como un producto de sumas finitas:

$$\hat{\theta} = \left[ \sum_{t=1}^N \varphi(t) \varphi(t)^T \right]^{-1} \left[ \sum_{t=1}^N \varphi(t) y(t) \right] \quad (2.4.11)$$

de la cual, conocido el orden del modelo:  $na$ ,  $nb$  y  $nk$ , es fácilmente calculable el valor de sus parámetros.

### Propiedades del Método de Mínimos Cuadrados

Considere la estimación (2.4.12) asumimos que los datos obtenidos de  $y(t) = \varphi^T(t) \theta_0 + e(t)$

$e(t)$  es un ruido blanco de media cero y variancia  $\lambda^2$  [1];

$\hat{\theta}$  converge a  $\theta_0$  cuando  $N$  tiende a infinito.

$\hat{\theta}$  es estimador insesgado de  $\theta_0$ . Un estimador es insesgado si  $E(\hat{\theta}) = \theta_0$  [5];

La matriz de covarianza de  $\hat{\theta}$  es dada por  $\text{cov}(\hat{\theta}) = \lambda^2 (\phi^T \phi)^{-1}$ ;

Un estimador insesgado de  $\lambda^2$  esta dado por:  $S^2 = 2 V(\hat{\theta}) / (N - n)$ , donde  $n$  es el número de parámetros del modelo.

### Criterios para la validación de la estructura de los modelos

Una aproximación natural para determinar la estructura del modelo es simplemente probar diferentes estructuras y comparar el resultado entre ellas. Los métodos de prueba comúnmente utilizados son:

- Test de la función error o pérdida
- Análisis del residuo

### Test de la función de error o pérdida

El método más simple es mirar sencillamente la función de pérdida  $V(\hat{\theta})$  directamente vinculada con el orden del modelo. Cuando incrementa el orden del modelo la función de pérdida decrece hasta que se mantiene constante o cambia lentamente. Otros métodos se basan en tests estadísticos de la función de pérdida o en la evaluación de diferentes criterios que tienen en cuenta la complejidad del modelo.

Un método estadístico para evaluar si la función de pérdida disminuye significativamente cuando incrementa el orden del modelo es el *F-test*. Esta prueba se basa en la independencia estadística de  $V^1(\theta)$  y  $V^2(\theta)$  donde los subíndices indican modelos con un número de parámetros  $p_1$  y  $p_2$  respectivamente. Para probar si la reducción de la función es significativa, cuando el número de parámetros aumenta de  $p_1$  a  $p_2$  se utiliza:

$$t = \frac{V^1(\theta) - V^2(\theta)}{V^2(\theta)} \frac{N - P_2}{P_1 - P_2} \quad (2.6)$$

Esta cantidad tiene una distribución  $F[p_1 - p_2, N - p_2]$ , entonces para un suficiente número de muestras  $N$ , se cumple la hipótesis:  $t > F_{\alpha}[p_1 - p_2, N - p_2]$  que la reducción de la función de pérdida es significativa con el aumento del número de parámetros, por lo tanto, el nuevo parámetro es aceptado con un nivel de confianza de  $1 - \alpha$ . En la práctica, el nivel de confianza tiene un rango de 0.01 a 0.1.

El criterio de análisis de complejidad puede realizarse simplemente adicionando a la función de pérdida un término extra que penalice la complejidad del modelo. En este caso se selecciona el mejor modelo que minimice el criterio.

El criterio de información de Akaike (AIC) disminuye con la función pérdida y aumenta con el número de parámetros, se define como:

$$AIC = \log(V) + \frac{2P}{N} \quad (2.7)$$

Donde  $V$  es la función de pérdida,  $P$  es el número de parámetros y  $N$  es el número de datos.

Otro criterio propuesto por Akaike es *Final Prediction Error Criterion (FPE)*:

$$FPE(p) = \frac{N + P}{N - P} V(\theta) \quad (2.8)$$

Una propiedad atractiva, tanto del criterio de AIC como FPE, es que el orden del modelo se determina como resultado de valor mínimo del criterio y no es necesario evaluarlo en función de unos niveles de confianza como es el caso del método estadístico.

### Análisis del Residuo

Se compara la salida real con la predicción del modelo, como sigue:

$$\varepsilon(t, \hat{\theta}) = y(t) - \hat{y}(t, \hat{\theta}). \quad (2.9)$$

Este error es una manera de representar las diferencias entre las variables observadas y el comportamiento del modelo estimado. Si el modelo es una buena representación del sistema, los residuos no deben estar correlacionados con la entrada.

La pregunta crucial, una vez identificado el modelo, es si el modelo obtenido es lo suficientemente bueno para los objetivos considerados. La validación permite comprobar si el modelo identificado representa el comportamiento real, sin embargo no olvidemos que ningún sistema puede ser modelado exactamente, siempre existirá incertidumbre en los modelos. El problema planteado tiene distintos aspectos:

1. El modelo, ¿satisface suficientemente bien los datos observados?
2. ¿Es el modelo suficientemente bueno para los requerimientos propuestos?
3. ¿Describe el modelo al sistema real?

En este trabajo para contestar estas interrogantes, se utilizará la prueba del criterio de Akaike y se estudiara la función de pérdida, a medida que este valor es menor el modelo se acerca mas al comportamiento real del sistema.



## **CAPITULO 3**

### **IDENTIFICACIÓN DE UN MODELO ARX UTILIZANDO MATLAB Y OCTAVE**

#### **3.1. Uso de Matlab en Identificación de Sistemas**

Matlab es un *software* propietario capacitado para desarrollar proyectos en donde se encuentren implicados elevados cálculos matemáticos y la visualización gráfica de los mismos. Matlab integra análisis numérico, cálculo matricial, proceso de señal y visualización gráfica en un entorno completo; dispone también en la actualidad de un amplio abanico de programas de apoyo especializados, denominados *Toolboxes*, que extienden significativamente el número de funciones incorporadas en el programa principal. Estos *Toolboxes* cubren en la actualidad prácticamente casi todas las áreas principales en el mundo de la ingeniería y la simulación, destacando entre ellos el *toolbox* de procesamiento de imágenes, señal, control robusto, estadística, análisis financiero, matemáticas simbólicas, redes neuronales, lógica difusa, identificación de sistemas, simulación de sistemas dinámicos, etc. Es un entorno de cálculo técnico, que se ha convertido en estándar de la industria, con capacidades no superadas en computación y visualización numérica.

Matlab dispone de una amplia colección de funciones aplicables al campo de la identificación de sistemas, agrupadas en el *System Identification Toolbox* (SIT), que constituyen una herramienta de gran utilidad para la identificación y modelado de sistemas dinámicos. Estas funciones incluyen diferentes algoritmos para el ajuste de parámetros en todo tipo de modelos lineales.

Así mismo, permiten examinar las propiedades de los modelos obtenidos, e incluso realizar un preprocesamiento de los datos utilizados para la identificación, en caso de que sea necesario.

#### **Los datos de entrada - salida**

Todo proceso de identificación parte de un conjunto de datos de entrada-salida obtenidos de forma experimental a partir del sistema físico que se pretende modelar. Los datos anteriores

suelen encontrarse almacenados en archivos *ascii*, que pueden ser cargados en el Workspace de Matlab mediante la función **load** para el posterior trabajo con los mismos.

Los datos de entrada-salida, cargados en el *toolbox*, deben recibirse en una matriz de dos columnas y  $N$  filas, siendo  $N$  el número de datos (muestras) de entrada-salida registrados. En la primera columna deben aparecer las salidas en sucesivos periodos de muestreo y en la segunda las correspondientes entradas.

### Tratamiento previo de los datos

Uno de los pasos en el proceso de identificación de sistemas es el tratamiento de los datos, el *Toolbox* de Identificación proporciona un conjunto de funciones que permiten realizar diferentes tratamientos a los datos de entrada – salida. Entre ellas existen funciones de filtrado, eliminación de valores medios, etc.

Comandos para la generación y manipulación de datos:

**dtrend** Elimina tendencias en un serie temporal.

**idfilt** Filtrado de datos mediante filtros Butterworth.

**idinput** Generación de señales de entrada para identificación.

**idresamp** Remplaza los datos por interpolación y decimación.

### Identificación Paramétrica

El *Toolbox* de Identificación (SIT) contiene una gran variedad de modelos paramétricos para sistemas lineales. Todos ellos se ajustan a la siguiente estructura general:

$$A(q)y(t) = \frac{B(q)}{F(q)}u(t) + \frac{C(q)}{D(q)}e(t) \quad (3.1)$$

donde  $u(t)$ ,  $y(t)$  y  $e(t)$  son la entrada, salida y ruido del sistema respectivamente, y  $A$ ,  $B$ ,  $C$ ,  $D$  y  $F$  son polinomios función del operador desplazamiento ( $q$ ).

El *toolbox* cuenta con un conjunto de funciones para realizar la estimación paramétrica:

**ar** Estimación de un modelo AR usando diversos criterios como adelante – atrás, el método de mínimos cuadrados, el método de Yule-Walker, el método geométrico enrejado.

**armax** Estimación de parámetros de un modelo ARMAX mediante el método de predicción del error.

**arx** Calcula la estimación de parámetros de un modelo ARX utilizando el método de mínimos cuadrados.

**bj** Estimación de un modelo Box-Jenkins mediante el método de predicción del error.

**oe** Estimación de un modelo Output-error usando el método de predicción del error.

**pem** Estimación de un modelo lineal general usando el método de predicción del error.

### **Validación de modelos**

Los métodos de validación proporcionados por el Toolbox de Identificación se basan en la simulación del modelo obtenido, y la comparación entre su salida y la respuesta real del sistema. Los comandos disponibles son:

**compare** Compara la salida simulada/predicha con la salida real del sistema.

**dlsim** Simulación de un modelo.

**pe** Cálculo de errores de predicción de un modelo.

**Predict** Calcula en m pasos predicciones futuras del modelo.

**resid** Calcula y prueba los residuos asociados al modelo.

### **3.2. Uso de Octave en Identificación de Sistemas**

**Octave** es un programa libre para realizar cálculos numéricos. MATLAB es considerado su equivalente comercial. Entre varias características que comparten se puede destacar que ambos ofrecen un intérprete permitiendo ejecutar órdenes en modo interactivo. Octave tiene una amplia herramienta para resolver problemas de álgebra lineal numérica, la búsqueda de las raíces de ecuaciones no lineales, integrar funciones ordinarias, la manipulación de polinomios, y la integración de ecuaciones diferenciales ordinarias y de ecuaciones algebraicas.

Octave no cuenta con un *toolbox* para identificación de sistemas, sin embargo para cumplir uno de los objetivos del proyecto se realizó algunos scripts utilizando el método de mínimos cuadrados para estimar un modelo de estructura ARX. Ver apéndice A el script `arx2g.m`. Para

obtener los datos de entrada y salida que se utilizarán en la estimación del modelo del sistema Pendubot del Laboratorio de Sistemas de Control.

El SIT ofrece una técnica de identificación que se encuentra implementada con el comando **arx**, se pudiera pensar porque no agrupar las carpetas que contiene este comando e implementarlas en octave, sin embargo no es tan sencillo como se piensa ya que este llama a su vez un conjunto de funciones propias de Matlab. El comando **arx** del SIT hace aproximadamente unas 24 llamadas, de las cuales unas 12 llamadas son a comandos propios de Matlab; es por esto que se hace imposible copiarlo e implementarlo en Octave. De allí la necesidad de programar una rutina que permita hacer la identificación de un sistema utilizando el método de mínimos cuadrados. Para esta identificación se utilizó la estructura ARX, ya que es la recomendada en las bibliografías más importantes.

### **3.3. El Pendubot**

El Pendubot (Péndulo Robot) es un sistema electromecánico conformado por dos articulaciones rígidas interconectadas entre sí a través de un eje que permite un movimiento de 360 grados entre ellas. El otro extremo de la primera articulación está acoplado directamente al eje del motor de corriente continua a través del cual se efectúa el control, mientras que la articulación 2 es de movimiento libre. El Pendubot puede considerarse como un brazo robótico de dos grados de libertad. Las dos articulaciones pueden ser pensadas como un péndulo cuyo movimiento es controlado en el primer caso por el par aplicado al motor, y en el segundo caso por el accionar de la articulación 1 [8].

Las articulaciones del Pendubot son de aluminio con 0,635cm de espesor cada una. La articulación 1 posee 15,25cm de longitud y está acoplada directamente al eje de un motor de corriente continua de 90V, de imán permanente controlado por corriente, mientras que la articulación 2 es de 22,86cm de longitud. El motor es manejado a través de un servo amplificador por modulación de ancho de pulsos (PWM) en modo de torque (o corriente) y ajustado para una ganancia de 1V a la entrada equivalente a 1,2A de salida [8].

El Pendubot presenta una serie de características interesantes que pueden ser usadas en el campo de la investigación y la educación. Se pueden implementar diversos métodos de control tales como control lineal, control no lineal, control adaptativo y robusto, lógica difusa, control inteligente, control híbrido y otros, así como desarrollar aplicaciones de identificación, compensación de fricción, control de oscilación o trayectoria, control de balanceo, etc.

### 3.4. El Péndulo

Se denomina péndulo físico a cualquier péndulo real que no tiene toda la masa concentrada en un punto. La figura 3.1 representa un cuerpo real que esta separado con un ángulo  $q$  de su posición de equilibrio. La distancia del eje al centro de gravedad es  $L$ , el momento de inercia de la articulación respecto a su centroide es  $I$ , la masa del péndulo  $m$  y  $cm$  el centro de masa.

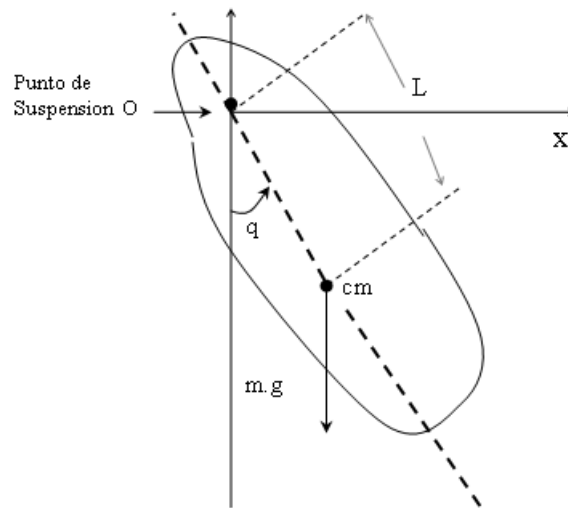


Figura 3.1 Péndulo Físico

#### Representación en Variables de Estado y Linealización del Péndulo

El péndulo de estudio presenta fricción entre el eje de motor y los cojinetes del servomotor. Un brazo robótico es afectado por fricción, por lo tanto el modelo de la dinámica del péndulo considerando la fuerza de fricción presente en el sistema se representa como:

$$(m\ell^2 + I)\ddot{q} + mLg \sin(q) = \tau - F \quad (3.2)$$

Donde  $F$  representa el modelo de fricción del sistema, cuya forma más simplificada se reduce a la fricción viscosa y se expresa como  $F = \alpha \dot{q}$ , que consiste en el término de amortiguamiento o disposición de energía del sistema. La ecuación del péndulo con fricción se expresa como:

$$(mL^2 + I)\ddot{q} + \alpha \dot{q} + mLg \sin(q) = \tau \quad (3.3)$$

A partir de (3.3) la ecuación dinámica del péndulo se puede escribir como:

$$\theta_1 \ddot{q} + \alpha \dot{q} + \theta_2 g \sin(q) = \tau, \quad (3.4)$$

donde  $\theta_1 = (mL^2 + I)$  y  $\theta_2 = mL$  son los parámetros físicos reagrupados, los cuales se consideran constantes,  $\alpha$  la fricción viscosa y  $\tau$  es el control. Haciendo el cambio de variable  $q_1 = q, q_2 = \dot{q}$ , se puede escribir la representación en variables de estado del sistema, obteniéndose:

$$f_1(q_1, q_2, u) = \dot{q}_1 = q_2 \quad (3.5)$$

$$f_2(q_1, q_2, u) = \dot{q}_2 = -\frac{\theta_2 g \sin(q_1)}{\theta_1} - \frac{\alpha q_2}{\theta_1} + \frac{\tau}{\theta_1}.$$

Los puntos de equilibrio del sistema se obtienen al igualar a cero, simultáneamente las ecuaciones diferenciales de primer orden (3.5) y considerar un par de alimentación al sistema nulo  $\tau = 0$ , es decir:

$$f_1(q_1, q_2, u) = 0, \Rightarrow q_2 = 0 \quad (3.6)$$

$$f_2(q_1, q_2, u) = 0, \Rightarrow \sin(q_1) = 0$$

El péndulo en estudio posee dos puntos de equilibrio naturales cuyas coordenadas en el sistema de coordenadas establecido son  $(0,0)$  y  $(\pi, 0)$ .

Las coordenadas generalizadas de los puntos de operación  $q_{op}$  se obtienen al fijar la posición  $q_1 = q_{1op}$  e igualar a cero las ecuaciones (3.6), parametrizando en función de  $q_{1op}$  se tiene que:

$$q_{1op} = q_{op} \quad (3.7)$$

$$q_{2op} = 0$$

$$\tau_{op} = \theta_2 g \sin(q_{op})$$

La no linealidad presente en el modelo del sistema está dada por el término  $\theta_2 g_{sen}(q)$ .

El análisis del sistema y el diseño de los algoritmos de control harán uso de una versión lineal del modelo la cual está basada en la expansión de la serie de Taylor:

$$f_1(q, \tau) = f_0(q_{op}, \tau_{op}) + \frac{\partial F}{\partial q} \Big|_{q_{op}, \tau_{op}} (q - q_{op}) + \frac{\partial F}{\partial \tau} \Big|_{q_{op}, \tau_{op}} (\tau - \tau_{op}) \quad (3.8)$$

siendo  $F = [f_1 f_2]^T$  el vector de las ecuaciones diferenciales de primer orden,  $q$  el vector de estados,  $\tau$  el control de entrada para el péndulo y  $q_{op}$  y  $\tau_{op}$  los valores de equilibrio para los estados y control respectivamente. El modelo linealizado alrededor de la posición de equilibrio es de la forma:

$$\dot{x} = A(x) + B(u) \quad (3.9)$$

$$\text{donde } x = \begin{bmatrix} (q_1 & -q_{1op}) \\ (q_2 & -q_{2op}) \end{bmatrix}; \quad u = [\tau \quad \tau_{op}]$$

La matriz  $A$  del sistema se obtiene al derivar las ecuaciones (3.6) respecto a los estados y evaluarla en  $q_{op}$  y  $\tau_{op}$ .

$$A = \frac{\partial F}{\partial q} = \begin{bmatrix} \frac{\partial f_1}{\partial q_1} & \frac{\partial f_1}{\partial q_2} \\ \frac{\partial f_2}{\partial q_1} & \frac{\partial f_2}{\partial q_2} \end{bmatrix} \Big|_{q_{op}, \tau_{op}} \quad (3.10)$$

La matriz  $B$  del sistema se obtiene al derivar las ecuaciones (3.6) respecto a la señal de control y evaluarla en  $q_{op}$  y  $\tau_{op}$ .

$$B = \frac{\partial F}{\partial \tau} = \begin{bmatrix} \frac{\partial f_1}{\partial \tau} \\ \frac{\partial f_2}{\partial \tau} \end{bmatrix} \Big|_{q_{op}, \tau_{op}} \quad (3.11)$$

El sistema lineal resultante para (3.6) dado por (3.9) tiene la forma:

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -\frac{\theta_2 g \cos(q_{1op})}{\theta_1} & -\frac{\alpha}{\theta_1} \end{bmatrix} x + \begin{bmatrix} 0 \\ \frac{1}{\theta_1} \end{bmatrix} u \quad (3.12)$$

Una vez conocido el modelo dinámico del péndulo se procede hacer la identificación del mismo.

### 3.5. Identificación de un Sistema Discreto

La función de transferencia pulso que se obtuvo en el trabajo [8], al calcular la transformada Z de la planta del sistema con retención tuvo la forma:

$$G(z) = \frac{b_1 z + b_2}{a_2 z^2 + a_1 z + a_0} \quad (3.13)$$

Se realizará la identificación del sistema alrededor de dos puntos de operación. Los puntos de operación fijados para realizar el trabajo experimental tienen como coordenadas de posición:

$$\begin{aligned} q_{1op} &= 0 [rad] & q_{2op} &= 0 & (0, 0) \\ q_{1op} &= \frac{\pi}{6} [rad] & q_{2op} &= 0 & (\frac{\pi}{6}, 0) \end{aligned}$$

En el presente capítulo se hace una comparación de las diferentes experiencias realizadas con el péndulo, haciendo uso del dSPACE sistema de supervisión y control conectada con el software interactivo Cockpit y Trace se siguieron los pasos de la práctica Nro. 2 realizada por González G. [8]. Se obtiene la respuesta temporal del sistema ante una señal de entrada binaria pseudo aleatoria (PRBS) para escalones de diferentes amplitudes 0.05, 0.1 y 0.2,



utilizando un diferencial de tiempo (DT) de 1 y 1/2 constante de tiempo para generar la señal PRBS.

Para cada una de estas experiencias se realiza la identificación del sistema usando una estructura de modelo ARX mediante el método de mínimos cuadrados. La estimación de los parámetros se realiza a través del comando **arx** del SIT haciendo uso del *script* realizado en Octave (Ver apéndice A).

### 3.6. Experiencias en el punto de operación ( $q_{op} = 0 [rad]$ , $q_{2op} = 0$ )

Con una señal de entrada binaria pseudo aleatoria (PRBS) a partir de un valor de diferencial amplitud  $DA=0.05 [Nm]$  y un diferencial de tiempo  $DT=0,15 [seg]$  (1/2 Cte de tiempo).

La siguiente figura 3.1 muestra la señal de entrada PRBS y la salida del sistema generada a partir de un  $DA=0,15 [seg]$  y un  $DT=0,15 [seg]$ .

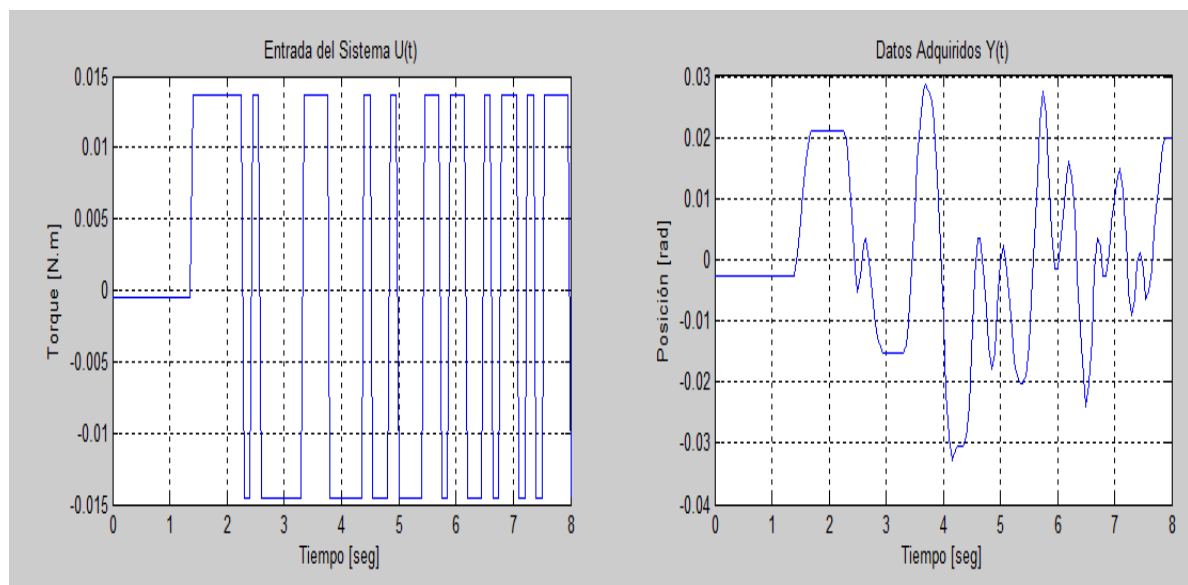


Figura 3.1 Señal de entrada PRBS  $DA=0,05 [Nm]$  y  $DT=0,15 [seg]$  y respuesta del sistema.

Salida del comando **arx** del SIT fijando los ordenes  $na=2$ ;  $nb=2$ ;  $nk=1$ . del modelo ARX.

```
Discrete-time IDPOLY model: A(q)y(t) = B(q)u(t) + e(t)
A(q) = 1 - 1.464 q^-1 + 0.622 q^-2
B(q) = 0.1696 q^-1 + 0.05183 q^-2
Estimated using ARX from data set datos
Loss function 8.34587e-007 and FPE 8.77114e-007
Sampling interval:0.01
```

### Comparación de las Salidas

a. ¿Qué tan bueno es el modelo? Una forma de hacerlo es simularlo y comparar la salida del modelo con la salida medida. Utilizando el comando `Compare` del *System Identification Toolbox*.

```
compare(zv,m1,'r')
```

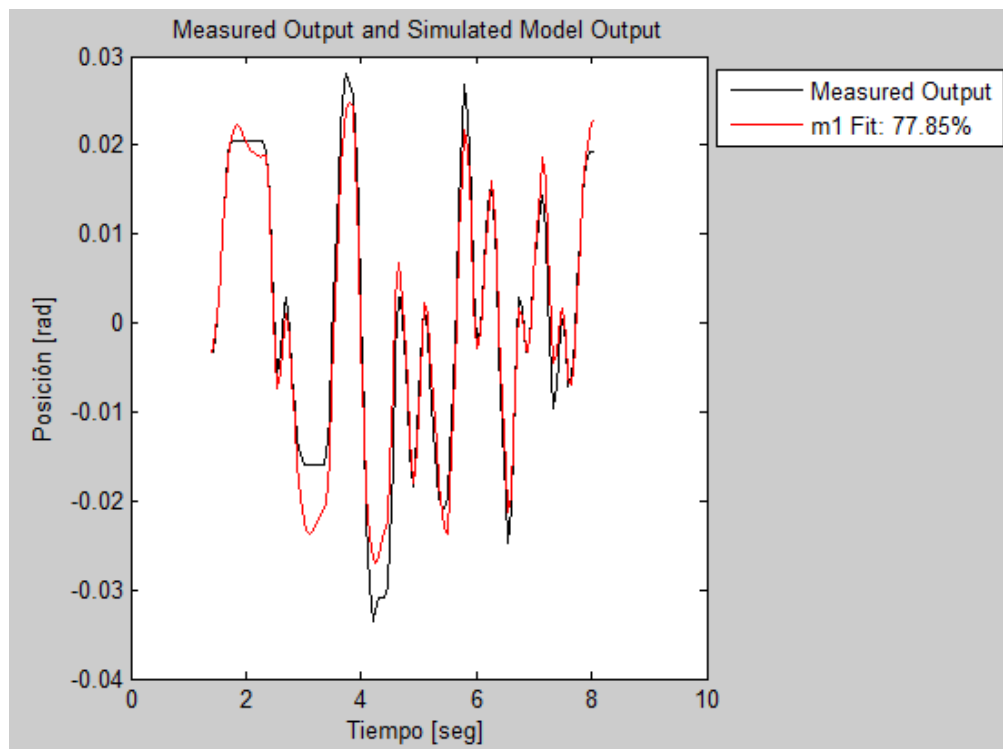


Figura 3.2. Salida real del sistema y la salida estimada con el comando `arx` del SIT.

b. Comparación con el Script `arx2g1.m` realizado en Octave (Ver Apéndice A) para estimar los parámetros del Modelo ARX utilizando el método mínimos cuadrados.

Parámetros	Comando ARX de MATLAB	Script en Octave	Porcentaje de Similitud
$a_1$	-1.464	-1.461	99.80%
$a_2$	0.622	0.619	99.52%
$b_1$	0.1696	0.1692	99.76%
$b_2$	0.05183	0.0533	97.24%

Tabla 2. Parámetros estimados por comando `arx` del SIT y Script realizado en Octave.

c. Comparación de la salida real y la estimada utilizando el *script* realizado en Octave.

Porcentaje de variación entre la salida real y la estimada = 75.59%

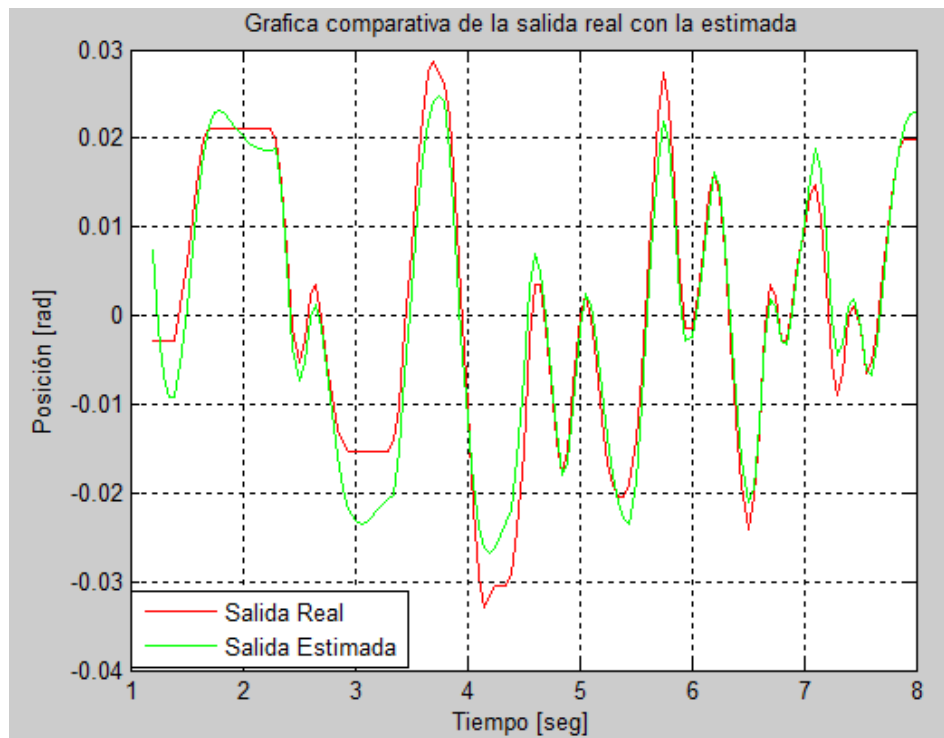


Figura 3.3. Salida real y la Salida Estimada utilizando el Script arx2gl.m realizado en Octave.

Con una señal de entrada PRBS a partir de un valor de diferencial amplitud  $DA=0.2$  [Nm] y un diferencial de tiempo  $DT=0,15$  [seg] (1/2 Cte de tiempo).

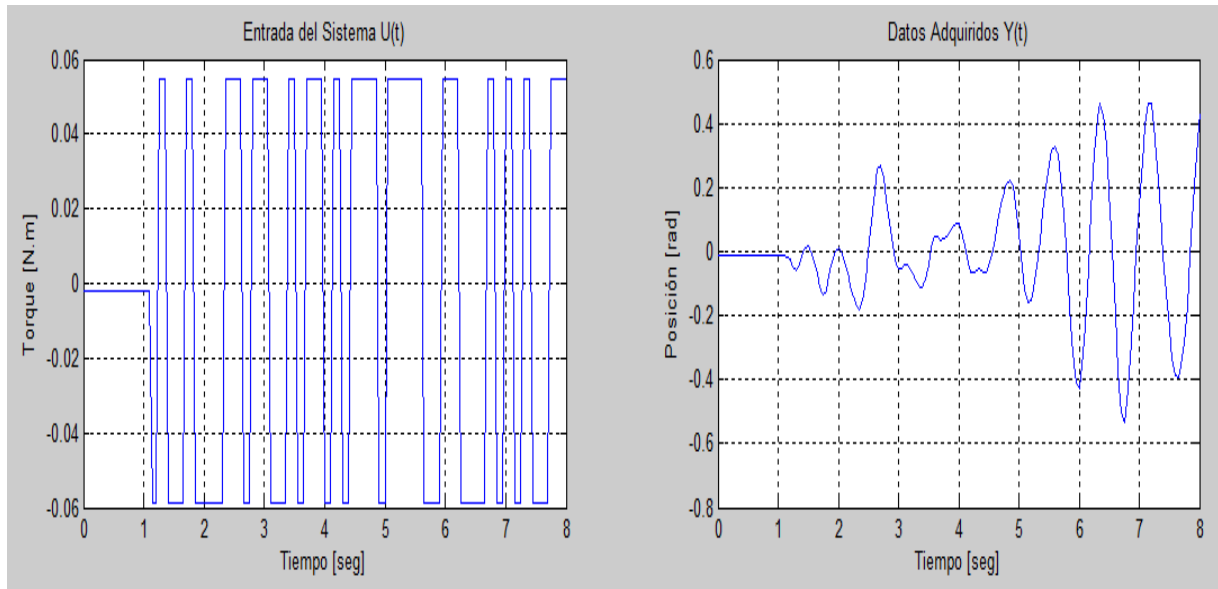


Figura 3.4. Señal de entrada PRBS,  $DA=0,2$  [Nm] ,  $DT=0,15$  [seg] y respuesta del sistema

Salida del comando **arx** del SIT fijando los ordenes  $na=3$ ;  $nb=2$ ;  $nk=1$ .del modelo ARX.

```
Discrete-time IDPOLY model:  $A(q)y(t) = B(q)u(t) + e(t)$ 
 $A(q) = 1 - 2.224 q^{-1} + 1.694 q^{-2} - 0.3877 q^{-3}$ 
 $B(q) = 0.1985 q^{-1} - 0.06994 q^{-2}$ 
Estimated using ARX from data set datos
Loss function 2.08849e-005 and FPE 2.22237e-005
Sampling interval:1
```

### Comparación del las Salidas

a. Utilizando el comando Compare del *System Identification Toolbox*.

```
compare(zv,m1,'r')
```

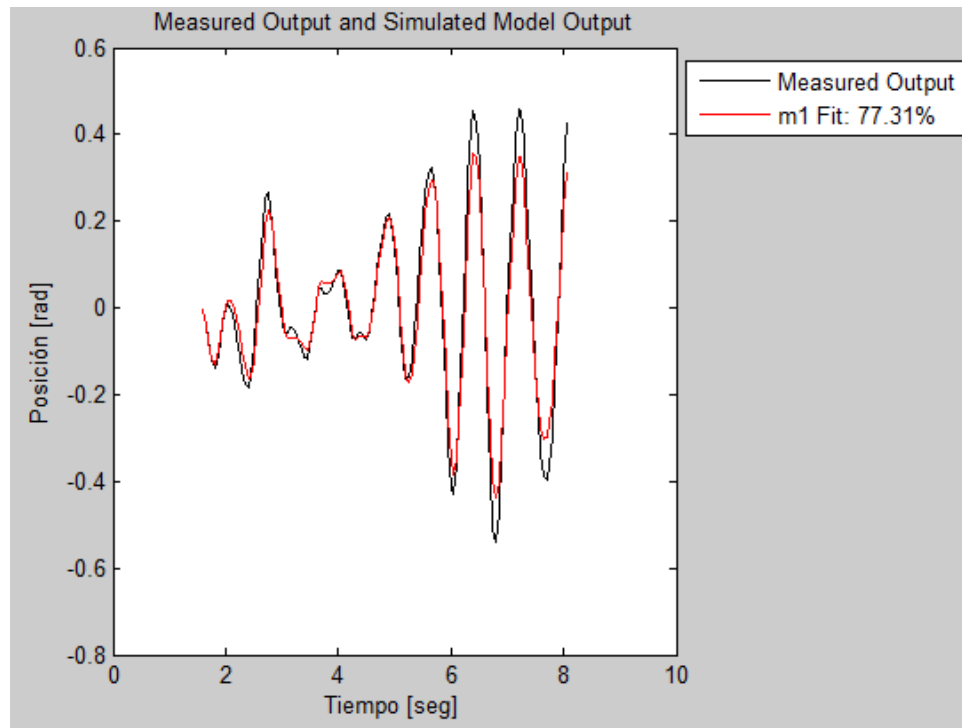


Figura 3.5. Salida real del sistema y la salida estimada con el comando `arx` del SIT.

b. Comparación con el *Script* realizado en Octave (Ver Apéndice A) para estimar los parámetros del Modelo ARX utilizando el método mínimos cuadrados.

Parámetro	Comando ARX de MATLAB	Script en Octave	Porcentaje de Similitud
$a_1$	-2.224	-2.199	98.87%
$a_2$	1.694	1.647	97.22%
$a_3$	-0.3877	-0.3618	93.32%
$b_1$	0.1985	0.2087	95.11%
$b_2$	-0.06994	-0.0780	89.66%

Tabla 3. Parámetros estimados por el comando `arx` del SIT y Script realizado en Octave.

c. Comparación de la salida real y la estimada utilizando el *script* realizado en Octave.

Porcentaje de variación entre la salida real y la estimada = 78.7%

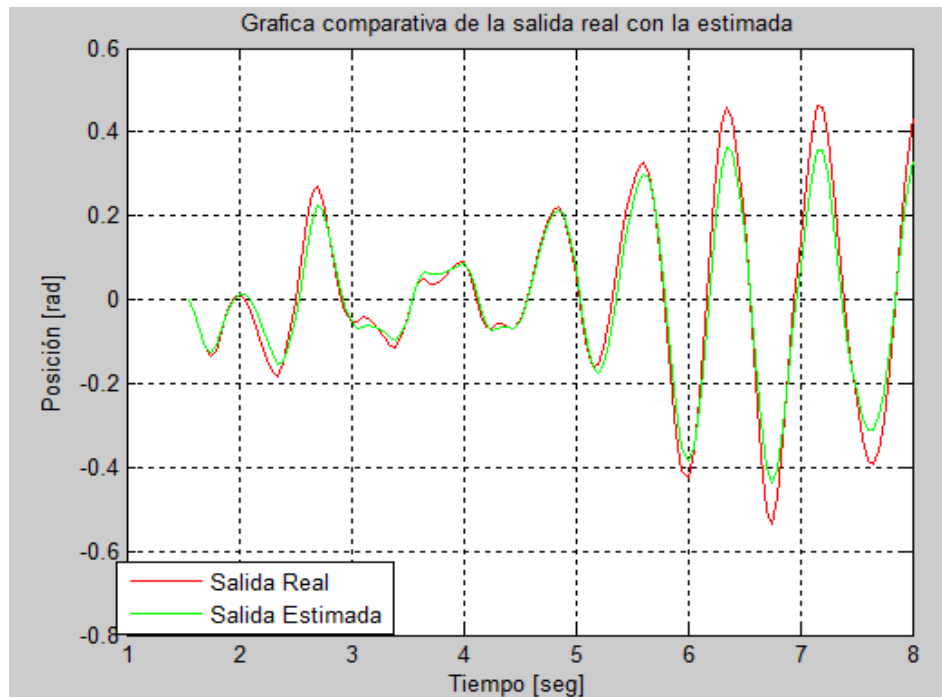


Figura 3.6. Salida real y la Salida Estimada utilizando el Script realizado en Octave.

Con una señal de entrada PRBS a partir de un valor de diferencial amplitud  $DA=0.1$  [Nm] y un diferencial de tiempo  $DT=0,30$  [seg] (1 Cte de tiempo).

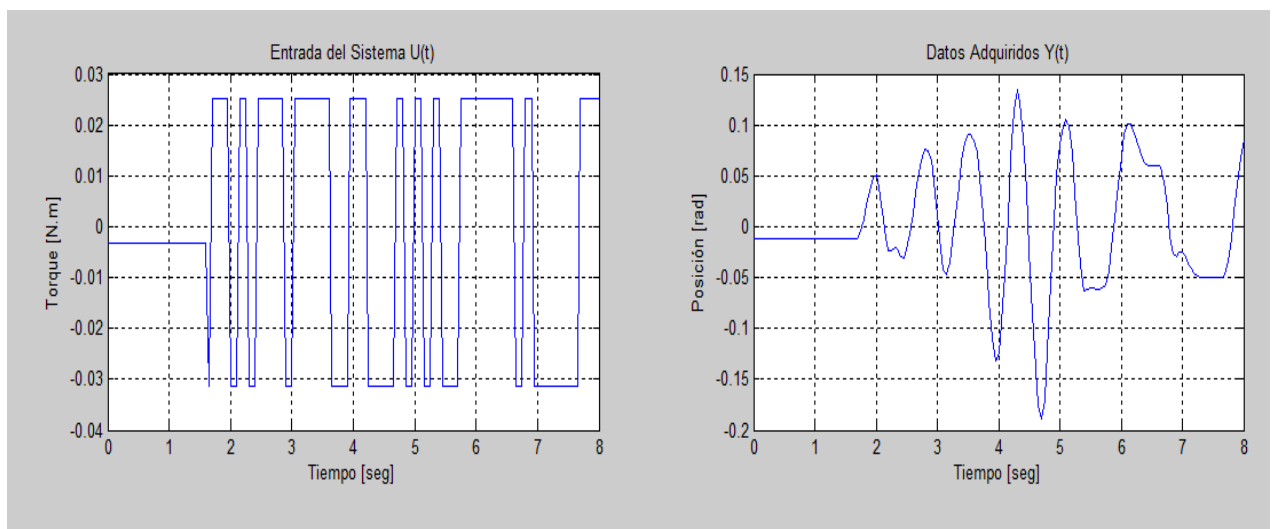


Figura 3.7. Señal de entrada PRBS,  $DA=0,1$  [Nm] y  $DT=0,30$  [seg] y respuesta del sistema.

Salida del comando **arx** del SIT fijando los ordenes  $na=3$ ;  $nb=1$ ;  $nk=1$  del modelo ARX.

```
Discrete-time IDPOLY model: A(q)y(t) = B(q)u(t) + e(t)
A(q) = 1 - 1.939 q^-1 + 1.239 q^-2 - 0.186 q^-3
B(q) = 0.1937 q^-1
Estimated using ARX from data set datos
Loss function 3.82943e-006 and FPE 4.02456e-006
Sampling interval:0.01
```

### Comparación de las Salidas

a. Utilizando el comando Compare del *System Identification Toolbox*.

```
compare(zv,m1,'r')
```

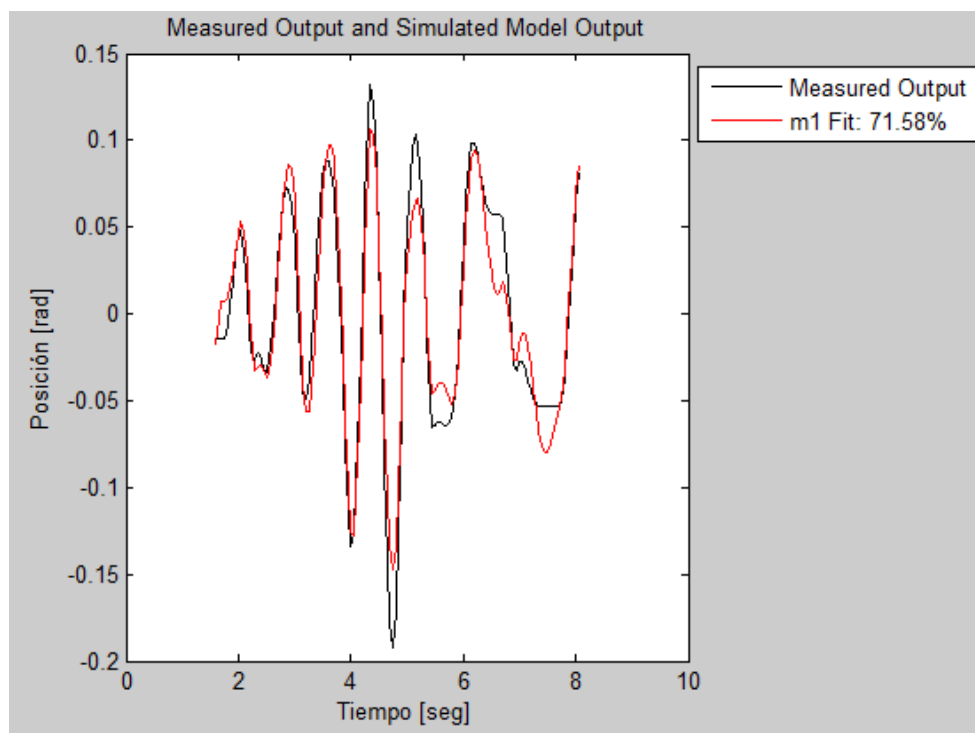


Figura 3.8. Salida real del sistema y la salida estimada con el comando arx del SIT.

**b.** Comparación de la salida real y la estimada utilizando el script realizado en Octave.

Porcentaje de variación entre la salida real y la estimada = 67.96%

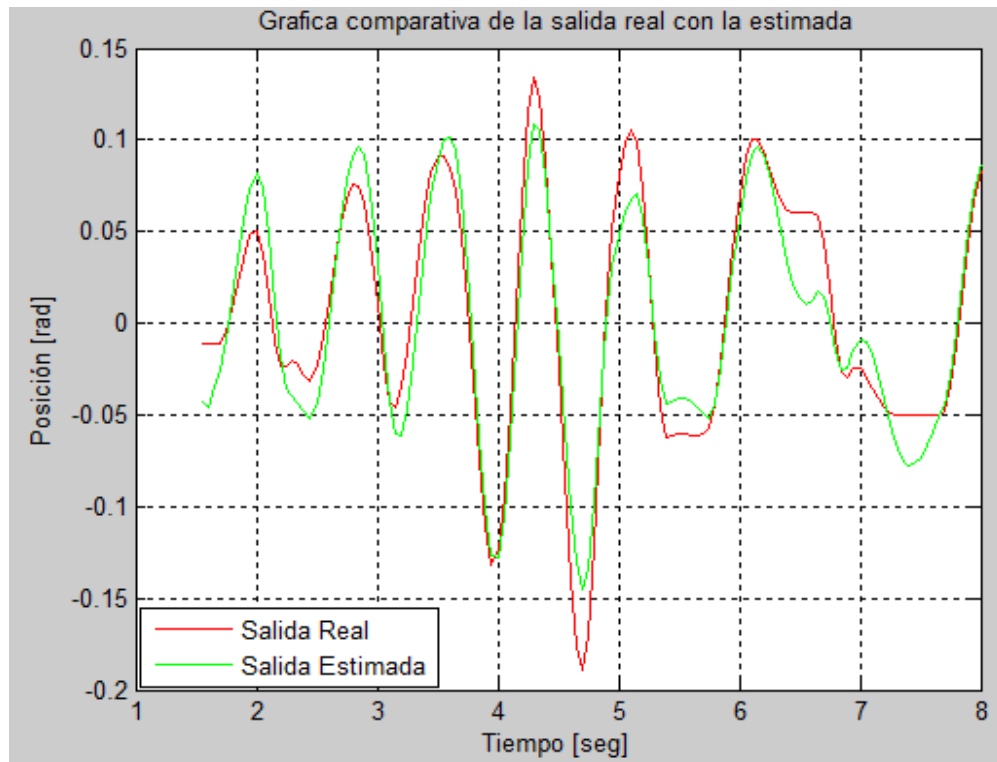


Figura 3.9. Salida real y la Salida Estimada utilizando el Script realizado en Octave.

**c.** Comparación con el Script realizado en Octave (Ver Apéndice A) para estimar los parámetros del Modelo ARX utilizando el método mínimos cuadrados.

Parámetro	Comando ARX de MATLAB	Script en Octave	Porcentaje de Similitud
$a_1$	-1.939	-1.8758	96.74%
$a_2$	1.239	1.1243	90.74%
$a_3$	-0.186	-0.1261	67.80%
$b_1$	0.1937	0.2033	95.28%

Tabla 4. Parámetros estimados por el comando `arx` del SIT y Script realizado en Octave.



Con una señal de entrada PRBS a partir de un diferencial amplitud  $DA=0.2$  [Nm] y un diferencial de tiempo  $DT=0.30$  [seg] (1 Cte de tiempo).

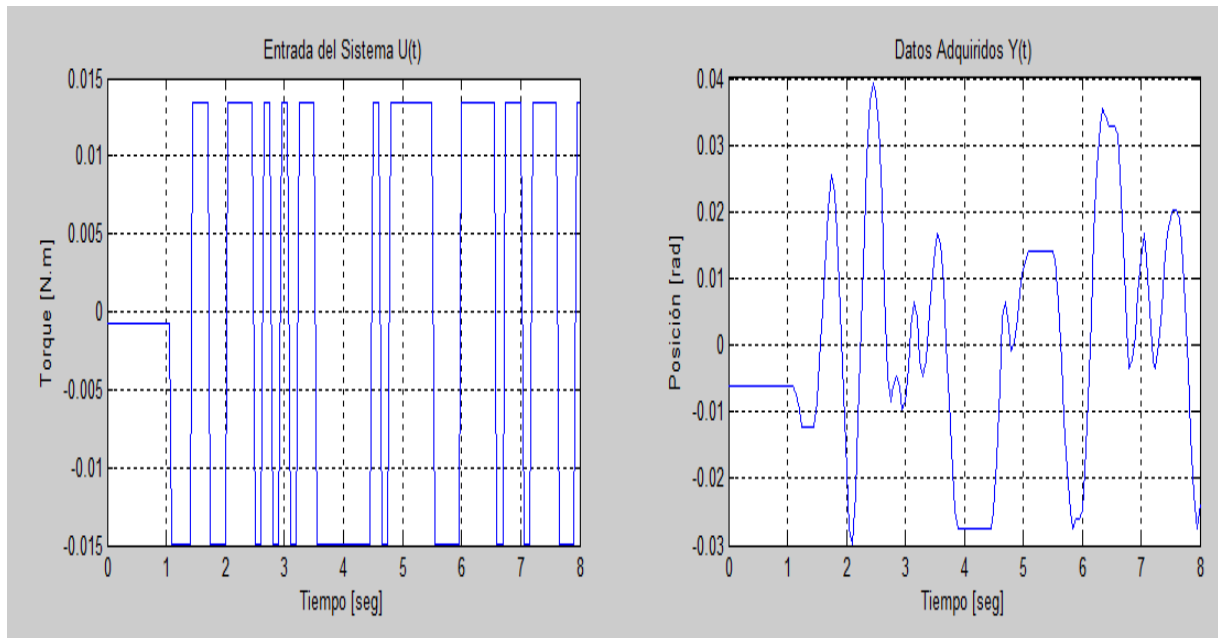


Figura 3.10. Señal de entrada PRBS,  $DA=0.2$  [Nm] ,  $DT=0.30$  [seg] y respuesta del sistema .

Salida del comando **arx** del SIT fijando los ordenes  $na=3$ ;  $nb=1$ ;  $nk=1$ .del modelo ARX.

```
Discrete-time IDPOLY model:  $A(q)y(t) = B(q)u(t) + e(t)$ 
 $A(q) = 1 - 1.809 q^{-1} + 1.128 q^{-2} - 0.2103 q^{-3}$ 
 $B(q) = 0.1624 q^{-1}$ 
Estimated using ARX from data set datos
Loss function 1.31881e-006 and FPE 1.38601e-006
Sampling interval:1
```

### Comparación de las Salidas

a. Utilizando el comando `Compare` del *System Identification Toolbox*.

```
compare(zv,m1,'r')
```

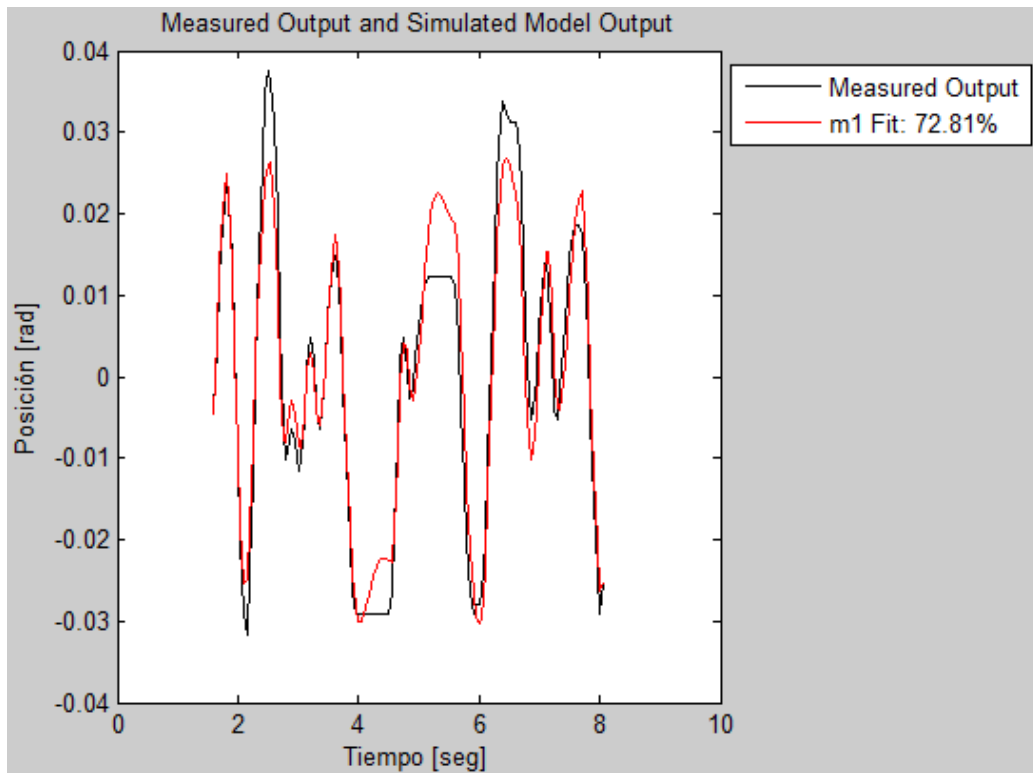


Figura 3.11. Salida real del sistema y la salida estimada con el comando **arx** de SIT.

b. Comparación con el Script realizado en Octave (Ver Apéndice A) para estimar los parámetros del Modelo ARX utilizando el método mínimos cuadrados.

Parámetro	Comando ARX de MATLAB	Script en Octave	Porcentaje de Similitud
$a_1$	-1.809	-1.744	96.41%
$a_2$	1.128	1.018	97.22%
$a_3$	-0.2103	-0.1586	90.25%
$b_1$	0.1624	0.1779	91.29%

Tabla 5. Parámetros estimados por el comando **arx** del SIT y Script realizado en Octave

c. Comparación de la salida real y la estimada utilizando el script realizado en Octave.

Porcentaje de variación entre la salida real y la estimada = 73.62%

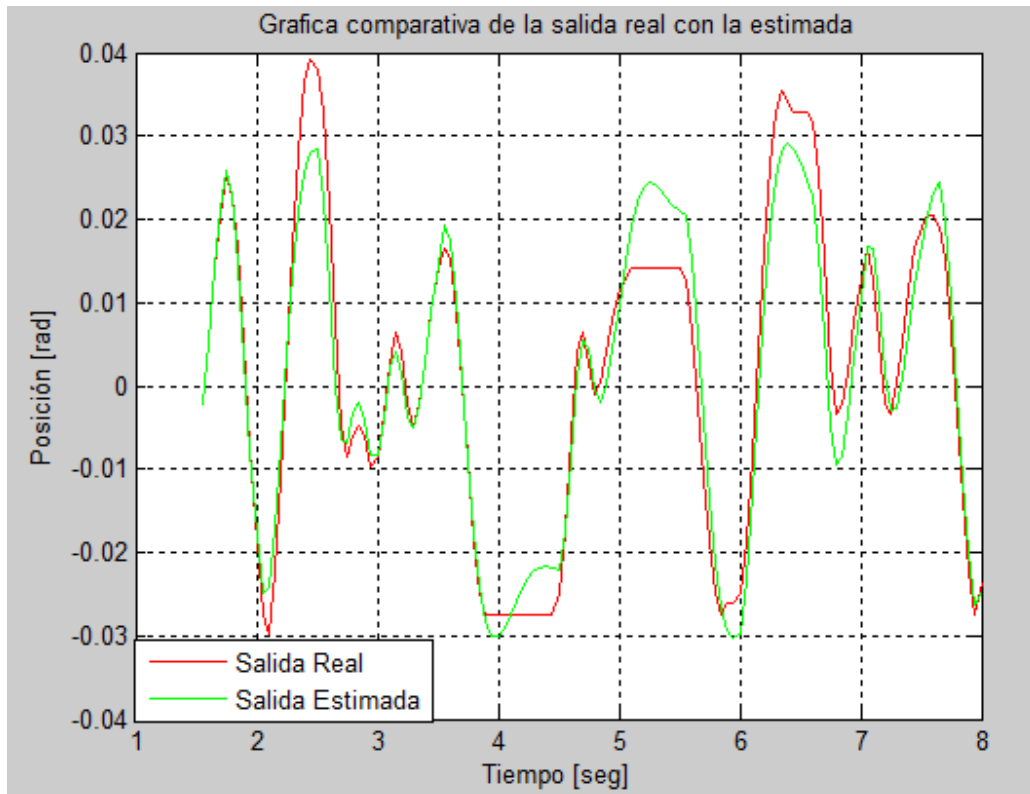


Figura 3.12. Salida real y la Salida Estimada utilizando el Script realizado en Octave.

### 3.7. Experiencias en el punto de operación $q_{1op} = \frac{\pi}{6} [rad]$ , $q_{2op} = 0$

Con una entrada PRBS a partir de un valor de diferencial amplitud DA=0.05 [Nm] y un diferencial de tiempo DT=0,15 [seg] (1/2 Cte de tiempo).

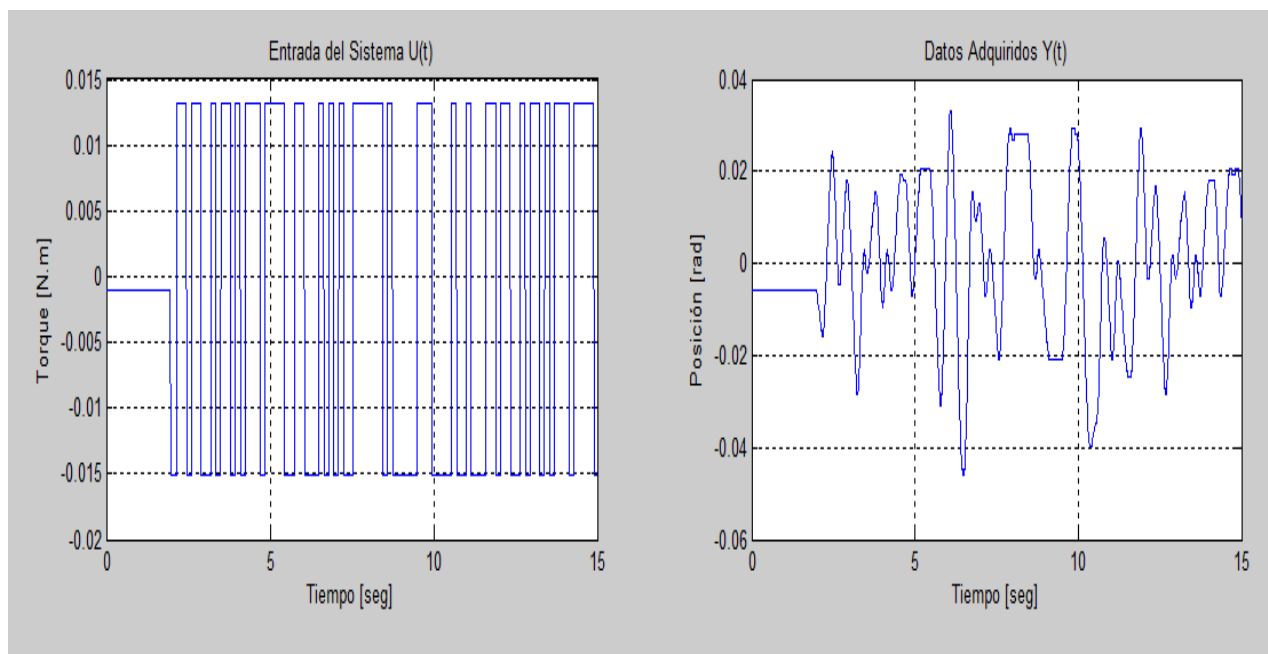


Figura 3.13. Señal de entrada PRBS, DA=0,05 [Nm] , DT=0,15 [seg] y respuesta del sistema .

Salida del comando **arx** del SIT fijando los ordenes  $na=4$ ;  $nb=3$ ;  $nk=2$ .del modelo ARX.

```
Discrete-time IDPOLY model: A(q)y(t) = B(q)u(t) + e(t)
A(q) = 1 - 0.9653 q^-1 - 0.396 q^-2 - 0.04861 q^-3 + 0.4277 q^-4
B(q) = 0.0164 q^-2 + 0.009723 q^-3 + 0.003554 q^-4
Estimated using ARX from data set datos
Loss function 2.83833e-007 and FPE 2.86855e-007
Sampling interval:1
```

### Comparación de las Salidas

a. Utilizando el comando `Compare` del *System Identification Toolbox*.

`compare(zv,m1,'r')`

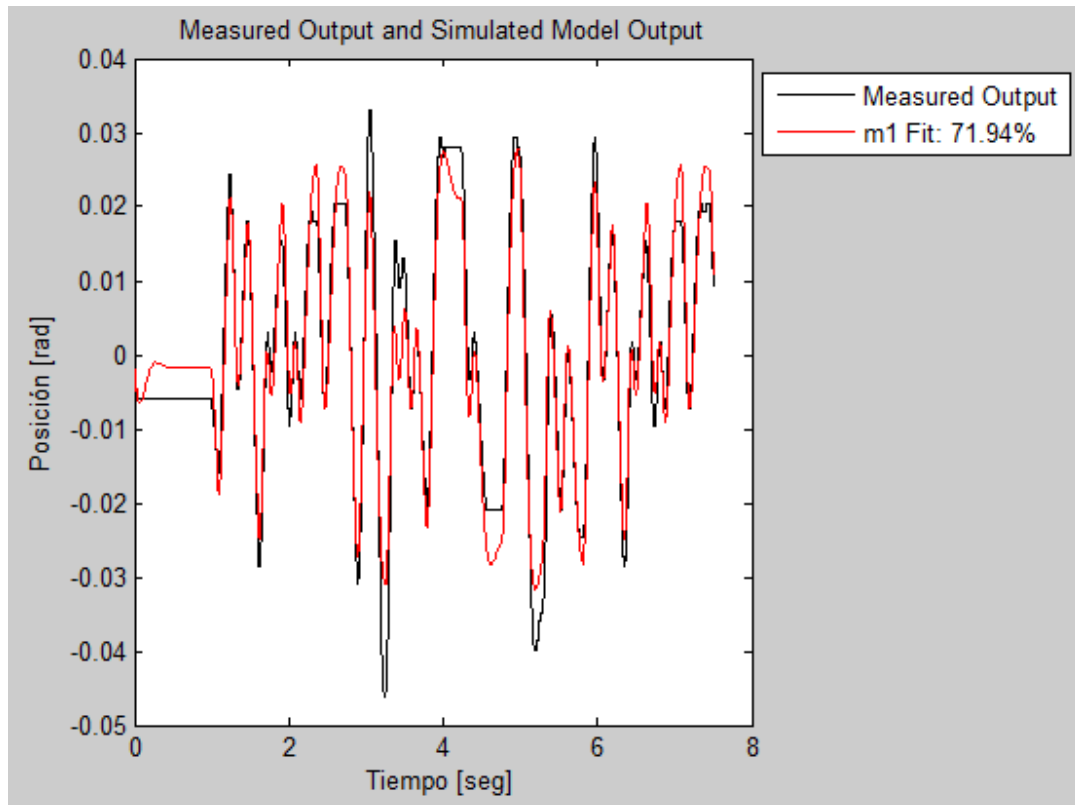


Figura 3.14. Salida real del sistema y la salida estimada con el comando **arx** del SIT

b. Comparación con el Script realizado en Octave (Ver Apéndice A) para estimar los parámetros del Modelo ARX utilizando el método mínimos cuadrados.

Parámetro	Comando ARX de MATLAB	Script en Octave	Porcentaje de Similitud
$a_1$	-0.9653	-0.9876	97.74%
$a_2$	-0.396	-0.364	91.97%
$a_3$	-0.0486	-0.0409	84.16%
$a_4$	0.4277	0.4104	95.95%
$b_1$	0.0164	0.0160	97.56%
$b_2$	0.0097	0.0095	97.39%
$b_3$	0.0036	0.0041	87.80%

Tabla 6. Parámetros estimados por el comando **arx** del SIT y Script realizado en Octave

c. Comparación de la salida real y la estimada utilizando el script realizado en Octave.

Porcentaje de variación entre la salida real y la estimada = 72.02%

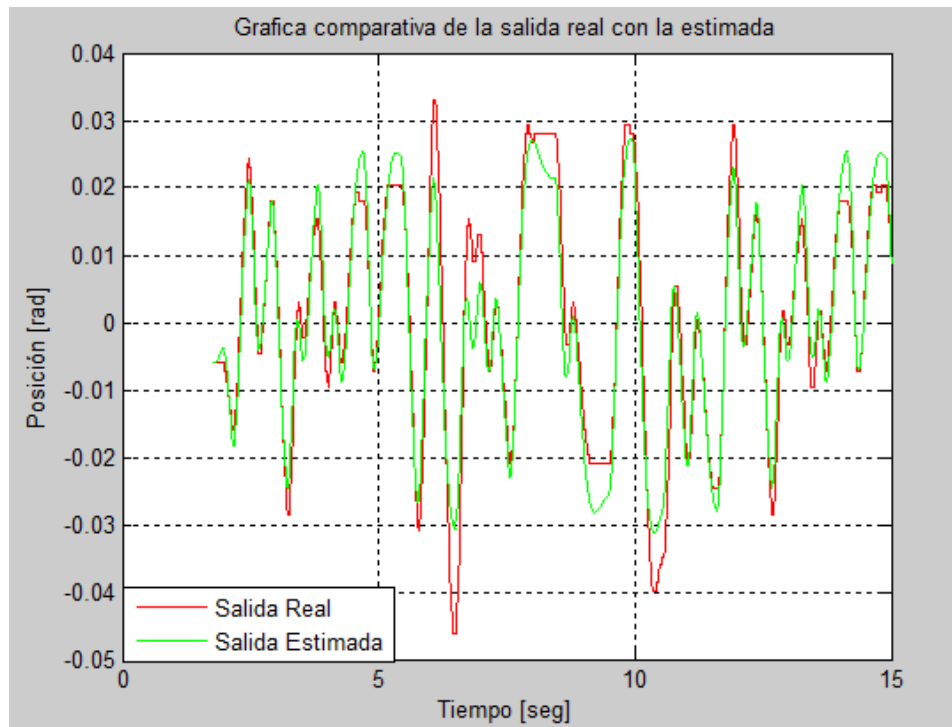


Figura 3.15. Salida real y la Salida Estimada utilizando el Script realizado en Octave.

Con una señal de entrada PRBS a partir de un valor de diferencial amplitud  $DA=0.2$  [Nm] y un diferencial de tiempo  $DT=0.15$  [seg] (1/2 Cte de tiempo).

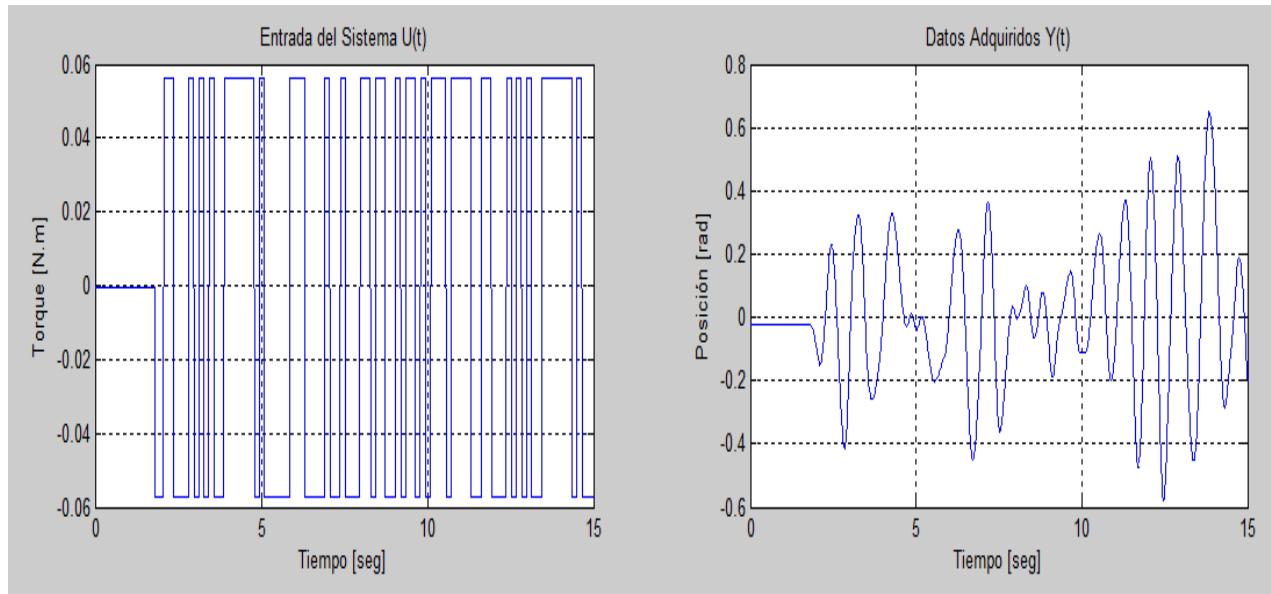


Figura 3.16. Señal de entrada PRBS  $DA=0.2$  [Nm] ,  $DT=0.15$  [seg] y respuesta del sistema

Salida del comando **arx** del SIT fijando los ordenes  $na=3$ ;  $nb=1$ ;  $nk=1$ . del modelo ARX.

```
Discrete-time IDPOLY model:  $A(q)y(t) = B(q)u(t) + e(t)$ 
 $A(q) = 1 - 1.427 q^{-1} - 0.12 q^{-2} + 0.5557 q^{-3}$ 
 $B(q) = 0.01619 q^{-1}$ 
Estimated using ARX from data set pendulo5
Loss function 6.10918e-007 and FPE 6.14626e-007
Sampling interval:1
```

### Comparación de las Salidas

a. Utilizando el comando `Compare` del *System Identification Toolbox*.

`compare(zv,m1,'r')`

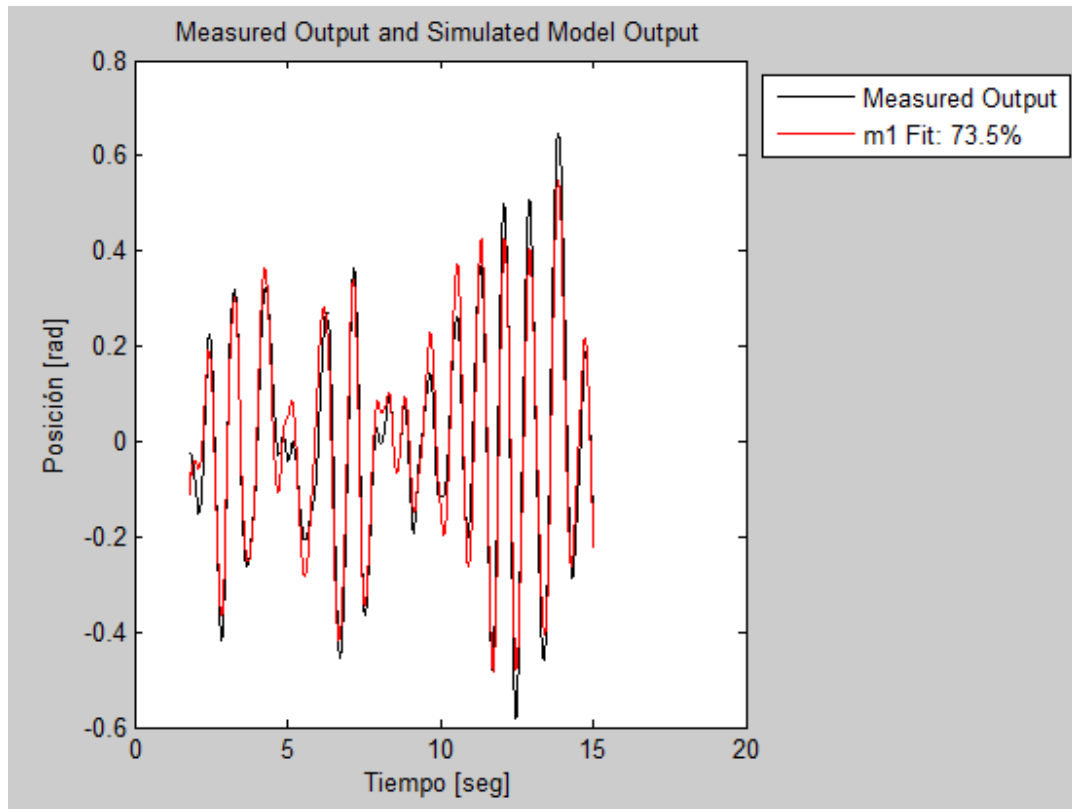


Figura 3.17. Salida real del sistema y la salida estimada con el comando `arx` de SIT.

b. Comparación con el *Script* realizado en Octave (Ver Apéndice) para estimar los parámetros del Modelo ARX utilizando el método mínimos cuadrados.

Parámetro	Comando ARX de MATLAB	Script en Octave	Porcentaje de Similitud
$a_1$	- 1.427	-1.450	98.41%
$a_2$	- 0.12	-0.07	58.33%
$a_3$	0.5557	0.5318	95.69%
$b_1$	0.0162	0.0168	96.43%

Tabla 7. Parámetros estimados por el comando `arx` del SIT y *Script* realizado en Octave



c. Comparación de la salida real y la estimada utilizando el *script* realizado en Octave.

Porcentaje de variación entre la salida real y la estimada = 71.98%

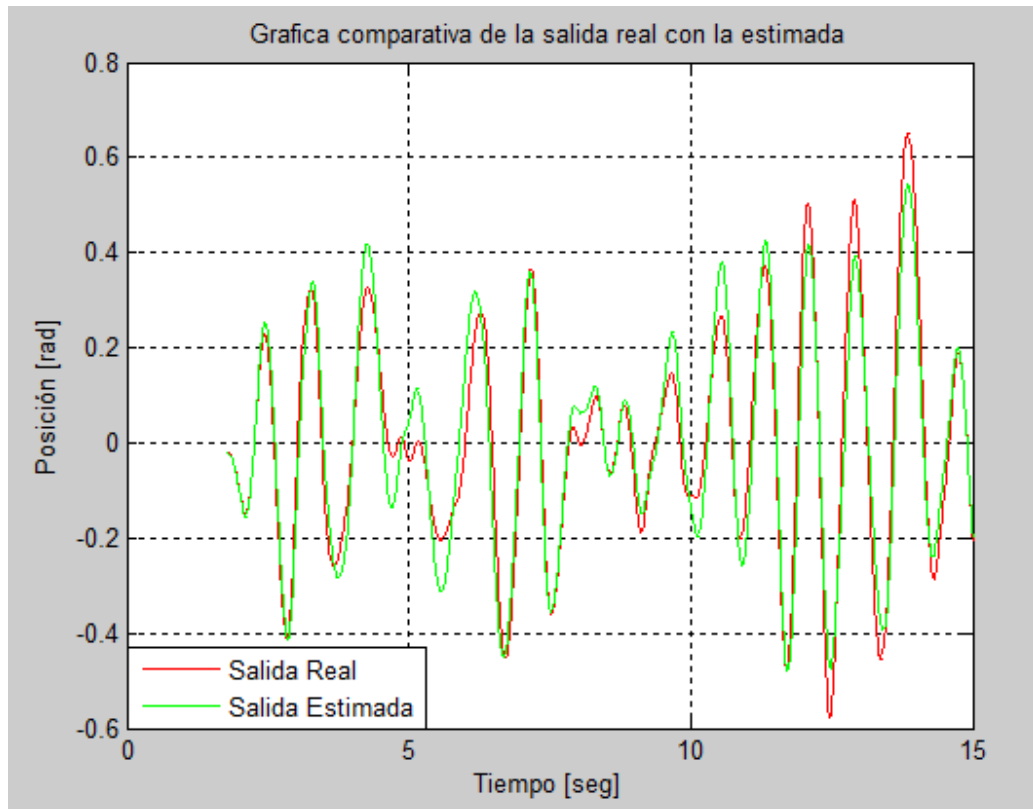


Figura 3.18. Salida real y la Salida Estimada utilizando el Script realizado en Octave.

Con una señal de entrada PRBS a partir de un valor de diferencial amplitud  $DA=0.1$  [Nm] y un diferencial de tiempo  $DT=0,30$  [seg] (1 Cte de tiempo).

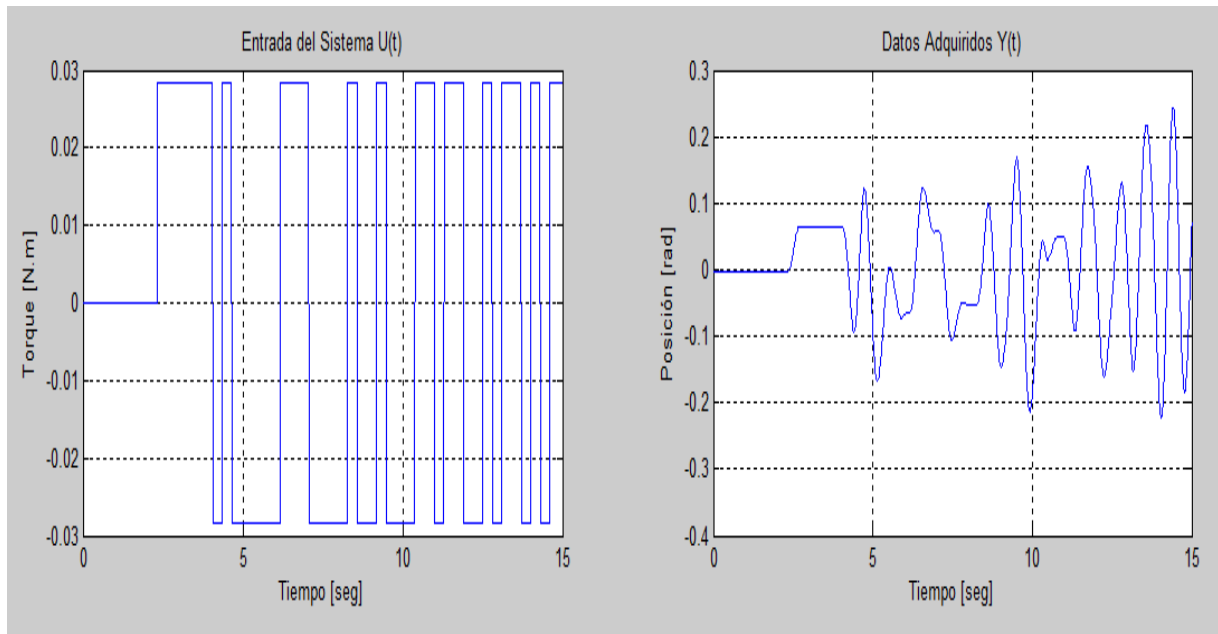


Figura 3.19. Señal de entrada PRBS  $DA=0,1$  [Nm] ,  $DT=0,30$  [seg] y respuesta del sistema .

Salida del comando **arx** del SIT fijando los ordenes  $na=3$ ;  $nb=2$ ;  $nk=1$ .del modelo ARX.

```
Discrete-time IDPOLY model:  $A(q)y(t) = B(q)u(t) + e(t)$ 
 $A(q) = 1 - 1.334 q^{-1} - 0.2839 q^{-2} + 0.627 q^{-3}$ 
 $B(q) = 0.003394 q^{-1} + 0.01558 q^{-2}$ 
Estimated using ARX from data set datos
Loss function 3.4775e-007 and FPE 3.5039e-007
Sampling interval:1
```

### Comparación de las Salidas

a. Utilizando el comando `Compare` del *System Identification Toolbox*.

`compare(zv,m1,'r')`

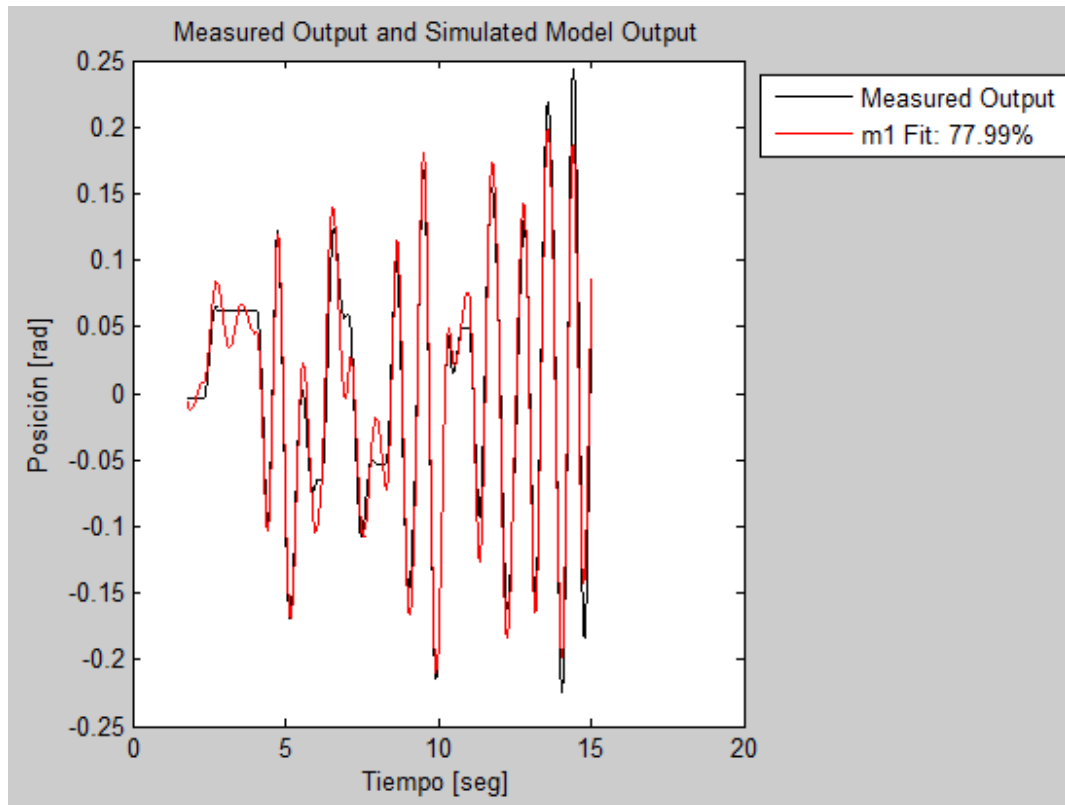


Figura 3.20. Salida real del sistema y la salida estimada con el comando `arx` de SIT.

b. Comparación con el *Script* realizado en Octave (Ver Apéndice A) para estimar los parámetros del Modelo ARX utilizando el método mínimos cuadrados.

Parámetro	Comando ARX de MATLAB	Script en Octave	Porcentaje de Similitud
$a_1$	- 1.334	-1.338	99.70%
$a_2$	- 0.2839	-0.2754	97.00%
$a_3$	0.627	0.623	99.36%
$b_1$	0.00334	0.0034	98,24%
$b_2$	0.0156	0.0156	100%

Tabla 8. Parámetros estimados por el comando `arx` del SIT y Script realizado en Octave

c. Comparación de la salida real y la estimada utilizando el script realizado en Octave.

Porcentaje de variación entre la salida real y la estimada = 77.65%

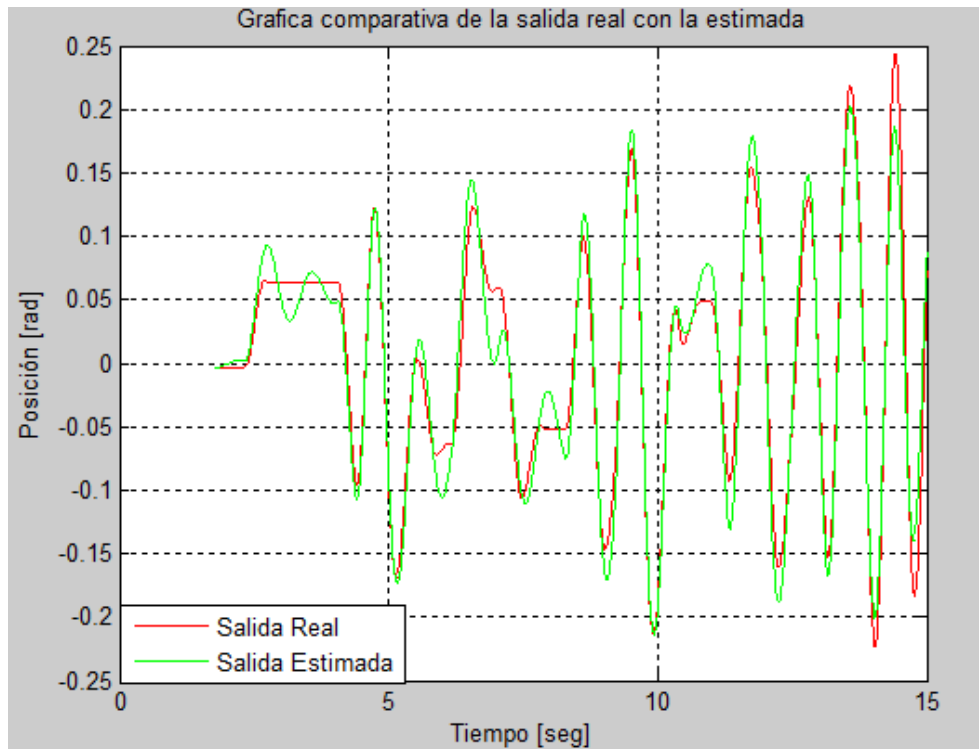


Figura 3.21. Salida real y la Salida Estimada utilizando el Script realizado en Octave.

**Con una señal de entrada PRBS a partir de un valor de diferencial amplitud  $DA=0.2$  [Nm] y un diferencial de tiempo  $DT=0,30$  [seg] (1 Cte de tiempo).**

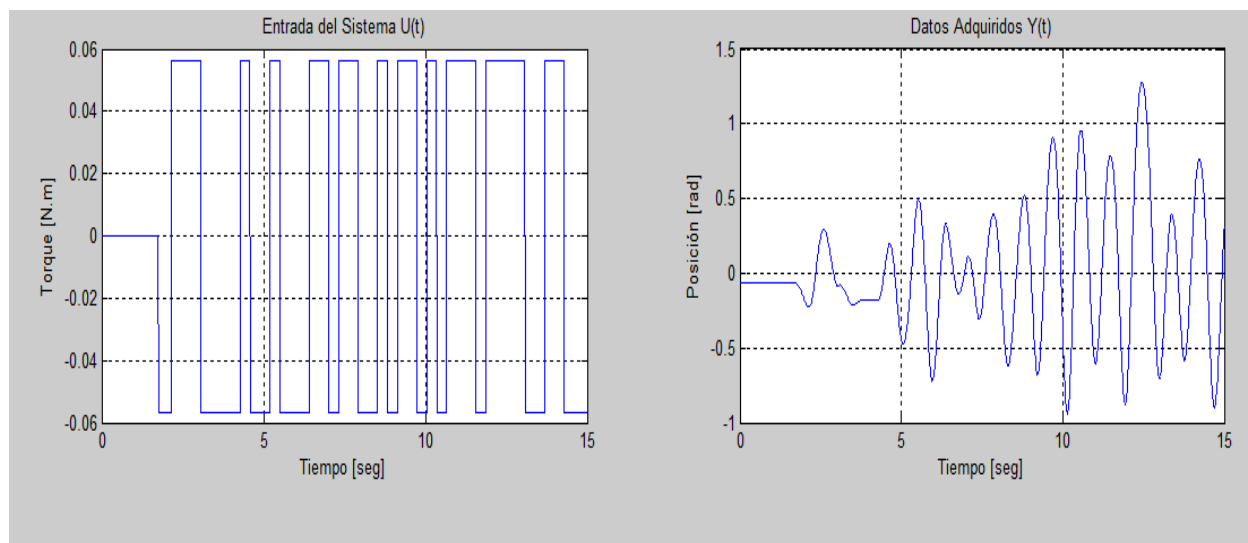


Figura 3.22. Señal de entrada PRBS  $DA=0,2$  [Nm] ,  $DT=0,30$  [seg] y respuesta del sistema

Salida del comando **arx** del SIT fijando los ordenes  $na=3$ ;  $nb=2$ ;  $nk=1$  del modelo ARX.

```
Discrete-time IDPOLY model: A(q)y(t) = B(q)u(t) + e(t)
A(q) = 1 - 1.99 q^-1 + 0.9949 q^-2
B(q) = 0.009968 q^-1
Estimated using ARX from data set pendulo5
Loss function 1.16393e-006 and FPE 1.16923e-006
Sampling interval:1
```

### Comparacion de las Salidas

a. Utilizando el comando **Compare** del *System Identification Toolbox*.

```
compare(zv,m1,'r')
```

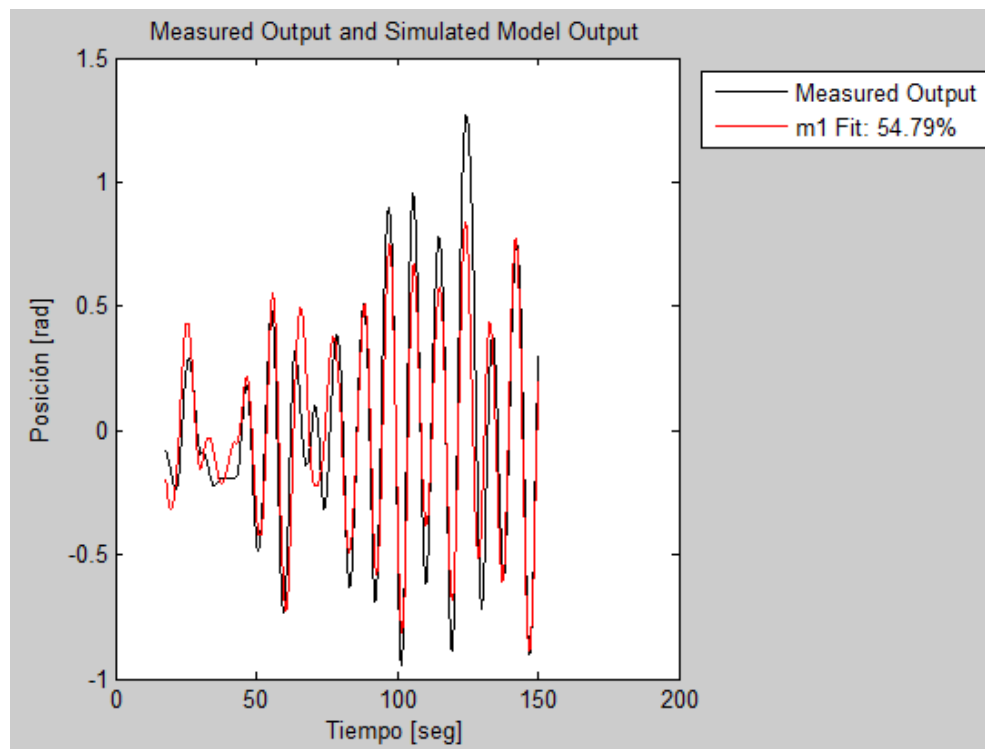


Figura 3.22. Salida real del sistema y la salida estimada con el comando **arx** de SIT.

b. Comparación con el *Script* realizado en Octave (Ver Apéndice A) para estimar los parámetros del Modelo ARX utilizando el método mínimos cuadrados.

Parámetro	Comando ARX de MATLAB	Script en Octave	Porcentaje de Similitud
$a_1$	-1.99	-1.99	100%
$a_2$	0.9949	0.9894	99.45%
$b_1$	0.0099	0.0125	79.2%

Tabla 9. Parámetros estimados por el comando **arx** del SIT y Script realizado en Octave

c. Comparación de la salida real y la estimada utilizando el *script* realizado en Octave.

Porcentaje de variación entre la salida real y la estimada = 50.2%



Figura 3.23. Salida real y la Salida Estimada utilizando el Script realizado en Octave

**Análisis de Resultados**

Con el fin de validar el modelo identificado por el script realizado en Octave, se comparara los diferentes ordenes ante una misma señal de entrada, se utilizara el sistema excitado con una señal de entrada PRBS de amplitud 0.05 [Nm] y un diferencial de amplitud 0,15 [seg], alrededor del origen.

<i>na</i>	<i>nb</i>	<i>nk</i>	Ajuste	Función de Pérdida (FP)	Criterio de Akaike (AIC)
2	1	1	60.14%	3.9762e-005	4.35
<b>2</b>	<b>2</b>	<b>1</b>	<b>76.16%</b>	<b>1.4672e-005</b>	<b>3.38</b>
3	1	1	72.34%	1.9749e-005	4.67
3	2	1	73.04%	1.9046e-005	4.68
3	3	1	73.03%	1.9333e-005	4.66
4	1	1	66.16%	3.0005e-005	4.49
4	2	1	67.21%	2.8595e-005	4.49
4	3	1	66.81%	2.9749e-005	4.47
4	4	1	65.79%	3.2081e-005	4.43

Esta tabla anterior muestra que el modelo optimo es  $na=2$ ,  $nb= 2$ , y  $nk=1$ ; ya que el ajuste es mayor, la función de perdida es menor y el criterio de Akaike es menor. Se dejo fijo el valor de  $nk$  igual a 1 debido a que experimentalmente [8] se obtuvo que el sistema no presenta retardo.

## CAPITULO 4

### INTERFAZ WEB CON PHP Y OCTAVE

#### 4. Interfaz Web con Php y Octave

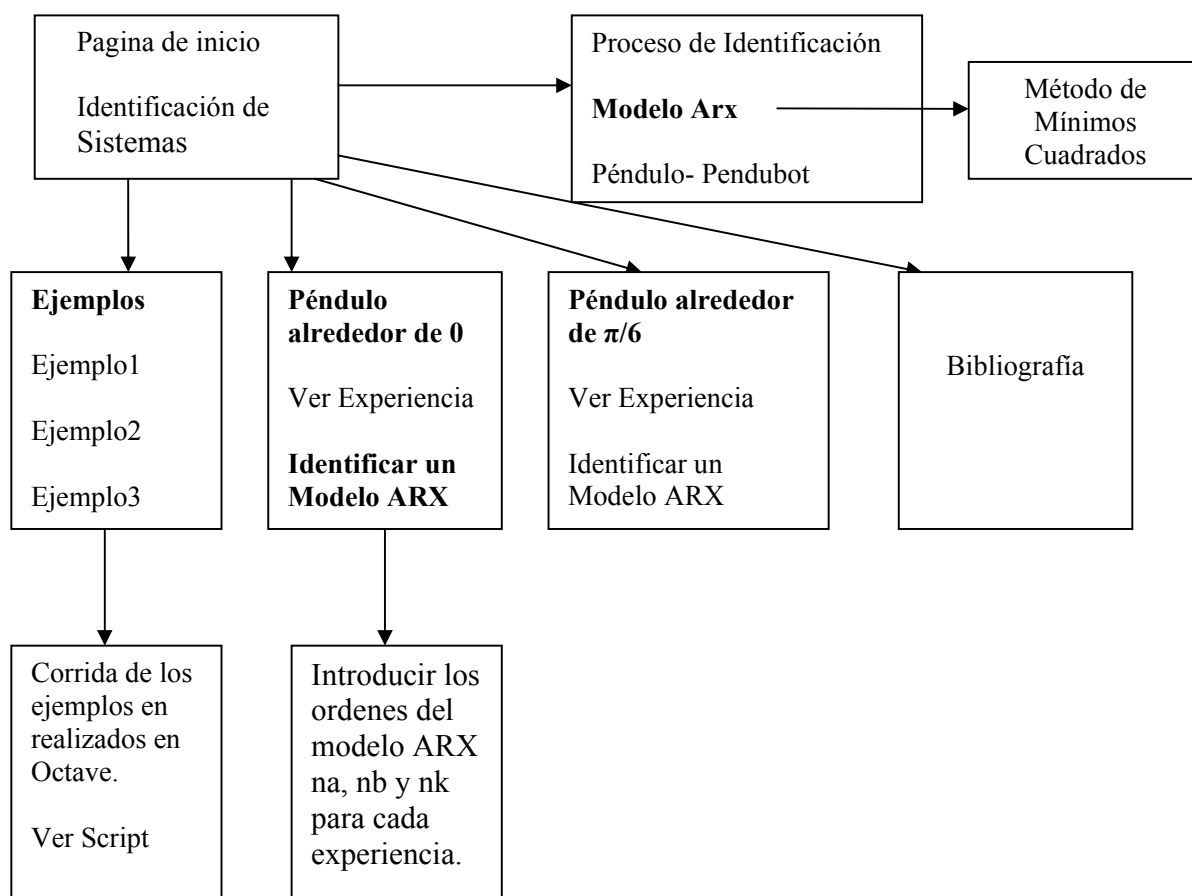


Diagrama de la Interfaz Web realizada en Php.



En este capítulo se presenta la interfaz realizada en PHP, como primer punto una introducción a la identificación de sistemas, el proceso de identificación de sistemas, se conocerá el método de estimación de parámetros mediante el uso de mínimos cuadrados, la estructura de modelos lineales discretos y el caso particular de modelos ARX. Existe un enlace que permite ver el vídeo de la captura de datos del Pendubot y se describirá el diseño del mismo.

Esta interfaz también permite a los usuarios descargar y ejecutar los scripts desarrollados en Octave. Estos programas aplican el método de mínimos cuadrados para la estimación de parámetros de los modelos. Permite que el usuario pueda identificar el péndulo ubicado en el laboratorio de control, usando una estructura de modelo ARX, así como también se puede ver cada una de las experiencias realizadas con el péndulo alrededor de  $q = (0,0)$  y  $q = (\frac{\pi}{6}, 0)$ , con distintas señales de entrada de tipo binaria pseudo aleatoria (PRBS).

El sistema permite la identificación del péndulo físico a distancia, de modo que cualquier estudiante puede acceder a ello por medio de Internet aprovechando la interacción entre PHP y Octave. Dicha interacción permite el diálogo entre el ordenador y el motor de cálculo de Octave instalado en el servidor, convirtiendo así cualquier computador con conexión a Internet en un potencial laboratorio. Dicho sistema es capaz de mostrar los resultados e imágenes de la estimación de parámetros de un modelo ARX. Una característica importante es que cuenta con un formulario, es decir una plantilla o página con espacios vacíos que admite introducir el valor de orden del polinomio  $A(q)$  ( $na$ ), orden del polinomio  $B(q)$  ( $nb$  y  $nk$ ), que es el retardo de la entrada  $u(t)$  con respecto a la salida  $y(t)$ , realizando así la estimación de cada experiencia según lo que el usuario desee.

En esta sección se describirá la funcionalidad del laboratorio virtual, presentando directamente ejemplos de la aplicación. Cada ejemplo describe los *scripts* que se encargarán de las estimaciones y que son utilizados para desarrollar la plataforma tecnológica, para lo cual se usan dos tipos de *script*, uno con extensión .php de PHP y el otro con extensión .m de Octave. Para realizar una explicación detallada de los *scripts* se tomará como referencia solo una de las experiencias, ya que para el resto de las experiencias se sigue el mismo procedimiento, es

decir se utiliza la misma plantilla de éste con mínimas modificaciones. La única modificación es que se cargará un archivo de datos (de entrada y salida del péndulo) diferente para cada caso de estudio.

#### 4.1. Página de Inicio

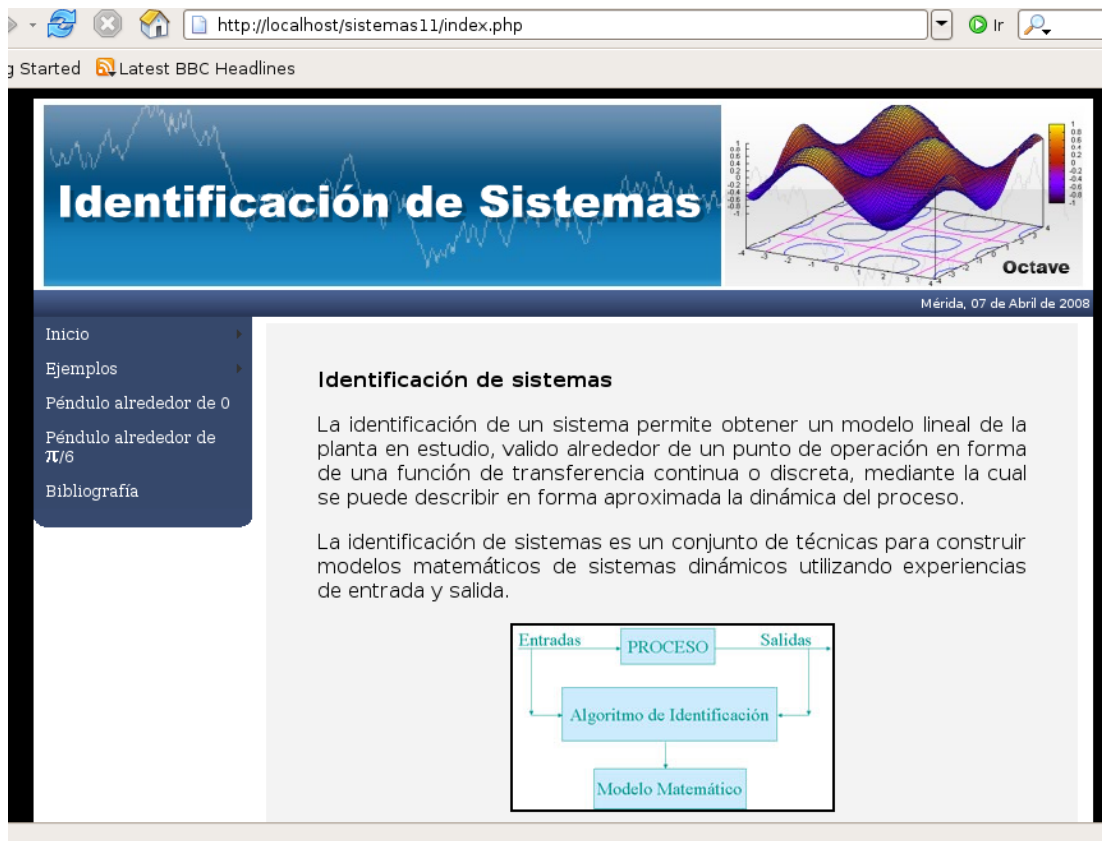


Figura 4.1. Página de Inicio de la Interfaz vía Web.

La página de inicio permite leer una breve introducción sobre identificación de sistemas. Seguidamente se muestra el *script* realizado en PHP y HTML y posteriormente se hace una explicación del mismo.

##### Script 4.1: inicio.php

```
/*Programa para realización de la página de inicio de identificación
de sistemas utilizando Octave
Autor: Br. María Elena Noriega
Tutor: Prof. Pablo Lischinsky
Realizado el 25/03/2008 */
<?php include('fecha.php');
```

```

?>
<html>
<head>
<title>Principal</title>
<link href="estilo.css" rel="stylesheet" type="text/css" />
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<style type="text/css">
<!--
body {
    background-color: #000000;
}
.Estilo1 {font-family: Verdana, Arial, Helvetica, sans-serif}
.Estilo2 {font-family: Verdana, Arial, Helvetica, sans-serif; font-
weight: bold; }
.Estilo3 {color: #0000CC}
--></style>
</head>
<body>
<!--Etiquetas table conforman la tabla principal del sitio -->
<table width="813" height="100%" border="0" align="center"
cellpadding="00" cellspacing="00" bgcolor="#FFFFFF">
    <!-- Etiquetas tr y td representan las columnas y las filas-->
    <tr><td height="150" colspan="3" valign="bottom">
        <div align="center">
            <?php include('banner.html'); ?>
        </div> </td>
    </tr><tr>
        <td height="21" colspan="3" background="images/barraS.jpg">
            <div align="right" class="fechaB">
                <?= show_date();
                ?> </div>
            </td> </tr>
    <tr>
        <td width="20%" valign="top">
            <table width="171" border="0" align="center" cellpadding="0"
cellspacing="0">
                <tr>
                    <td width="171" height="19" valign="top">
                        <?php include("menu.html");?> </td>
                    </tr><tr>
                        <td height="17" valign="top"></td> </tr>
            </table>
        </td> <td width="80%" colspan="2" valign="middle">
            <table width="98%" height="100%" border="0" align="center"
cellpadding="0" cellspacing="5" >
                <tr>
                    <td height="225" valign="top" bgcolor="#F3F3F3"><blockquote>
                        <br> <p class="Estilo2">Identificación de sistemas</p>

```

```

        <p align="justify" class="Estilo1">La identificación de un
sistema permite obtener un modelo lineal de la planta en estudio,
valido alrededor de un punto de operación en forma de una función de
transferencia continua o discreta, mediante la cual se puede
describir en forma aproximada la dinámica del proceso.</p>
        <p align="justify" class="Estilo1">La identificación de
sistemas es un conjunto de técnicas para construir modelos
matemáticos de sistemas dinámicos utilizando experiencias de entrada
y salida.<br>
        </p>
    </blockquote><p align="center" class="Estilo1"><br>
        <br>
        Esquema General de Identificación de Sistemas
    </p>
    <p align="right" class="Estilo1"><a href="proceso.php"
class="Estilo3"><u>Proceso de identificación</u></a></p></td>
</tr>
</table></td></tr> <tr>
<td height="43" colspan="3" background="images/barraI.jpg" >
<div align="center">
    <?php include('pie.html');
    ?>
</div></td></tr>
</table>
</body>
</html>

```

### Explicación del Script inicio.php

El código PHP es un lenguaje de programación interpretado usado normalmente para la creación de páginas web dinámicas. PHP es un acrónimo recursivo que significa **PHP Hypertext Pre-processor** (inicialmente *PHP Tools*, o, *Personal Home Page Tools*) se combina con la sigla HTML de **HyperText Markup Language** (Lenguaje de Etiquetas de Hipertexto), es el lenguaje predominante para la construcción de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes.

<?php Indica el principio de PHP

include('fecha.php'); incluye el archivo fecha.php que contiene la función de la fecha actual.

?> cierra el bloque de PHP.

<html> Define el inicio del documento HTML.

<head> Define la cabecera del documento HTML.

`<title>Principal</title>` Define el título de la página.

Etiquetas style CSS las cuales contiene el estilo de la página: color de fondo, de letra, tipo.. etc.

```
<style type="text/css">
<table width="813" height="100%" border="0" align="center"
cellpadding="00" cellspacing="00" bgcolor="#FFFFFF"> Etiqueta table,
Conforma la tabla principal del sitio.
```

`<?php include('banner.html');` Se inicia PHP y se incluye el archivo banner.html que contiene el banner hecho en flash.

`<?php include("menu.html");` Se abre PHP y carga el archivo menu.html que contiene el menu del sitio.

`<?php include('pie.html');` Carga el archivo pie.php que contiene el pie de pagina.

Después de conocer un poco sobre identificación de sistemas, hay un enlace que permite al usuario ver el proceso de identificación, mostrando un esquema del mismo.

Esta página se hace siguiendo los mismos lineamientos explicados anteriormente.

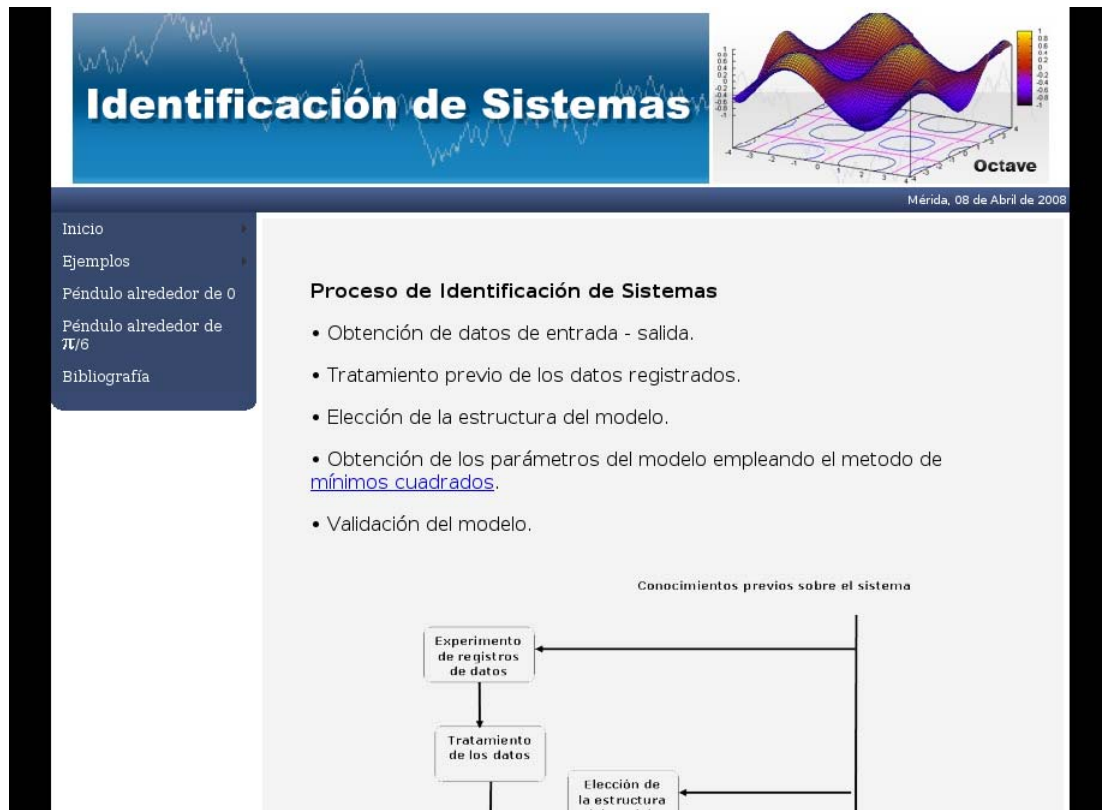


Figura 4.2. Enlace al proceso de identificación de sistemas.

En esta página hay un vínculo hacia el método de mínimos cuadrados, el cual explica resumida y detalladamente el principio del mismo. El enlace se hace de la siguiente manera:

`<a href="minimo.php" class="Estilo3 Estilo5">` Crea el enlace o hipervínculo a `minimo.php` el cual contiene la página que explica el método de mínimos cuadrados (Ver Figura 4.3).

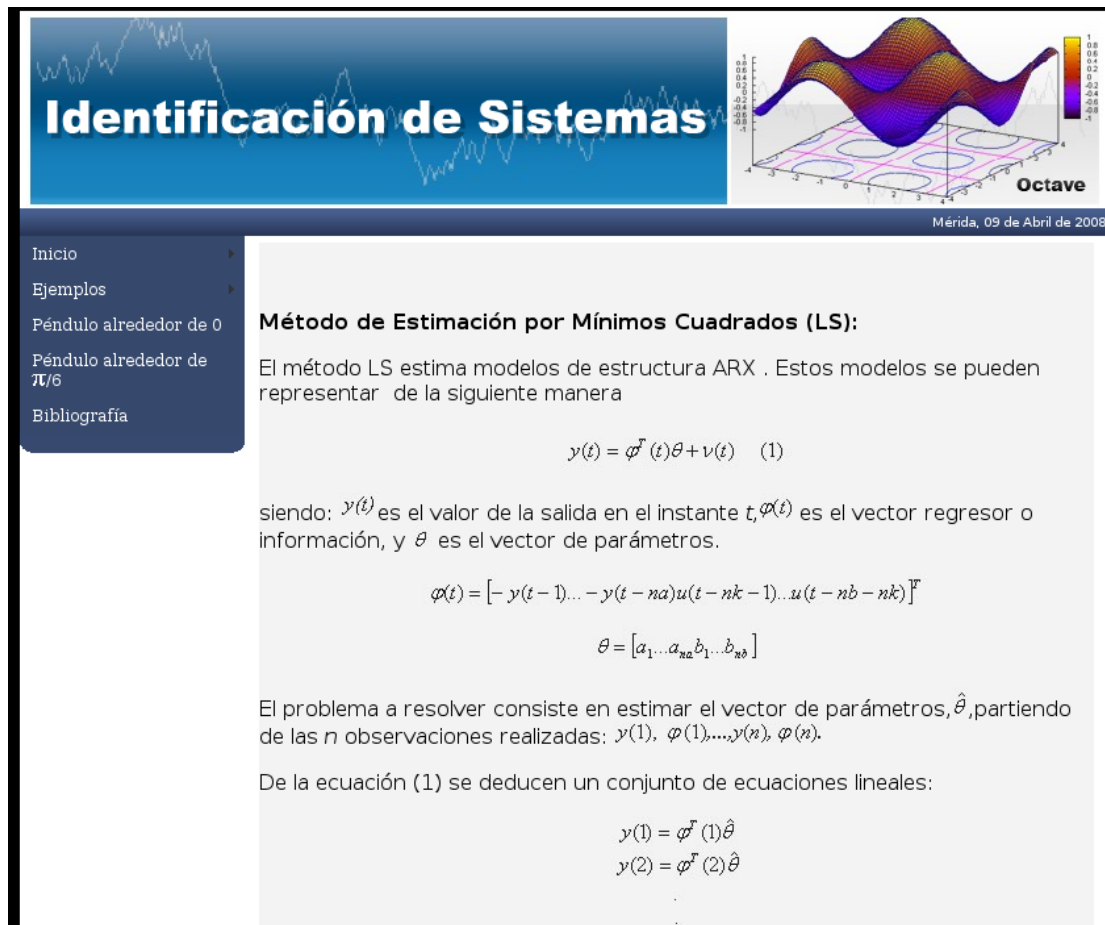


Figura 4.3. Página del método de mínimos cuadrados.

Posteriormente se accede, mediante el menú, a una de las estructuras de modelos lineales discretos el modelo ARX, para que el usuario se familiarice con esta estructura que será utilizada en el proceso de identificación del péndulo (Ver Figura 4.4).

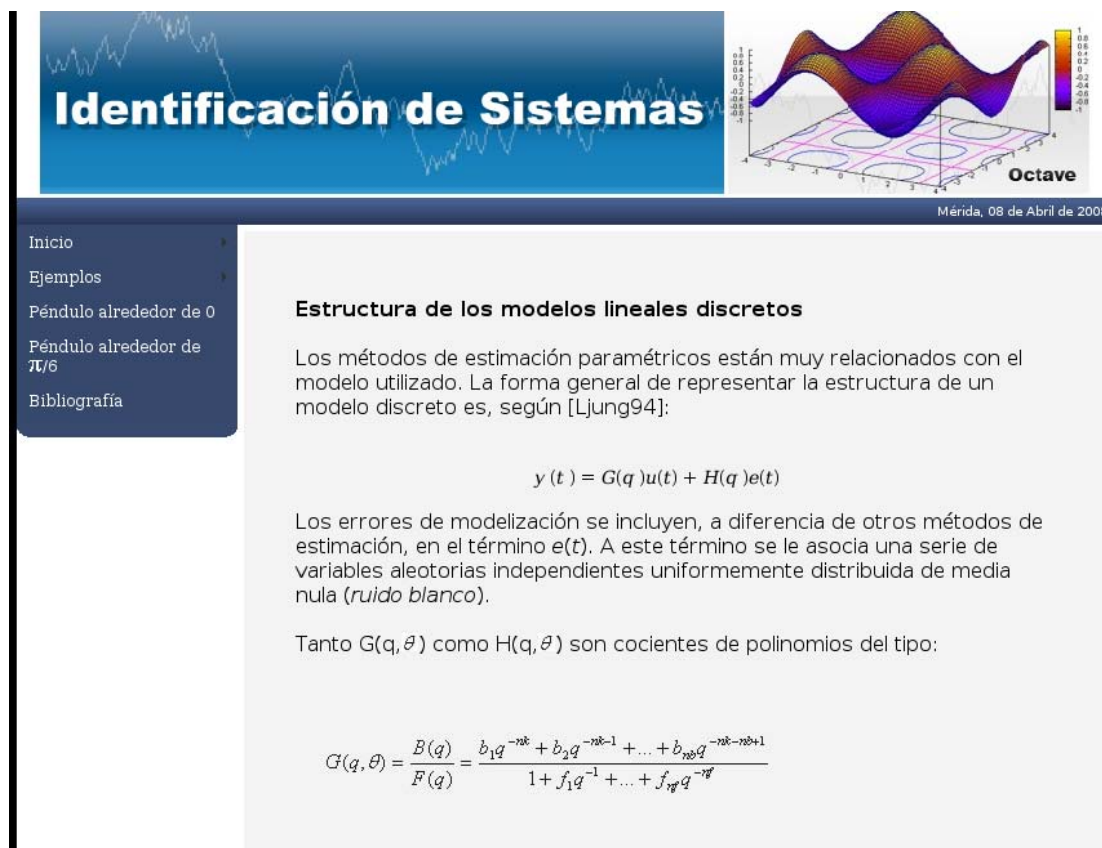


Figura 4.4. Interfaz gráfica de la estructura de los modelos lineales discretos.

Luego se accede a una descripción del péndulo y una foto del Pendubot ubicado en el Laboratorio de Sistemas de Control. El Pendubot es usado para realizar experiencias de entrada/salida y a partir de estos datos hacer la identificación vía Internet. Se puede ver desde aquí un video que muestra el proceso de captura de datos del Pendubot (Ver Figura 4.5). El vídeo es descargado en la siguiente dirección <http://es.youtube.com/watch?v=lvOI64HuYxQ>.





Figura 4.5. Interfaz gráfica del péndulo y Pendubot.

El vídeo se muestra con el siguiente comando:

```
<a href="#" class="Estilo3" onClick= "ventana('video.html')"> Enlace a
video.html
```

Script 4.2. video.html

```
<table width="200" border="0" cellpadding="0" cellspacing="0">
  <tr>
    <td><object width="499" height="434">
      <paramname="movie"
value="http://www.youtube.com/v/lvOI64HuYxQ&hl&fmt=18=es">
      <param name="wmode" value="transparent">
```

```

        <embed src="http://www.youtube.com/v/lvOI64HuYxQ&hl&fmt=18=es"
type="application/x-shockwave-flash" wmode="transparent" width="499"
height="434"></embed>
    </object></td>
</tr> <tr>
    <td align="center" style="font-family:Verdana;"><strong>Obtencion
de datos del pendubot</strong></td>
</tr>
</table>

```

El comando `table` crea una tabla en el cual se muestra el vídeo, la etiqueta `<embed>` es la que se utiliza para insertar archivos de vídeo. A través del atributo `src` hay que especificar la ruta y el nombre del archivo de vídeo.

`<paramname="movie"` La forma más común de insertar un archivo de Flash.

La etiqueta `<param>` no necesita etiqueta de cierre, y ha de contener los atributos `name` y `value`. El atributo `name` indica el nombre de la característica que va a ser definida, y `value` indica su valor. Las animaciones Flash se reproducen de forma automática al cargarse la página y su reproducción es continua.

La siguiente opción del menú accede a los ejemplos. Se muestran tres típicos ejemplos de estimación de parámetros utilizando el método de mínimos cuadrados. Esta aplicación ejecuta el *script* elaborado en Octave y retorna el resultado, que en este caso es una imagen de tipo png. También se puede descargar el *script*.

**Ejemplo 1:** consiste en la estimación de parámetros de cuatro modelos

$$m1: \hat{y} = b_0$$

$$m2: \hat{y} = b_0 + b_1 u$$

$$m3: \hat{y} = b_0 + b_1 u + b_2 u^2$$

$$m4: \hat{y} = b_0 + b_1 u + b_2 u^2 + b_3 u^3$$

Se estima la curva de regresión que mejor se ajusta a los datos, exponiendo que el modelo que mejor se ajusta es el modelo 4, modelo de 3° orden, ya que la función de pérdida  $V(\hat{\theta})$  es menor con respecto a los otros modelos (Ver Figura 4.6).

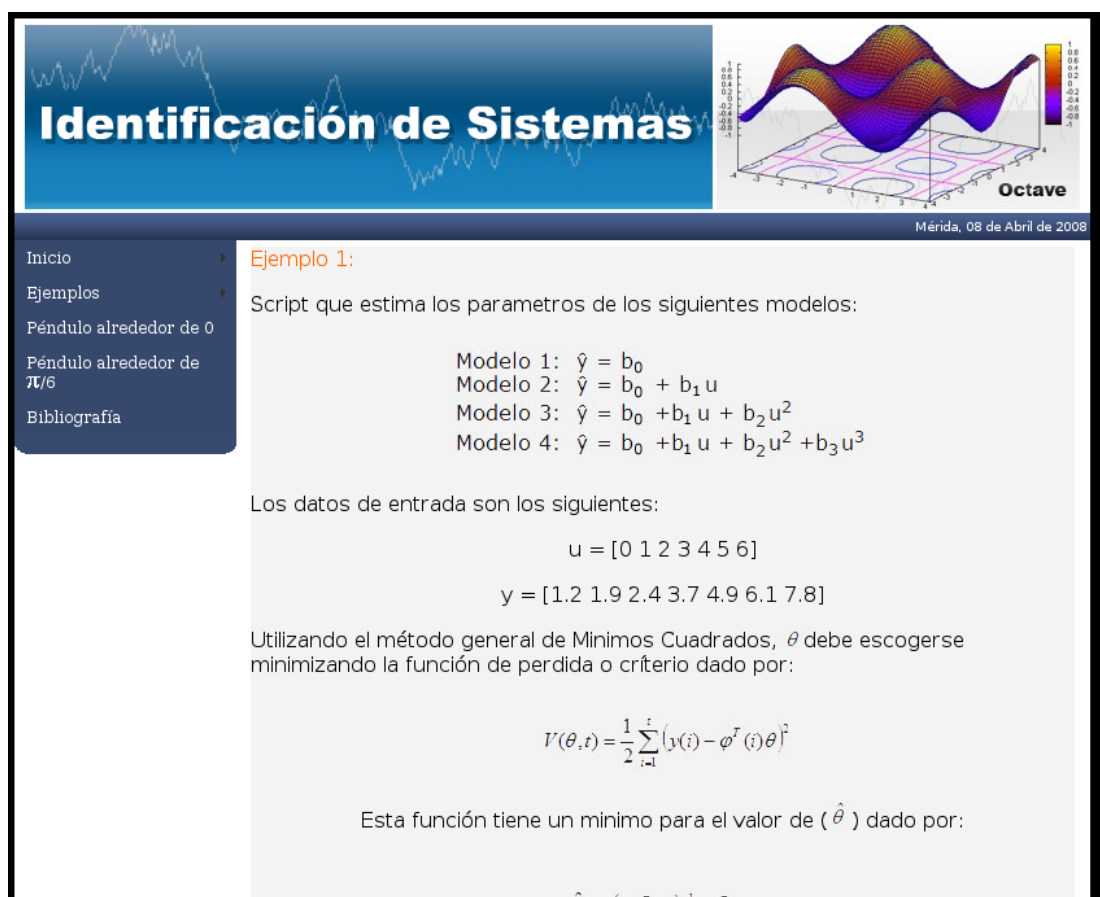


Figura 4.6. Interfaz gráfica del ejemplo 1.

## Script 4.3: ejemplo1.php

```
<?php
/*Programa que realiza la ejecución de GNU Octave a través de la
función passthru y retorna el resultado, en este caso una imagen de
tipo png.
Autor: Br. Maria Elena Noriega R.
Tutor: Prof. Pablo Lischinsky.
Realizado el 01/03/2008.*/
header("Content-type: image/png");/*Función que retorna una imagen de tipo
png.*/
passthru('/usr/bin/octave --silent ejercicio1.m');/*Función que ejecuta a
Octave.*/
?>
```

El script ejemplo1.php es usado para generar la imagen (en este caso) o para mostrar a través de una ventana lo que ha realizado Octave.

La línea `header("Content-type: image/png")` sirve para indicar que en la cabecera http se va devolver una imagen de tipo PNG. Para ello se utiliza `image/png`.

La función `passthru()` de PHP es la función mas importante ya que se lleva a cabo la ejecución de Octave, esta función ejecuta un programa externo en este caso Octave y muestra la salida para este ejemplo, es una imagen del tipo PNG donde se observa la gráfica comparativa de los modelos aproximados por mínimos cuadrados.

La función `passthru()` permite la ejecución del comando `--silent`, que a su vez realiza la ejecución del archivo de Octave llamado `ejercicio1.m`. Ver apéndice para ver *script*.

Para que la gráfica resultante del *script* realizado en Octave pueda ejecutarse es indispensable colocar el siguiente comando en cada uno de los scripts elaborados en Octave:

```
__gnuplot_set__terminal png
```

Gnuplot soporta muchos recursos para diferentes gráficos. Gnuplot usa `set__terminal` para generar la salida de un gráfico, en este caso la salida es una imagen de tipo PNG (Ver Figura 4.7).

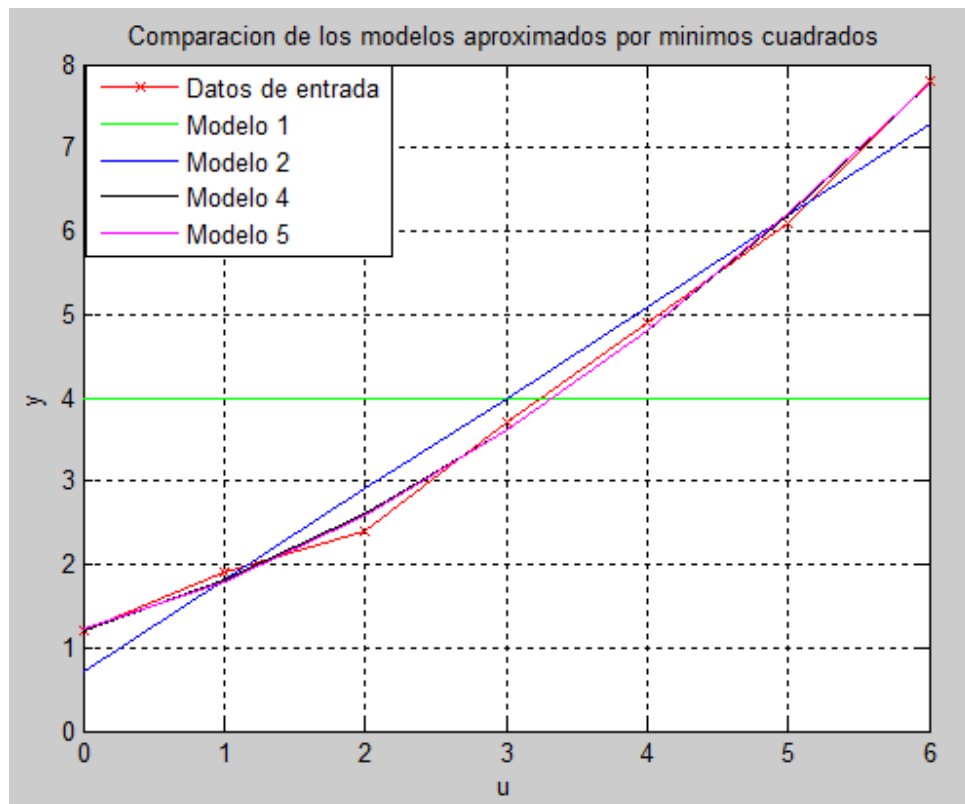


Figura 4.7. Imagen resultante del ejemplo1 realizado en Octave.

Parámetros estimados y función de pérdida  $V(\theta)$ , para cada modelo:

Modelo 1  $b_0: 4$ ,  $V(\theta): 17.28$

Modelo 2  $b_0: 0.7107$ ,  $b_1: 1.096$ ,  $V(\theta): 0.4498$

Modelo 3  $b_0: 1.2048$ ,  $b_1: 0.5036$ ,  $b_2: 0.0988$ ,  $V(\theta): 0.0398$

Modelo 4  $b_0: 1.2214$ ,  $b_1: 0.4480$ ,  $b_2: 0.1238$ ,  $b_3: -0.0028$ ,  $V(\theta): 0.0389$

Se observa que el modelo que mejor se ajusta a los datos es el modelo 4, ya que presenta el menor valor en la función de pérdida.

**Ejemplo 2:** Se hace la identificación del Pendubot, estas experiencias entrada/salida se utiliza para identificar el siguiente modelo (Ver Figura 4.8):

$$y(t) + a_1 y(t-1) + a_2 y(t-2) = b_1 u(t-1) + b_2 u(t-2) + e(t) \quad (4.1)$$

➤ Datos Conocidos:

$$\{u(1), y(1), u(2), y(2), \dots, u(n), y(n)\}$$

➤ Parámetros desconocidos:

$$\theta = \begin{bmatrix} a_1 \\ a_2 \\ b_1 \\ b_2 \end{bmatrix}$$

Se realiza la estimación de dos maneras a través de ecuación normal del método de mínimos cuadrados (ecuación 2.4.8) y mediante el gradiente es decir a partir de:

$$V(\theta) = \sum_{t=1}^N E^2(t) = \sum_{t=1}^N [y(t) + a_1 y(t-1) + a_2 y(t-2) - b_1 u(t-1) - b_2 u(t-2)]^2 \quad (4.3)$$

Desarrollando la ecuación (4.3) y realizando las derivadas parciales con respecto a cada uno de los parámetros  $\theta = (a_1 \ a_2 \ b_1 \ b_2)$ . Se obtiene un sistema de ecuaciones de la forma:

$$\underbrace{\begin{bmatrix} y(t-1)^2 & y(t-1)y(t-2) & -y(t-1)u(t-1) & -y(t-1)u(t-2) \\ y(t-1)y(t-2) & y(t-2)^2 & -y(t-2)u(t-1) & -y(t-2)u(t-2) \\ -y(t-1)u(t-1) & -y(t-2)u(t-1) & u(t-1)^2 & u(t-1)u(t-2) \\ -y(t-1)u(t-2) & -y(t-2)u(t-2) & u(t-1)u(t-2) & u(t-2)^2 \end{bmatrix}}_A \underbrace{\begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \hat{b}_1 \\ \hat{b}_2 \end{bmatrix}}_{\hat{\theta}} = \underbrace{\begin{bmatrix} -y(t)y(t-1) \\ -y(t)y(t-2) \\ y(t)u(t-1) \\ y(t)u(t-2) \end{bmatrix}}_B$$

De esta manera se estima el vector de parámetro  $\hat{\theta}$  :

$$A\hat{\theta} = B \Rightarrow \hat{\theta} = A \setminus B$$

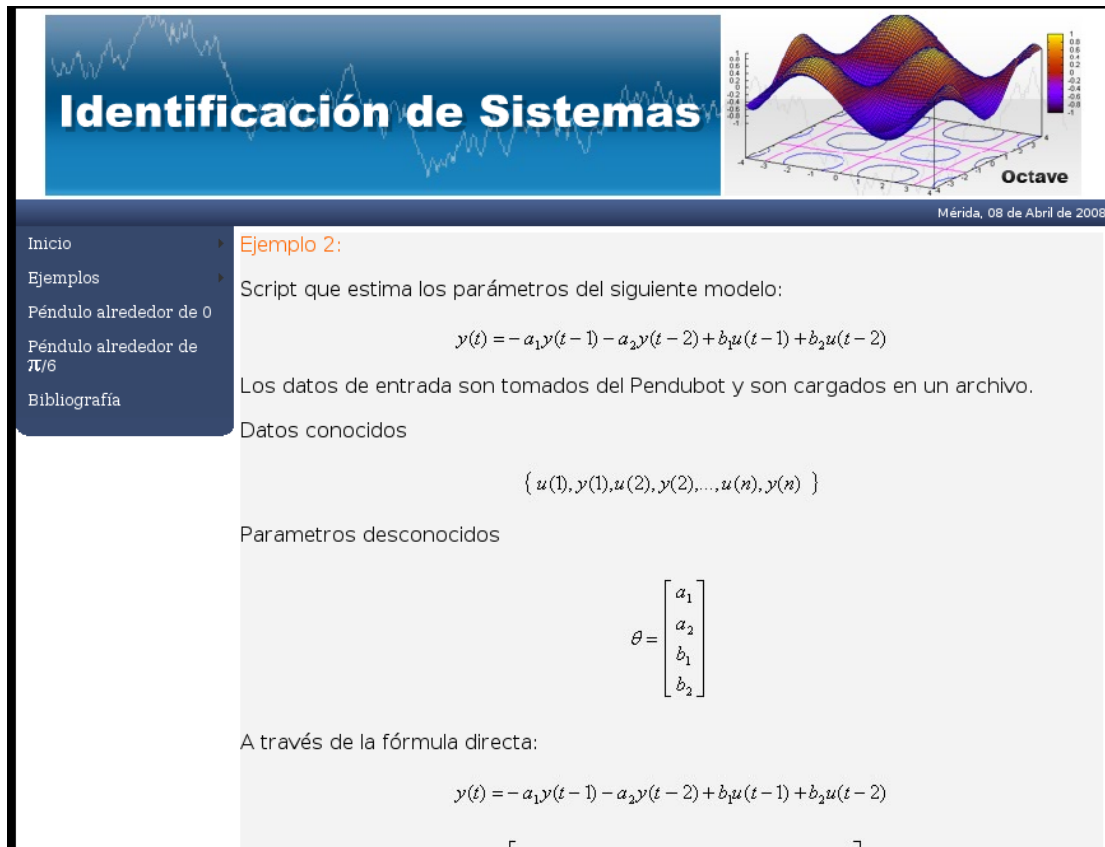


Figura 4.8: Interfaz gráfica del ejemplo 2.

En esta página se ejecuta el *script* realizado en Octave siguiendo los mismos comandos del ejemplo anterior. Permittiéndole al usuario descargar el *script* en el caso que así lo desee.

Para descargar el *script* se editan las siguientes líneas de comando:

```
<a href="ejercicios/2/practical.m" target="_blank"
class="Estilo1"><u>Descargar Script</u></a>
```

Dentro de esta etiqueta y de su cierre (`</a>`) encontraremos el enlace a la rutina. Para que un enlace esté activo debemos indicar dentro de él el destino del mismo con el atributo `href` que suele llamarse *anchor text* y es el contenido que será visible en el navegador.

**Ejemplo 3:** Programa que estima los parámetros de dos sistemas, este ejemplo es tomado de [2].

Los sistemas son simulados generando 1000 datos, con una entrada binaria pseudo aleatoria de amplitud  $\sigma = 1$ . Se estima los parámetros mediante el método de mínimos cuadrados.

En este ejemplo se muestran los resultados gráficamente y se muestran los resultados de los parámetros estimados. Para imprimir estos resultados en la ventana se edita el siguiente script en PHP:

Script 4.4: resul\_3.php

```
<?php
/*Programa que realiza la ejecución de GNU Octave a través de la
función passthru y retorna el resultado, en este caso una imagen de
tipo png.
Autor: Br.Maria Elena Noriega
Tutor: Prof.Pablo Lischinsky.
Realizado el 25/03/2008.*/
header("Content-type: text/html");/*Función que envia el resultado.*/
passthru('/usr/bin/octave --silent resul_ejerc3.m');/*Función que
ejecuta a GNU Octave.*/
?>
```

En este caso, el *script* retorna un resultado tipo texto es por esto que la función `header("Content-type: text/html");` indica que en la cabecera http se devuelve un archivo de texto.

Igualmente se puede descarga el *script* hecho en Octave de la misma manera como fue explicado en el ejemplo anterior.



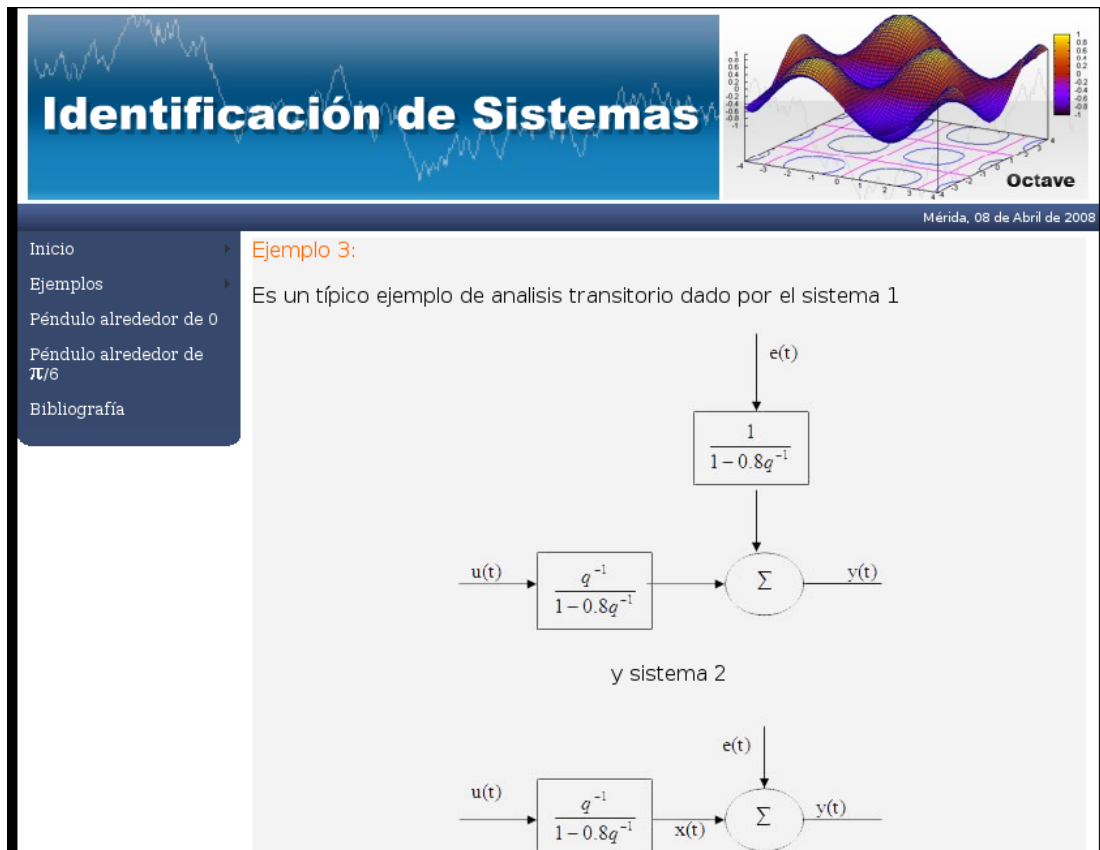


Figura 4.9: Interfaz gráfica del ejemplo 3.

Siguiendo con las opciones de menú se localiza las experiencias con el péndulo alrededor de un punto de operación 0. En esta página se puede observar las experiencias para cada valor diferente de diferencial de amplitud (DA) y diferencial de tiempo (DT). Se puede ver gráficamente el valor de DT y DA para una señal de entrada binaria pseudo aleatoria. Y se puede hacer la identificación del péndulo usando una estructura de modelo ARX introduciendo en un formulario diferentes valores de  $na$ ,  $nb$  y  $nk$ , de forma análoga como lo hace el SIT mediante el comando **arx** (Ver Figura 4.10).



Figura 4.10: Interfaz gráfica para realizar las experiencias del péndulo alrededor de 0.

Para ver cada una de las experiencias se escriben las siguientes líneas de comando:

```
<a href="#" class="Estilo3"
onClick="ventana('ejercicios/4/43/verExpe/graf_ejemplo4_3.php') ">Ver
experiencia 3</a><a href="experiencia_3.php" class="Estilo3">
```

Luego de crear el vínculo con la etiqueta `< a >` con atributo href se muestra el contenido que será visible en el navegador haciendo una llamada al evento con `onClick` para cargar las experiencias entrada/salida que encuentra en el archivo `graf_ejemplo4_3.php` (Ver Figura 4.11).

## Script 4.5: graf\_ejemplo4-3.php

/\*Programa que realiza la ejecución de GNU Octave a través de la función passthru y retorna el resultado, en este caso una imagen de tipo png.

Autor: Br.Maria Elena Noriega

Tutor: Prof.Pablo Lischinsky.

Realizado el 25/03/2008.\*/

```
<?php
```

```
header("Content-type: image/png"); /* Función que retorna una imagen de tipo png. */
```

```
passthru('/usr/bin/octave --silent graf_ejemplo4_3.m');/*Función que ejecuta a GNU Octave.*/
```

```
?>
```

Con este script se levanta la siguiente ventana que muestra las distintas experiencias:

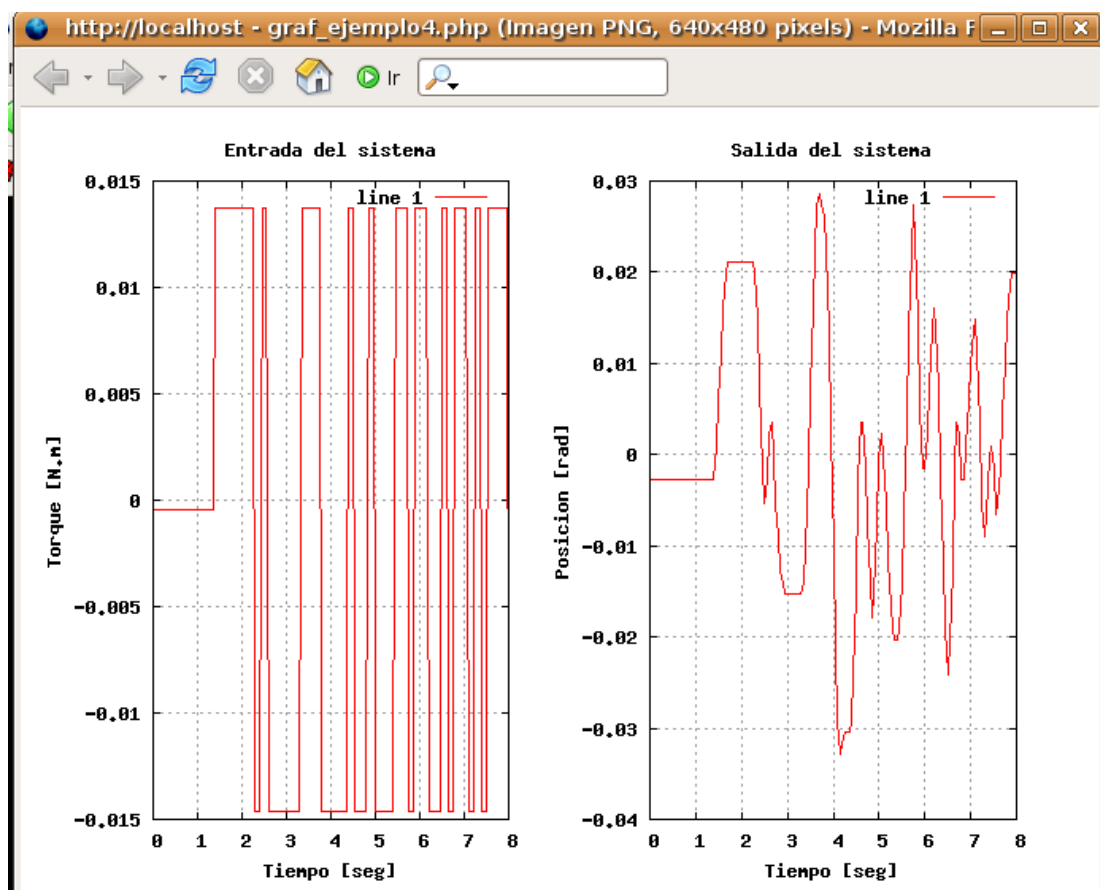


Figura 4.11: Ventana que muestra una experiencia del péndulo.

Posteriormente, para identificar el modelo ARX utilizando cada una de las experiencias se llena el formulario y se validan cada uno de los campos con la siguiente función, utilizando javascript.

```
<script language='javascript'>
function validar(formX){ //Función validar que contiene las
condiciones de validar los campos
if(formX.na.value == ""){ //Condición si el campo esta vacío
    alert("Ingrese Valor na");//Mensaje en caso de error
    formX.na.focus();// Colocamos el cursor en el elemento
    formX.na.value = ""; //Limpiamos el campo
    return false;//Retornamos false para que no se envíe el
formulario
}
if(isNaN(formX.na.value)){//Condición para saber si es numérico o no
    alert("Solo se admiten datos numéricos");
    formX.na.focus();
    formX.na.value = "";
    return false;
}
if(formX.na.value >6 || formX.na.value <=0){//Condición que evalúa el
rango de valor permitido
    alert("Solo admite datos orden desde 1 hasta 6");
    formX.na.focus();// Colocamos el cursor en el elemento
    formX.na.value = "";//Limpiamos el campo
    return false; //Retorna false para que no se envíe el
formulario
}
if(formX.nb.value == ""){
    alert("Ingrese Valor nb");
    formX.nb.focus();// Colocamos el cursor en el elemento
    formX.nb.value = "";//Limpiamos el campo
    return false; //Retorna false para que no se envíe el
formulario
}
if(isNaN(formX.nb.value)){
    alert("Solo se admiten datos numéricos");
    formX.nb.focus();// Coloca el cursor en el elemento
    formX.nb.value = "";//Limpia el campo
    return false; //Retorna false para que no se envíe el
formulario
}
if(formX.nb.value >6 || formX.nb.value <=0){
    alert("Solo admite datos orden desde 1 hasta 6");
    formX.nb.focus();// Coloca el cursor en el elemento
    formX.nb.value = "";//Limpia el campo
    return false; //Retorna false para que no se envíe el
formulario
}
}
```

```

if(formX.nk.value == ""){
    alert("Ingrese Valor nk");
    formX.nk.focus();// Coloca el cursor en el elemento
    formX.nk.value = ""; //Limpia el campo
    return false; //Retorna false para que no se envíe el
formulario
}
if(isNaN(formX.nk.value)){
    alert("Solo se admiten datos numéricos");
    formX.nk.focus();// Coloca el cursor en el elemento
    formX.nk.value = ""; //Limpia el campo
    return false; //Retorna false para que no se envíe el
formulario
}
if(formX.nk.value >6 || formX.nk.value <=0){
    alert("Solo admite datos orden desde 1 hasta 6");
    formX.nk.focus();// Coloca el cursor en el elemento
    formX.nk.value = ""; //Limpia el campo
    return false; //Retorna false para que no se envíe el
formulario
}
if(formX.nb.value > formX.na.value){
    alert("El valor de nb no puede ser mayor que el de na");
    formX.nb.focus();//Coloca el cursor en el elemento
    formX.nb.value = ""; //Limpia el campo
    return false; //Retorna false para que no se envíe el
formulario
}
if(formX.nk.value > formX.nb.value){
    alert("El valor de nb no puede ser mayor que el de na");
    formX.nk.focus();//Coloca el cursor en el elemento
    formX.nk.value = ""; //Limpia el campo
    return false; //Retorna false para que no se envíe el
formulario
}

```

Luego de validar cada uno de los campos se construye el formulario como sigue: (Ver Figura 4.12).

```

<form name="datos" action="ejercicios/4/recibir.php" method="post"
onSubmit="return validar()">
<input type="text" name="na"/>
<input type="text" name="nb"/>
<input type="text" name="nk"/>
<input type="submit" value="enviar" name="enviar"/></form>

```

Las etiquetas `<form>...</form>` delimitan el comienzo y el final de un formulario. Estos datos son trasladados al *script* recibir.php mediante el método post (los datos son adjuntados al cuerpo del formulario).

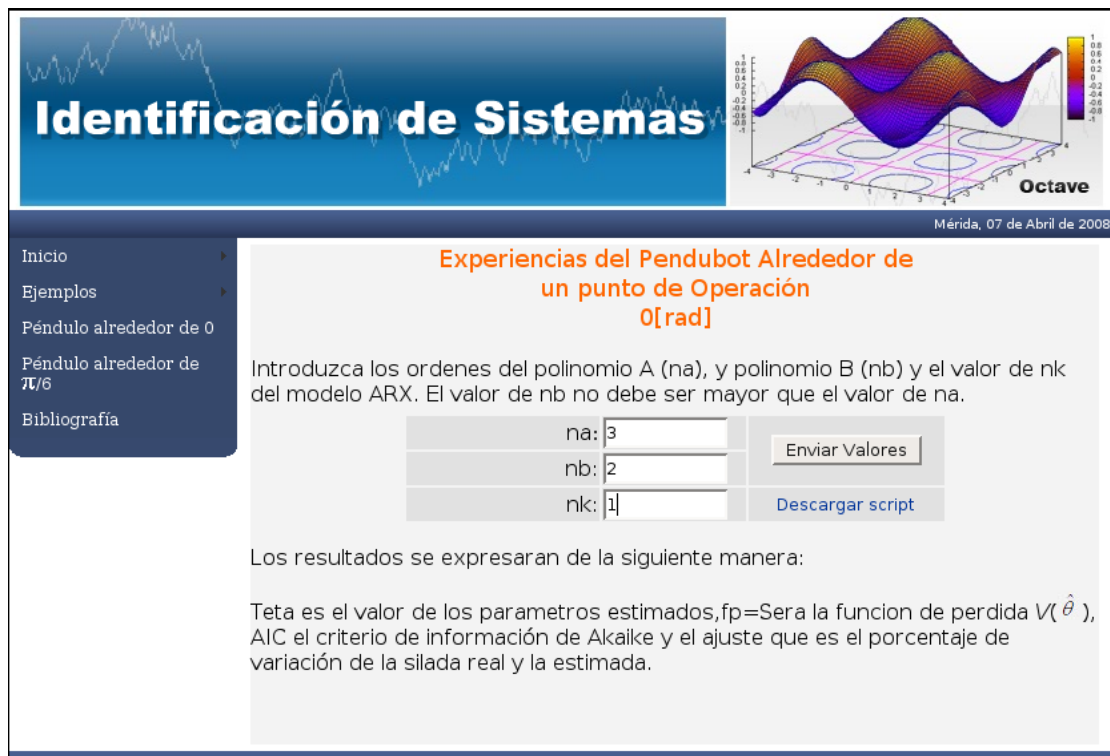


Figura 4.12: Formulario para realizar la estimación de parámetros de cada una de las experiencias del Péndulo.

En la ventana (Figura 4.12) se hace una reseña de la presentación de los resultados. La conexión entre PHP y Octave para realizar la ejecución del archivo .m se sigue de la misma manera como se explico anteriormente. De la misma manera se le permite al usuario descargar el *script*.

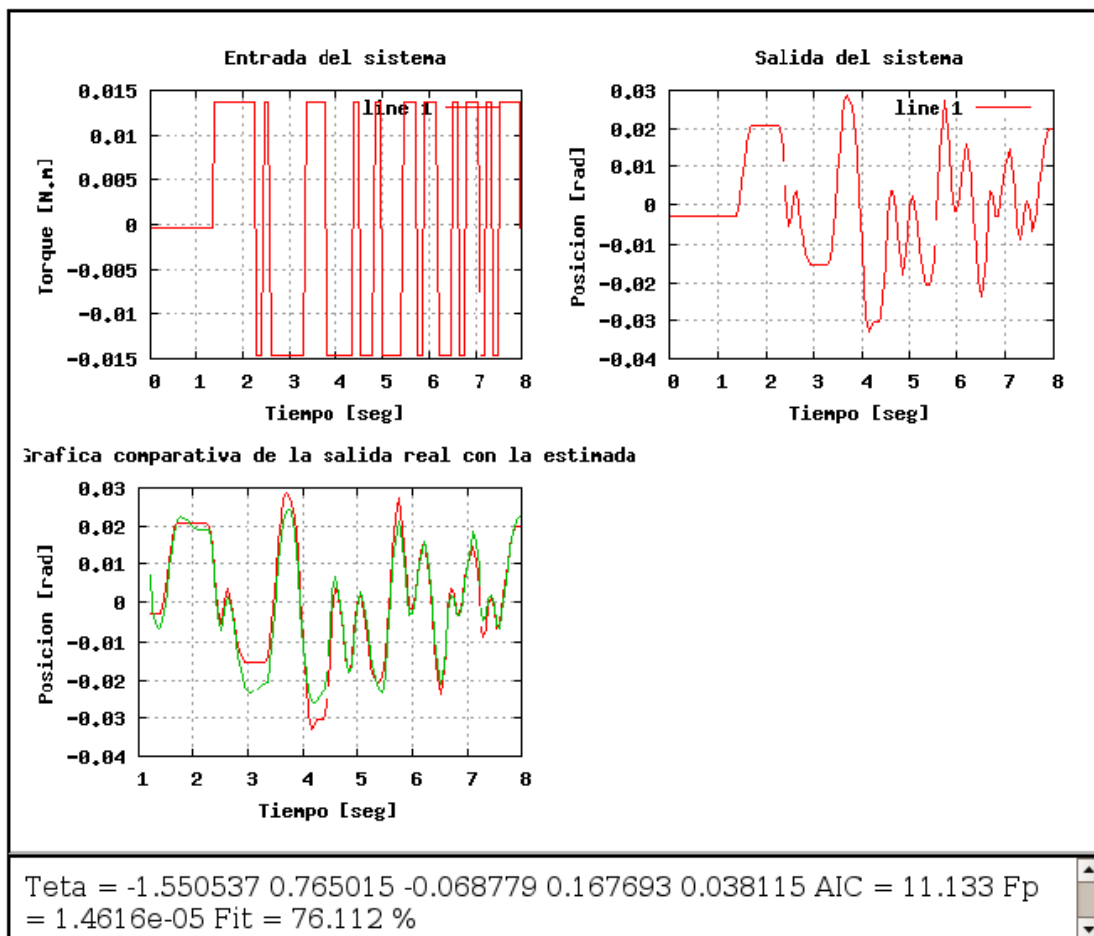


Figura 4.13: Interfaz gráfica resultante del script realizado en Octave para realizar la identificación de la experiencia del péndulo.

Los resultados mostrados en la figura 4.13 expresan lo siguiente:

Teta: Los parámetros estimados por el método de mínimos cuadrados.

AIC: Criterio de información de Akaike, para determinar la mejor estructura (de acuerdo a los ordenes) del modelo.

Ajuste: Porcentaje de similitud entre la salida real y la salida estimada.

En esta aplicación de igual forma se realiza la identificación del péndulo alrededor de  $\pi/6$ , con los mismos eventos anteriormente mencionados, pero la misma no se muestra aquí pues no es necesario dado que significaría mostrar prácticamente el mismo contenido ya mostrado.

Finalmente la interfaz muestra la bibliografía manejada para llevar a cabo esta aplicación vía Web (Ver Figura 4.14).

**Identificación de Sistemas**

Mérida, 09 de Abril de 2006

**Bibliografía**

- T. Söderström y P. Stoica. *System Identification*. Prentice Hall International, 1989.
- L. Ljung. *"System Identification. Theory for the user"*. Prentice Hall. 1994.
- Glenda Z. González. Desarrollo de experiencias de modelado y control a tiempo real de un Péndulo. Proyecto de grado en EISULA. Universidad de los Andes Mérida, 2002.
- Andy M. Ramírez. Uso de GNU Octave para simulación de Sistemas no Lineales. Proyecto de grado en EISULA. Universidad de los Andes Mérida, 2006.
- Yaneth Godoy. Cálculo numérico vía Web con PHP y GNU Octave: algunas aplicaciones en sistemas de control. Proyecto de grado en EISULA. Universidad de los Andes Mérida, 2006.
- Manual de PHP. (2007). Disponible en: <http://www.php.net/docs.php>.
- J. Eaton. GNU Octave.(2006).Disponible en: <http://www.octave.org/>.

Universidad de Los Andes

Figura 4.14: Interfaz gráfica la bibliografía utilizada.



## CAPITULO 5

### CONCLUSIONES Y RECOMENDACIONES

La identificación de sistemas permite construir modelos lineales discretos a partir de datos experimentales de entrada/salida, Matlab es un *software* propietario, el cual es una herramienta de cálculo numérico que proporciona un conjunto de algoritmos para la estimación de parámetros. De allí que se realizó una interfaz Web, beneficiándonos de la interacción entre PHP y Octave, presentado unos ejemplos de estimación de parámetros mediante el método de mínimos cuadrados.

Se logró identificar el péndulo a partir de las experiencias entrada/salida obtenidas, utilizando una estructura de modelos ARX, permitiendo al usuario introducir los valores de  $na$ ,  $nb$  y  $nk$  en un formulario Web. Se compararon estas estimaciones con el comando **arx** del *System Identification Toolbox* de Matlab. Se desarrollaron diferentes ejemplos para entender la estimación de parámetros utilizando el método de mínimos cuadrados.

La interfaz diseñada es muy sencilla y presenta una explicación introductoria a la identificación de sistemas y describe el prototipo experimental utilizado para obtener las diferentes experiencias. Además permite que el usuario pueda ejecutar los scripts elaborados en Octave para identificar el sistema. La interfaz permite también ver las diferentes experiencias realizadas en el péndulo, teniendo acceso a un video que muestra el procedimiento de la captura de datos del Pendubot. Se permitió que el usuario pueda descargar los programas (*scripts*) desarrollados en Octave.

La construcción de una herramienta de software libre vía Internet presenta el diseño de una plataforma para la implementación de un laboratorio virtual interactivo. El sistema tiene la ventaja de que el usuario puede disponer de esta herramienta las 24 horas del día, ya que es accesible desde cualquier sitio con conexión a Internet.

La herramienta diseñada facilita al usuario la utilización de la aplicación, pues Octave no requiere de ningún tipo de instalación ni licencia, es decir sin costo adicional.

Algunas recomendaciones son las siguientes:

- Desarrollar en Octave un *Toolbox* de identificación de sistemas.
- Ampliar Octave para aplicaciones en teoría de control.
- Aprovechar Octave para realizar laboratorios virtuales debido a que se ejecuta perfectamente desde una página Web elaborada en PHP.
- A partir de este estudio el estudiante podrá hacer estimaciones de modelos dinámicos utilizando otros métodos de estimación.

## BIBLIOGRAFÍA

- [1] T. Söderström y P. Stoica. *System Identification*. Prentice Hall International, 1989.
- [2] L. Ljung. *System Identification. Theory for the user*. Prentice Hall, 1994.
- [3] Garrido, Santiago. Identificación, estimación y control de sistemas no-líneales mediante RGO. Universidad Carlos III de Madrid. Departamento de Ingeniería de Sistemas y Automática. Leganés, 1999.
- [4] D. Wackerly, W. Mendenhall y R. Scheaffer. *Estadística matemática con aplicaciones*. THOMSON, 2002.
- [5] Paul L. Meyer. *Probabilidad y Aplicaciones Estadísticas*. Addison\_Wesley, 1992.
- [6] W. Keith Nicholson. *Álgebra lineal con aplicaciones*. Mc Graww Hill. 2003.
- [7] F. Gil, J. Tejedor, A. Yague, S. Alonzo y A. Gutiérrez. *Creaciones de sitios web con PHP 4*. Mc Graw Hill, 2001.
- [8] Glenda Z. González. Desarrollo de experiencias de modelado y control a tiempo real de un Péndulo. Proyecto de grado en EISULA. Universidad de los Andes. Facultad de Ingeniería. Mérida 2002.
- [9] Andy M. Ramírez. Uso de GNU Octave para simulación de Sistemas no Lineales. Proyecto de grado en EISULA. Universidad de los Andes. Facultad de Ingeniería. Mérida 2006.

- [10] Yaneth Godoy. Cálculo numérico vía Web con PHP y GNU Octave: algunas aplicaciones en sistemas de control. Proyecto de grado en EISULA. Universidad de los Andes. Facultad de Ingeniería. Mérida 2006.
- [11] Candelas, F., Sánchez J. Recursos didácticos basados en Internet para el apoyo a la enseñanza de materias de área de ingeniería de sistemas y automática. España. 2005.
- [12] Torres, Leticia. Simulación de sistemas de control en Java: partes conceptuales y metodologicas para un laboratorio virtual. Proyecto de grado en EISULA. Universidad de los Andes. Facultad de Ingeniería. Mérida 2006.
- [13] López, Maria. Identificacion de Sistemas. Aplicación al modelado de un motor continuo. Obtenido el día 22 de marzo de 2008, desde [http://www.depeca.uah.es/docencia/ING-ECA/ctr\\_avz/Identif.PDF](http://www.depeca.uah.es/docencia/ING-ECA/ctr_avz/Identif.PDF)
- [14] Manual de PHP. (2007). Disponible en: <http://www.php.net/docs.php>.
- [15] J. Eaton. GNU Octave.(2006).Disponible en: <http://www.octave.org/>.
- [16] The MathWorks, Inc (1994-2008). Disponible en : <http://www.mathworks.com/>.

## APÉNDICE A

Este apéndice contiene los *scripts* realizados en Octave.

1. *Script* realizado en Octave que ejecuta el ejemplo 1 mostrado en la interfaz.

### Ejemplo1.m

```
#Programa que estima la recta de regresion que mejor se ajusta a
los datos, utlizando el metodo de #minimos cuadrados simple.El
modelo que mejor se ajusta a los datos es aquel cuya funcion de
#perdida sea minima.
#Autor: Br. Maria Elena Noriega R.
#Tutor: Prof. Pablo Lischinsky.
#Realizado el 28/03/2008.
#Datos de entrada y salida
u = [0 1 2 3 4 5 6]';
y = [1.2 1.9 2.4 3.7 4.9 6.1 7.8]';
figure;
plot(u,y,'ro',2)
hold on
grid
title('Comparación de los modelos aproximados por mínimos
cuadrados')
xlabel('u')
ylabel('y')
%Estimación de parámetros dado el Modelo 1
%y=b0;
fi = [ 1 1 1 1 1 1 1 ]';
%Ecuación normal del método de mínimos cuadrados
%teta = inv(fi'*fi)*fi'*y;
tetal=fi\y;
teta_vector= [tetal tetal tetal tetal tetal tetal tetal];
plot(u,teta_vector,'g-')
%Cálculo de la función de pérdida
v1=0;
for i=1:7
    v1=(y(i) - fi(i,:)*tetal)^2+v1;
end
vt=v1/2;
%Estimación de parámetros dado el Modelo 2
% y = bo + b1 u
fi = [ 1 1 1 1 1 1 1 ;
      0 1 2 3 4 5 6 ]';
```

```
%Ecuación normal del método de mínimos cuadrados
%teta = inv(fi'*fi)*fi'*y;
teta2=fi\y;
plot(u,teta2(1)+teta2(2)*u,'r-+')
%Cálculo de la función de pérdida
v2=0;
for i=1:7
    v2=(y(i) - fi(i,:)*teta2)^2 + v2;
end
vt=v2/2;
%Estimación de parámetros dado el Modelo 3
% y = bo + b1 u + b2 u^2
fi=[1 0 0;1 1 1;
    1 2 4;1 3 9;
    1 4 16; 1 5 25;1 6 36];
%Ecuación normal del método de mínimos cuadrados
%teta = inv(fi'*fi)*fi'*y
teta3=fi\y;
plot(u,teta3(1)+teta3(2)*u+teta3(3)*u.^2,'b')
%Cálculo de la función de pérdida
v3 = 0;
for i=1:7
    v3=(y(i) - fi(i,:)*teta3)^2 + v3;
end
v3=v3/2;
%Estimación de parámetros dado el Modelo 4
%y = bo + b1 u + b2 u^2+ b3 u^3'
fi=[1 0 0 0;
    1 1 1 1;
    1 2 4 8;
    1 3 9 27;
    1 4 16 64;
    1 5 25 125;
    1 6 36 216];
%Ecuación normal del metodo de minimos cuadrados
%teta = inv(fi'*fi)*fi'*y;
teta4=fi\y;
plot(u,teta4(1)+teta4(2)*u+teta4(3)*u.^2+teta4(4)*u.^3,'m--x')
%Calculo de la funcion de perdida
v4 = 0;
for i=1:7
    v4=(y(i) - fi(i,:)*teta4)^2 + v4;
end
v4=v4/2;
```

2. *Script* realizado en Octave que ejecuta el ejemplo 2 mostrado en la interfaz.

### Ejemplo 2.m

```
#Programa que realiza la estimación de parámetros de un modelo de
la forma
#  $Y(t)+a_1*Y(t-1)+a_2*Y(t-2)=b_1*U(t-1)+b_2*U(t-2)$ 
#utilizando el método de mínimos cuadrados resolviéndolo por la
ecuación normal y por la forma
#matricial
#Autor: Br. Maria Elena Noriega R.
#Tutor: Prof. Pablo Lischinsky.
#Realizado el 28/03/2008.

%Cargar la experiencia del pendubot
load prbs302
y=trace_y(2,:);
u=trace_y(1,:);
t=trace_x;
%Tratamiento de los datos, elimina las tendencias en los datos de
entrada y salida
y=dtrend(y);
u=dtrend(u);
subplot(2,2,1)
plot(t,y)
title('Datos Adquiridos Y(t)')
subplot(2,2,2)
plot(t,u);
title('Entrada del Sistema U(t)');
y=y';
u=u';
t=t';
% Método de Mínimos Cuadrados
% a) A través de la formula
%Modelo:  $y(t)+a_1*y(t-1)+a_2*y(t-2)=b_1*u(t-1)+b_2*u(t-2)$ 
%Calculo de la matriz Fi
for i=3:1501
    y1(i,1)=-y(i-1);
    y2(i,1)=-y(i-2);
    u1(i,1)=u(i-1);
    u2(i,1)=u(i-2);
end
fi=[y1 y2 u1 u2];
%Ecuación normal del método de mínimos cuadrados
%teta = inv(fi'*fi)*fi'*y;
Tetal=fi\y;
%Cálculo del y_estimado
ym1(1,1)=y(1,1);
ym1(2,1)=y(2,1);
```

```

for i=3:1501
    ym1(i,1)=-Teta1(1,1)*ym1(i-1)-Teta1(2,1)*ym1(i-
2)+Teta1(3,1)*u(i-1)+Teta1(4,1)*u(i-2);
end
%grafica comparativa de la salida real y la salida estimada
subplot(2,2,3)
plot(t,y,'r',t,ym1,'b')
grid
title('Salida Real y Salida Estimada');
%Calculo de la funcion de perdida
E1=abs(y-ym1);
V1=.5*(E1'*E1);
% b) Estimación de parámetros utilizando la Forma Matricial
%V(Teta)= Sumatoria(E^2)
a11=0;a12=0;a13=0;a14=0;
a21=0;a22=0;a23=0;a24=0;
a31=0;a32=0;a33=0;a34=0;
a41=0;a42=0;a43=0;a44=0;
b1=0;b2=0;b3=0;b4=0;
for i=3:1501
    a11 = a11 + y(i-1)^2;
    a12 = a12 + y(i-1)*y(i-2);
    a13 = a13 - y(i-1)*u(i-1);
    a14 = a14 - y(i-1)*u(i-2);
    b1  = b1 - y(i)*y(i-1);
    a21 = a21 + y(i-1)*y(i-2);
    a22 = a22 + y(i-2)^2;
    a23 = a23 - y(i-2)*u(i-1);
    a24 = a24 - y(i-2)*u(i-2);
    b2  = b2 - y(i)*y(i-2);
    a31 = a31 - y(i-1)*u(i-1);
    a32 = a32 - y(i-2)*u(i-1);
    a33 = a33 + u(i-1)^2;
    a34 = a34 + u(i-1)*u(i-2);
    b3  = b3 + y(i)*u(i-1);
    a41 = a41 - y(i-1)*u(i-2);
    a42 = a42 - y(i-2)*u(i-2);
    a43 = a43 + u(i-1)*u(i-2);
    a44 = a44 + u(i-2)^2;
    b4  = b4 + y(i)*u(i-2); end
A2=[a11 a12 a13 a14;a21 a22 a23 a24;a31 a32 a33 a34;a41 a42 a43
a44;];
B2=[b1;b2;b3;b4];
Teta2=A2\B2;
%Cálculo del y_estimado obtenido a través de la forma matricial
ym2(1,1)=y(1,1);
ym2(2,1)=y(2,1);
for i=3:1501
    ym2(i,1)=-Teta2(1,1)*ym2(i-1)-Teta2(2,1)*ym2(i-
2)+Teta2(3,1)*u(i-1)+Teta2(4,1)*u(i-2);

```



```
end
%Grafica comparativa de la salida real y la salida estimada
subplot(2,2,4)
plot(t,y,'r',t,ym2,'b')
grid
title('Salida real y Salida Estimada Forma Matricial');
%Calculo de la funcion de perdida
%V(Teta)= E'E
E2=abs(y-ym2);
V2=E2'*E2;
```

3. *Script* realizado en Octave que ejecuta el ejemplo 3 mostrado en la interfaz.

### Ejemplo 3.m

```
#Programa que estima los parámetros de dos sistemas 1 y 2
respectivamente
#Se estimaran los parámetros para los dos sistemas
#Autor: Br. Maria Elena Noriega R.
#Tutor: Prof. Pablo Lischinsky.
#Realizado el 28/03/2008.
#La entrada sera una senal binaria pseudo aleatoria con una
amplitud de 1
u=idinput(1023,'prbs',[-1 1]);
n=size(u);
#Simulacion del sistema 1
q1=tf([1],[1 -0.8],1);
for i=1:n(1)
    e(i) = randn(1);
end
T=1:n(1);
[y_ax1,x]=lsim(q1,u,T);
q2=tf([1 0],[1 -0.8],1);
e=e';
[y_ax2,x]=lsim(q2,e,T);
yt1=y_ax1+y_ax2;
subplot(2,2,1)
plot(u)
title('Entrada al Sistema')
subplot(2,2,2)
plot(yt1)
title('Salida del Modelo1')
#Simulacion del sistema 2
e=e';
yt2=y_ax1+e;
subplot(2,2,3)
plot(yt2)
title('Salida del Modelo2')
```

```
#Estimación de parámetros para el sistema1 mediante método de
mínimos cuadrados
a11=0; a12=0; a22=0;
b1=0; b2=0;
for i=2:n(1)
    a11 = a11 + yt1(i-1)^2;
    a12 = a12 + yt1(i-1)*u(i-1);
    a22 = a22 + u(i-1)^2;
    b1  = b1 + yt1(i)*yt1(i-1);
    b2  = b2 + yt1(i)*u(i-1);
end
A=[a11 -a12;-a12 a22];
B=[-b1;b2];
Teta1=A\B;
#Estimación de parámetros para el sistema 2 mediante método de
mínimos cuadrados
a11=0; a12=0; a22=0;
b1=0; b2=0;
for i=2:n(1)
    a11 = a11 + yt2(i-1)^2;
    a12 = a12 + yt2(i-1)*u(i-1);
    a22 = a22 + u(i-1)^2;
    b1  = b1 + yt2(i)*yt2(i-1);
    b2  = b2 + yt2(i)*u(i-1);
end
A=[a11 -a12;-a12 a22];
B=[-b1;b2];
Teta2=A\B;
```

4. *Script* realizado en Octave que ejecuta la estimación de parámetros para las experiencias del péndulo.

#### **arx2g.m**

```
#ESTIMACION DE PARAMETROS UTILIZANDO MODELO ARX MEDIANTE EL METODO
DE MINIMOS CUADRADOS
#Autor: Br. Maria Elena Noriega R.
#Tutor: Prof. Pablo Lischinsky.
#Realizado el 28/03/2008.
warning("off");
load prbscincomedia
y=trace_y(2,:);
u=trace_y(1,:);
t=trace_x;
y=dtrend(y);
u=dtrend(u);
y=y';
u=u';
t=t';
```

```
datos=[y,u];
n=size(y);
n=n(:,1);
subplot(2,2,1)
plot(t,u),grid;title('Entrada del Sistema U(t)'),xlabel('Tiempo
[seg]'),ylabel('Torque [N.m]');
title('Entrada del sistema')
subplot(2,2,2)
plot(t,y),grid,title('Datos Adquiridos Y(t)'),xlabel('Tiempo
[seg]'),ylabel('Posición [rad]');
title('Salida del sistema')
na=input('Introduce el valor de na:');
nb=input('Introduce el valor de nb:');
nk=input('Introduce el valor de nk:');
a11=0;
a12=0;
a13=0;
a14=0;
a15=0;
b11=0;
b12=0;
b13=0;
b14=0;
b15=0;
switch (na)
    case (1)
        for i=2:n
            a11(i,1)=y(i-1);
        end
    case (2)
        for i=2:n
            a11(i,1)=y(i-1);
        end
    case (3)
        for i=2:n
            a11(i,1)=y(i-1);
        end
        for i=3:n
            a12(i,1)= y(i-2);
        end
    case (4)
        for i=2:n
            a11(i,1)=y(i-1);
        end
        for i=3:n
            a12(i,1)= y(i-2);
        end
        for i=4:n
            a13(i,1)= y(i-3);
        end
    case (5)
        for i=2:n
            a11(i,1)=y(i-1);
        end
```

```
        for i=3:n
            a12(i,1)=y(i-2);
        end
    for i=4:n
        a13(i,1)= y(i-3);
    end
        for i=5:n
            a14(i,1)= y(i-4);
        end
        case (5)
            for i=2:n
                a11(i,1)=y(i-1);
            end
            for i=3:n
                a12(i,1)=y(i-2);
            end
            for i=4:n
                a13(i,1)= y(i-3);
            end
            for i=5:n
                a14(i,1)= y(i-4);
            end
        end
        for i=6:n
            a15(i,1)=y(i-5);
        end
        case (6)
            for i=2:n
                a11(i,1)=y(i-1);
            end
            for i=3:n
                a12(i,1)=y(i-2);
            end
            for i=4:n
                a13(i,1)= y(i-3);
            end
            for i=5:n
                a14(i,1)= y(i-4);
            end
            for i=6:n
                a15(i,1)=y(i-5);
            end
            for i=7:n
                a16(i,1)=y(i-6);
            end
        end
        otherwise
            disp('solo hasta orden 6');
            break;
    end
end
switch (nb)
```

```
        case (1)
    for i=nk+1:n
        b11(i,1)=u(i-nk);
    end
        case(2)
    for i=nk+1:n
        b11(i,1)=u(i-nk);
    end
    for i=nk+nb:n
        b12(i,1)=u(i-nk-nb+1);
    end
        case (3)
    for i=nk+1:n
        b11(i,1)=u(i-nk);
    end
    for i=nk+2:n
        b12(i,1)= u(i-nk-1);
    end
    for i=nk+nb:n
        b13(i,1)= u(i-nk-nb+1);
    end
        case (4)
    for i=nk+1:n
        b11(i,1)=u(i-nk);
    end
    for i=nk+2:n
        b12(i,1)= u(i-nk-1);
    end
    for i=nk+3:n
        b13(i,1)= u(i-nk-2);
    end
    for i=nk+nb:n
        b14(i,1)=u(i-nk-nb+1);
    end
        case (5)
    for i=nk+1:n
        b11(i,1)=u(i-nk);
    end
    for i=nk+2:n
        b12(i,1)= u(i-nk-1);
    end
    for i=nk+3:n
        b13(i,1)= u(i-nk-2);
    end
    for i=nk+4:n
        b14(i,1)=u(i-nk-3);
    end
    for i=nk+nb:n
        b15(i,1)=u(i-nk-nb+1);
    end
```

```
        case (6)
    for i=nk+1:n
        b11(i,1)=u(i-nk);
    end
    for i=nk+2:n
        b12(i,1)= u(i-nk-1);
    end
    for i=nk+3:n
        b13(i,1)= u(i-nk-2);
    end
    for i=nk+4:n
        b14(i,1)=u(i-nk-3);
    end
    for i=nk+5:n
        b15(i,1)=u(i-nk-4);
    end
    for i=nk+nb:n
        b16(i,1)=u(i-nk-nb+1);
    end
    otherwise
        disp('solo hasta orden 6');
    end
    a=0; b=0;
    fi=zeros(n,na+nb);
    if(na==1)
        a=[a11];
    else if (na==2)
        a=[a11 a12];
    else if (na==3)
        a=[a11 a12 a13];
    else if (na==4)
        a=[a11 a12 a13 a14];
    else if(na==5)
        a=[a11 a12 a13 a14 a15];
    else (na==6);
        a=[a11 a12 a13 a14 a15 a16];
    end end end end end
    if(nb==1)
        b=[b11];
    else if (nb==2)
        b=[b11 b12];
    else if (nb==3)
        b=[b11 b12 b13];
    else if (nb==4)
        b=[b11 b12 b13 b14];
    else if(nb==5)
        b=[b11 b12 b13 b14 b15];
    else(nb==6);
        b=[b11 b12 b13 b14 b15 b16];
    end end end end end
```

```
%CALCULO DEL TETA
fi=[-a b];
Teta=fi\y
Teta=Teta';
%CALCULO DEL YESTIMADO
A=Teta(1:na);
B=Teta(na+1:na+nb);
switch (na)
    case(1)
        ym1(1,1)=y(1,1);
        if(nb==1)
            for i=2:n
                ym1(i,1)=-A(1,1)*ym1(i-1)+B(1,1)*u(i-1);
            end
        end
        if(nb==2)
            for i=3:n
                ym1(i,1)=-A(1,1)*ym1(i-1)+B(1,1)*u(i-1)+B(1,2)*u(i-2);
            end
        end
        if(nb==3)
            for i=4:n
                ym1(i,1)=-A(1,1)*ym1(i-1)+B(1,1)*u(i-1)+B(1,2)*u(i-2)+B(1,3)*u(i-3);
            end
        end
        if(nb==4)
            for i=5:n
                ym1(i,1)=-A(1,1)*ym1(i-1)+B(1,1)*u(i-1)+B(1,2)*u(i-2)+B(1,3)*u(i-3)+B(1,4)*u(i-4);
            end
        end
        if(nb==5)
            for i=6:n
                ym1(i,1)=-A(1,1)*ym1(i-1)+B(1,1)*u(i-1)+B(1,2)*u(i-2)+B(1,3)*u(i-3)+B(1,4)*u(i-4)+B(1,5)*u(i-5);
            end
        end
        if(nb==6)
            for i=7:n
                ym1(i,1)=-A(1,1)*ym1(i-1)+B(1,1)*u(i-1)+B(1,2)*u(i-2)+B(1,3)*u(i-3)+B(1,4)*u(i-4)+B(1,5)*u(i-5)+B(1,6)*u(i-6);
            end
        end
    case(2)
        if(nb==1)
            for i=3:n
                ym1(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)+B(1,1)*u(i-1);
            end
        end
```

```
end
if(nb==2)
    for i=3:n
        yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)+B(1,1)*u(i-
1)+B(1,2)*u(i-2);
    end
end
if(nb==3)
    for i=4:n
        yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)+B(1,1)*u(i-
1)+B(1,2)*u(i-2)+B(1,3)*u(i-3);
    end
end
if(nb==4)
    for i=5:n
        yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)+B(1,1)*u(i-
1)+B(1,2)*u(i-2)+B(1,3)*u(i-3)+B(1,4)*u(i-4);
    end
end
if(nb==5)
    for i=6:n
        yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)+B(1,1)*u(i-
1)+B(1,2)*u(i-2)+B(1,3)*u(i-3)+B(1,4)*u(i-4)+B(1,5)*u(i-5);
    end
end
if(nb==6)
    for i=7:n
        yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)+B(1,1)*u(i-
1)+B(1,2)*u(i-2)+B(1,3)*u(i-3)+B(1,4)*u(i-4)+B(1,5)*u(i-
5)+B(1,6)*u(i-6);
    end
end
case(3)
    if(nb==1)
        for i=4:n
            yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-
3)+B(1,1)*u(i-1);
        end
    end
    if(nb==2)
        for i=4:n
            yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-
3)+B(1,1)*u(i-1)+B(1,2)*u(i-2);
        end
    end
    if(nb==3)
        for i=4:n
            yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-
3)+B(1,1)*u(i-1)+B(1,2)*u(i-2)+B(1,3)*u(i-3);
        end
    end
```



```
end
if(nb==4)
    for i=5:n
        yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)+B(1,1)*u(i-1)+B(1,2)*u(i-2)+B(1,3)*u(i-3)+B(1,4)*u(i-4);
    end
end
if(nb==5)
    for i=6:n
        yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)+B(1,1)*u(i-1)+B(1,2)*u(i-2)+B(1,3)*u(i-3)+B(1,4)*u(i-4)+B(1,5)*u(i-5);
    end
end
if(nb==6)
    for i=7:n
        yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)+B(1,1)*u(i-1)+B(1,2)*u(i-2)+B(1,3)*u(i-3)+B(1,4)*u(i-4)+B(1,5)*u(i-5)+B(1,6)*u(i-6);
    end
end
case(4)
    if(nb==1)
        for i=5:n
            yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-A(1,4)*ym1(i-4)+B(1,1)*u(i-1);
        end
    end
    if(nb==2)
        for i=5:n
            yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-A(1,4)*ym1(i-4)+B(1,1)*u(i-1)+B(1,2)*u(i-2);
        end
    end
    if(nb==3)
        for i=5:n
            yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-A(1,4)*ym1(i-4)+B(1,1)*u(i-1)+B(1,2)*u(i-2)+B(1,3)*u(i-3);
        end
    end
    if(nb==4)
        for i=5:n
            yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-A(1,4)*ym1(i-4)+B(1,1)*u(i-1)+B(1,2)*u(i-2)+B(1,3)*u(i-3)+B(1,4)*u(i-4);
        end
    end
    if(nb==5)
        for i=6:n
```

```
        yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)+B(1,1)*u(i-1)+B(1,2)*u(i-2)+B(1,3)*u(i-  
3)+B(1,4)*u(i-4)+B(1,5)*u(i-5);  
    end  
end  
if(nb==6)  
    for i=7:n  
        yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)+B(1,1)*u(i-1)+B(1,2)*u(i-2)+B(1,3)*u(i-  
3)+B(1,4)*u(i-4)+B(1,5)*u(i-5)+B(1,6)*u(i-6);  
    end  
end  
case(5)  
    if(nb==1)  
for i=6:n  
    yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)-A(1,5)*ym1(i-5)+B(1,1)*u(i-1);  
end  
        end  
        if(nb==2)  
for i=6:n  
    yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)-A(1,5)*ym1(i-5)+B(1,1)*u(i-1)+B(1,2)*u(i-2);  
end  
        end  
        if(nb==3)  
for i=6:n  
    yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)-A(1,5)*ym1(i-5)+B(1,1)*u(i-1)+B(1,2)*u(i-  
2)+B(1,3)*u(i-3);  
end  
        end  
        if(nb==4)  
for i=6:n  
    yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)-A(1,5)*ym1(i-5)+B(1,1)*u(i-1)+B(1,2)*u(i-  
2)+B(1,3)*u(i-3)+B(1,4)*u(i-4);  
end  
        end  
        if(nb==5)  
for i=6:n  
    yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)-A(1,5)*ym1(i-5)+B(1,1)*u(i-1)+B(1,2)*u(i-  
2)+B(1,3)*u(i-3)+B(1,4)*u(i-4)+B(1,5)*u(i-5);  
end  
        end  
        if(nb==6)  
for i=7:n
```

```
        yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)-A(1,5)*ym1(i-5)+B(1,1)*u(i-1)+B(1,2)*u(i-  
2)+B(1,3)*u(i-3)+B(1,4)*u(i-4)+B(1,5)*u(i-5)+B(1,6)*u(i-6);  
    end  
    end  
        case(6)  
            if(nb==1)  
for i=7:n  
            yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)-A(1,5)*ym1(i-5)-A(1,6)*ym1(i-6)+B(1,1)*u(i-1);  
            end  
                end  
                if(nb==2)  
for i=7:n  
yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)-A(1,5)*ym1(i-5)-A(1,6)*ym1(i-6)+B(1,1)*u(i-  
1)+B(1,2)*u(i-2);  
            end  
                end  
                if(nb==3)  
for i=7:n  
            yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)-A(1,5)*ym1(i-5)+B(1,1)-A(1,6)*ym1(i-6)*u(i-  
1)+B(1,2)*u(i-2)+B(1,3)*u(i-3);  
            end  
                end  
                if(nb==4)  
for i=7:n  
            yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)-A(1,5)*ym1(i-5)-A(1,6)*ym1(i-6)+B(1,1)*u(i-  
1)+B(1,2)*u(i-2)+B(1,3)*u(i-3)+B(1,4)*u(i-4);  
            end  
                end  
                if(nb==5)  
for i=7:n  
            yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)-A(1,5)*ym1(i-5)-A(1,6)*ym1(i-6)+B(1,1)*u(i-  
1)+B(1,2)*u(i-2)+B(1,3)*u(i-3)+B(1,4)*u(i-4)+B(1,5)*u(i-5);  
            end  
                end  
                if(nb==6)  
for i=7:n  
            yml(i,1)=-A(1,1)*ym1(i-1)-A(1,2)*ym1(i-2)-A(1,3)*ym1(i-3)-  
A(1,4)*ym1(i-4)-A(1,5)*ym1(i-5)-A(1,6)*ym1(i-6)+B(1,1)*u(i-  
1)+B(1,2)*u(i-2)+B(1,3)*u(i-3)+B(1,4)*u(i-4)+B(1,5)*u(i-  
5)+B(1,6)*u(i-6);  
            end  
        endend  
        otherwise  
            disp('solo hasta orden 6'); end
```

```
%CRITERIO UTILIZADO PARA LA DETERMINACION DEL ORDEN DEL %MODELO (O
ESTRUCTURA DEL MODELO)
%CRITERIO DE AKAIKE
%AIC(p)=log V(teta)+ 2p/N
%Siendo V(teta)=1/n sumatoria E^2(t,teta)
%E(t,teta)=y(t)-yestiamdo(t/teta)
%CALCULO DEL ERROR
Error=abs(y-ym1);
%CÁLCULO DE LA FUNCION DE PERDIDA
Var=(1/n)*(Error'*Error);

%CRITERIO DE AKAIKE
%AIC(p)=N log V(teta) + 2*p
p=na+nb;
AIC=abs((log(Var))+((2*p)/n))
Fp=Var*(1+p/n)/(1-p/n)
subplot(2,2,3)
plot(t,y,'r',t,ym1,'g'),grid
title('Grafica comparativa de la salida real con la
estimada'),xlabel('Tiempo [seg]'),ylabel('Posición [rad]');
Ajuste = 100*(1 - norm(ym1 - y)/norm(y-mean(y)))
```

En el apéndice B se muestran algunos *scripts* realizados en PHP para diseñar la interfaz Web.

### 1. *Script* PHP que levanta la pagina de inicio de la interfaz.

#### **inicio.php**

```
/* Programa para la construcción de la pagina de inicio de la
interfaz
Autor: Br. Maria Elena Noriega R.
Tutor: Prof. Pablo Lischinsky.
Realizado el 28/03/2008.*/
<?php include('fecha.php');//Comando para incluir el archivo
fecha.php q contiene la funcion de la fecha actual
?>
<html>
<head>
<title>Principal</title>
<link href="estilo.css" rel="stylesheet" type="text/css" />
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"
/>
<!-- etiquetas style CSS las cuales contiene el estilo de la
pagina: color de fondo,de letra,tipo.. etc-->
<style type="text/css">
<!--
body {
    background-color: #000000;
}
.Estilo1 {font-family: Verdana, Arial, Helvetica, sans-serif}
.Estilo2 {font-family: Verdana, Arial, Helvetica, sans-serif;
font-weight: bold; }
.Estilo3 {color: #0000CC}
-->
</style>
</head>
<body>
<!--Etiquetas table conforman la tabla principal del sitio -->
<table width="813" height="100%" border="0" align="center"
cellpadding="00" cellspacing="00" bgcolor="#FFFFFF">
    <!-- Etiquetas tr y td representan las columnas y las filas-->
    <tr>
        <td height="150" colspan="3" valign="bottom">
            <div align="center">
                <?php include('banner.html');// Incluir el archivo
                banner.html que contiene el banner en flash ?>
            </div> </td> </tr> <tr>
                <td height="21" colspan="3" background="images/barraS.jpg">
```

```

<div align="right" class="fechaB">
  <? = show_date();// funcion cargada del archivo fecha.php
?> </div> </td> </tr> <tr>
  <td width="20%" valign="top">
    <table width="171" border="0" align="center" cellpadding="0"
cellspacing="0">
      <tr>
        <td width="171" height="19" valign="top">
          <?php include("menu.html");// Cargar el archivo menu.html que
contiene el menu del sitio ?> </td>
        </tr> <tr>
          <td height="17" valign="top"></td> </tr>
</table> </td>
      <td width="80%" colspan="2" valign="middle">
        <table width="98%" height="100%" border="0" align="center"
cellpadding="0" cellspacing="5" > <tr>
          <td height="225" valign="top"
bgcolor="#F3F3F3"><blockquote>
            <br> <p class="Estilo2">Identificación de sistemas</p>
            <p align="justify" class="Estilo1">La identificación de
un sistema permite obtener un modelo lineal de la planta en
estudio, valido alrededor de un punto de operación en forma de una
función de transferencia continua o discreta, mediante la cual se
puede describir en forma aproximada la dinámica del proceso.</p>
            <p align="justify" class="Estilo1">La identificación de
sistemas es un conjunto de técnicas para construir modelos
matemáticos de sistemas dinámicos utilizando experiencias de
entrada y salida.<br>
            </p> </blockquote>
            <p align="center"
class="Estilo1"><br>
            <br> Esquema General de Identificación de Sistemas
            </p> <p align="right" class="Estilo1"><a
href="proceso.php" class="Estilo3"><u>Proceso de
identificación</u></a></p></td>
          </tr> </table> </td> </tr> <tr>
          <td height="43" colspan="3" background="images/barraI.jpg" >
            <div align="center">
              <?php include('pie.html'); // cargar el archivo pie.php que
contiene el pie de pagina ?> </div></td>
            </tr></table>
</body>
</html>

```

## 2. Programa que construye el menú desplegable de la interfaz.

### Menu

```
/* Programa para la construcción del menú desplegable.
Autor: Br. Maria Elena Noriega R.
Tutor: Prof. Pablo Lischinsky.
Realizado el 28/03/2008.*/
<style type="text/css">
<!--
#apDiv1 {
    position:absolute;
    left:183px;
    top:17px;
    width:173px;
    height:120px;
    z-index:1;
}
-->
</style>
<script src="SpryAssets/SpryMenuBar.js"
type="text/javascript"></script>
<link href="SpryAssets/SpryMenuBarVertical.css" rel="stylesheet"
type="text/css" />
<ul id="MenuBar1" class="MenuBarVertical">
    <li><a class="MenuBarItemSubmenu MenuBarItemSubmenu"
href="#">Inicio</a>
        <ul>
            <li><a href="index.php">Identificaci&ocirc;n de
sistemas</a></li>
            <li><a href="proceso.php">Proceso de
identificaci&ocirc;n</a></li>
            <li><a href="estructura.php">Modelo ARX</a></li>
            <li><a href="pendulo.php">P&eacute;ndulo</a></li>
        </ul>
    </li>
    <li><a href="#" class="MenuBarItemSubmenu">Ejemplos</a>
        <ul>
            <li><a href="ejemplos.php">Ejemplo 1</a></li>
            <li><a href="ejemplosC1.php">Ejemplo 2</a></li>
            <li><a href="ejemplosC2.php">Ejemplo 3</a></li>
        </ul>
    </li>
    <li><a href="ejemplosC4.php">P&eacute;ndulo alrededor de 0</a>
</li>
    <li><a href="ejemplosC5.php">P&eacute;ndulo alrededor de <br />
        /6 </a></li>
</ul>
<script type="text/javascript">
<!--
```

```
var MenuBar1 = new Spry.Widget.MenuBar("MenuBar1",
{imgRight:"SpryAssets/SpryMenuBarRightHover.gif"});
//-->
</script>
```

**3. Programa que valida los datos del formulario y se enlaza con .php ejecutando el *script* para realizar la estimación de las diferentes experiencias.**

### **Experiencia.php**

/\* Programa que valida los datos del formulario, crea el formulario y hace el enlace que permite ejecutar el script en Octave para realizar la estimación de las diferentes experiencias del Péndulo y descargar el script.

Autor: Br. Maria Elena Noriega R.

Tutor: Prof. Pablo Lischinsky.

Realizado el 28/03/2008.\*/

```
<?php include('fecha.php'); ?>
<html>
<head>
<title>Ejemplos varios</title>
<link href="estilo.css" rel="stylesheet" type="text/css" />
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"
/>
<script language='javascript'>
function validar(formX){ //Funcion validar q contiene las
condiciones de validar los campos
if(formX.na.value == ""){ //Condicion si el campo esta vacio
    alert("Ingrese Valor na");//Mensaje en caso de error
    formX.na.focus(); // Colocamos el cursor en el elemento
    formX.na.value = ""; //Limpiamos el campo
    return false;//Retornamos false para q no se envíe el
formulario
}
if(isNaN(formX.na.value)){//Condicion para saber si es numerico o
no
    alert("Solo se admiten datos numericos");
    formX.na.focus();
    formX.na.value = "";
    return false;
}
if(formX.na.value >6 || formX.na.value <=0){//Condicion q evalua
el rango de valor permitido
    alert("Solo admite datos orden desde 1 hasta 6");
    formX.na.focus();
    formX.na.value = "";
    return false;
}
}
```



```
if(formX.nb.value == ""){
    alert("Ingrese Valor nb");
    formX.nb.focus();
    formX.nb.value = "";
    return false;
}
if(isNaN(formX.nb.value)){
    alert("Solo se admiten datos numericos");
    formX.nb.focus();
    formX.nb.value = "";
    return false;
}
if(formX.nb.value >6 || formX.nb.value <=0){
    alert("Solo admite datos orden desde 1 hasta 6");
    formX.nb.focus();
    formX.nb.value = "";
    return false;
}
if(formX.nk.value == ""){
    alert("Ingrese Valor nk");
    formX.nk.focus();
    formX.nk.value = "";
    return false;
}
if(isNaN(formX.nk.value)){
    alert("Solo se admiten datos numericos");
    formX.nk.focus();
    formX.nk.value = "";
    return false;
}
if(formX.nk.value >6 || formX.nk.value <=0){
    alert("Solo admite datos orden desde 1 hasta 6");
    formX.nk.focus();
    formX.nk.value = "";
    return false;
}
if(formX.nb.value > formX.na.value){
    alert("El valor de nb no puede ser mayor que el de na");
    formX.nb.focus();
    formX.nb.value = "";
    return false;
}
if(formX.nk.value > formX.nb.value){
    alert("El valor de nb no puede ser mayor que el de na");
    formX.nk.focus();
    formX.nk.value = "";
    return false;
}
}
}
</script>
```

```
<style type="text/css">
<!--
body {
    background-color: #000000;
}
.Estilo1 {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    font-size: 18px;
    color: #FF6600;
}
.Estilo2 {font-family: Verdana, Arial, Helvetica, sans-serif}
.Estilo3 {color: #003399}
.Estilo4 {color: #0000CC}
.Estilo4 {font-family: Verdana, Arial, Helvetica, sans-serif;
color: #0000CC; }
.Estilo5 {font-size: 12px}
-->
</style>
</head>
<body>
<table width="813" height="100%" border="0" align="center"
cellpadding="00" cellspacing="00" bgcolor="#FFFFFF">
    <tr>
        <td height="150" colspan="3" valign="bottom">
            <div align="center">
                <?php include('banner.html');?>
            </div> </td>
    </tr> <tr>
        <td height="21" colspan="3" background="images/barraS.jpg">
            <div align="right" class="fechaB">
                <?= show_date();?>
            </div> </td>
    </tr> <tr>
        <td width="20%" valign="top">
            <table width="147" border="0" align="center" cellpadding="0"
cellspacing="0">
                <tr> <td width="147" height="19" valign="top">
                    <?php include("menu.html");
                    ?></td></tr> <tr>
                        <td height="17" valign="top"></td>
                    </tr> </table>
                </td> <td width="80%" colspan="2" valign="middle"><table
width="98%" height="100%" border="0" align="center"
cellpadding="0" cellspacing="5" >
                    <tr>

<td height="225" valign="top" bgcolor="#F3F3F3"><p align="center"
class="Estilo1">Experiencias del Pendubot Alrededor de<br>
un punto de Operación<br>
```

```
0[rad]</p>
    <form name="datos" action="ejercicios/4/4-
1/generar_graf_4.php" method="post" onSubmit="javascript:return
validar(this)">
        <span class="Estilo2">Introduzca los ordenes del
polinomio A (na), y polinomio B (nb) y el valor de nk del modelo
ARX. El valor de nb no debe ser mayor que el valor de na. </span>
        <table width="64%" border="0" align="center">
            <tr>
                <td colspan="7" class="Estilo2 Estilo5"><div
align="justify"></div></td>
            </tr> <tr>
                <td width="32%" align="right"
bgcolor="#E1E1E1"><span class="Estilo2">na</span></td>
                <td width="13%" bgcolor="#E1E1E1"><input id="na"
name="na" type="text" size="10"/></td>
                <td width="33%" rowspan="2" align="center"
bgcolor="#E1E1E1" class="Estilo2"><input type="submit"
value="Enviar Valores" name="enviar" /></td>
            </tr> <tr>
                <td align="right" bgcolor="#E1E1E1"
class="Estilo2">nb:</td>
                <td bgcolor="#E1E1E1"><input name="nb" type="text"
size="10"/></td>
            </tr> <tr>
                <td align="right" bgcolor="#E1E1E1"
class="Estilo2">nk:</td>
                <td bgcolor="#E1E1E1"><input name="nk" type="text"
size="10"/></td>
                <td align="center" bgcolor="#E1E1E1"
class="Estilo2"><a href="ejercicios/4/4-1/arx2gl.m"
target="_blank" class="Estilo3"> Descargar script</a></td> </tr>
        </table>
        <p><span class="Estilo2">Los resultados se
expresaran de la siguiente manera:</span></p>
        <p class="Estilo2">Teta es el valor de los parametros
estimados,fp=Sera la funcion de perdida <em>V</em>(), AIC el
criterio
de información de Akaike y el ajuste que es el porcentaje de
variación de la silada real y la estimada.</p>
    </form>
    <p>&nbsp;</p>
    <table width="637" border="0" cellpadding="0"
cellspacing="0">
        <tr>
            <td><span class="Estilo1 Estilo2"><a
href="ejemplosC2.php" class="Estilo4"></a></span></td>
            <td><div align="right"></div></td>
        </tr> </table></td>
</tr> </table>
```

```
</td> </tr> <tr>
<td height="43" colspan="3" background="images/barraI.jpg" >
<div align="center">
  <?php include('pie.html'); ?>
</div> </td>
</tr>
</table>
</body>
</html>
```

4. Programa que realiza el levantamiento de la página (en esta caso, una ventana) donde se observará el resultado que arroja el programa que se esta ejecutando en el servidor.

#### **ejemplo.php**

```
<?php
/*Programa que realiza la ejecución de GNU Octave a través de la
función passthru y retorna el resultado, en este caso una imagen
de tipo png.
Autor: Br. Maria Elena Noriega R.
Tutor: Prof. Pablo Lischinsky.
Realizado el 01/03/2008.*/
header("Content-type: image/png");/*Función que retorna una imagen
de tipo png.*/
passthru('/usr/bin/octave --silent ejercicio1.m');/*Función que
ejecuta a GNU Octave.*/
?>
```

