Software para Identificación de Sistemas

En la práctica la solución analítica de los problemas de identificación puede ser muy difícil, por lo que se debe recurrir a algoritmos y computadores. Es el caso del TOOLBOX "System Identification" de MATLAB. (ver también lsselect, EXCEL)

En este TOOLBOX la función **arx** está diseñada para determinar parámetros de modelos como el que aquí se ha expuesto (perturbados por ruido blanco).

help arx entrega lo que sigue*:*

ARX Computes LS-estimates of ARX-models.

M = ARX(DATA,ORDERS)

M: returned with the estimated parameters of the ARX model

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

along with estimated covariances and structure information.

For single output systems, M is an IDPOLY model object, while for multi-outputs systems, M is an IDARX object.

See IDPROPS, IDPROPS IDPOLY, or IDPROPS IDARX for all details.

DATA: The estimation data as an IDDATA object. See HELP IDDATA.

ORDERS = [na nb nk], the orders and delays of the above model.
For multi-output systems, ORDERS has as many rows as there are outputs

na is then an ny|ny matrix whose i-j entry gives the order of the polynomial (in the delay operator) relating the j:th output to the i:th output. Similarly nb and nk are ny|nu matrices. (ny:# of outputs, nu:# of inputs). For a time series, ORDERS = na only.

An alternative syntax is m = ARX(DATA,'na',na,'nb',nb,'nk',nk)

*In the multi-output case, ARX minimizes the norm of E'*inv(LAMBDA)*E,*
where E are the prediction errors and LAMBDA is the NoiseVariance of
an initial model (default the unit matrix).

Parameters and options associated with the algorithm can be set as property/
value pairs in any order. Omitted properties are given default values. Typical
options to use are:

FOCUS ('Prediction', 'simulation', 'stability' or a prefilter.)
INPUTDELAY
FIXEDPARAMETER (Parameters in a nominal model to remain fixed.)
NOISEVARIANCE (Determines the output norm in the multioutput case.)
MAXSIZE (Determines the size of the largest matrix to be formed.)

See also IDPROPS ALGORITHM and IDPROPS IDMODEL for more details,
ARXSTRUC, SELSTRUC for how to estimate the order parameters, and
AR, ARMAX, BJ, IV4, N4SID, OE, PEM for other estimation routines.
Overloaded methods
help idarx/arx.m

Por ejemplo, supongamos que la planta está regida por la ecuación de diferencia

$$y_1(t) = -a_1\, y_1(t-1) - a_2\, y_1(t-2) + b_2\, u(t-4) + b_3 u(t-5) + w(t)\,.$$

$$y_1(t) = -a_1\, y_1 z^{-1} - a_2\, y_1 z^{-2} + \left(b_1\, u + b_2 u z^{-1}\right) z^{-4} + w(t)$$

$$y_1(t) = -a_1\, y_1(t-1) - a_2\, y_1(t-2) + b_1\, u(t-4) + b_2 u(t-5) + w(t)\,.$$

$$y_1(t) = -a_1\, y_1 z^{-1} - a_2\, y_1 z^{-2} + \left(b_1\, u + b_2 u z^{-1}\right) z^{-4} + w(t)$$

Si conociéramos la estructura del modelo da la planta, entonces nuestro modelo sería

$$\hat{y}_1(t) = -\hat{a}_1\, y_1 z^{-1} - \hat{a}_2\, y_1 z^{-2} + \left(\hat{b}_1\, u + \hat{b}_2 u z^{-1}\right) z^{-4}$$

$$y_1(t) = -\hat{a}_1\, y_1 z^{-1} - \hat{a}_2\, y_1 z^{-2} + \left(\hat{b}_1\, u + \hat{b}_2 u z^{-1}\right) z^{-4} + \varepsilon(t)$$

Entonces, comparando con los polinomios *A* y *B,*

$$A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_{n_a} z^{-n_a}$$
$$B(z^{-1}) = b_1 + b_2 z^{-1} + b_3 z^{-2} + \ldots + b_{n_b} z^{-n_b+1}$$

y considerando un retardo mínimo n_k para la entrada *u,* es decir,

$$A(z^{-1})y = B(z^{-1}) z^{-n_k} u + w$$

se tiene que

$n_a = 2$; $n_b = 2$; $n_k = 4$. (También: n_a = N° de parámetros de *A*; n_b = N° de parámetros de *B* ; n_k = retardo mínimo de *u*).

y *w(t)* es ruido blanco (media cero) (Modelo ARX). (Notar que el proceso de encontrar la *estructura* consiste en determinar los órdenes n_a y n_b de los polinomios *A* y *B* y del retardo n_k de la señal de entrada *u(t)* y, equivalentemente, si el modelo está dado en formato theta). Por lo tanto:

DATA = [y u]
ORDERS = [2 2 4]

En DATA = [*y u*], *y* es un vector columna que contiene los valores de *y(t)* para *t* = 1, 2, 3, ... ,N (igual que en el formato XY); *u* es un vector columna si hay una entrada (o una matriz si hay *r* entradas, cuyas columnas son la *r* entradas) que contiene los valores de *u(t)* para *t* = 1, 2, 3, ...,N.

Entonces

Mod21= arx([y u],[2 1 1])

calcula los parámetros del modelo que minimizan el error medio cuadrático

$$E\left\{\varepsilon^2\right\}$$

a través de aproximaciones en el tiempo de los valores esperados.

$$\hat{\theta}_t = \left[\sum_{i=1}^{t} \varphi_i \varphi_i^T\right]^{-1} \sum_{i=1}^{t} \varphi_i y_i \approx \hat{\theta}$$

o, equivalentemente,

$$\hat{\theta}_t = \left[X^T X\right]^{-1} X^T Y \approx \hat{\theta}$$

(ver ID2_717_2007_c.doc)

Además calcula las desviaciones standard de la estimación, el error cuadrático, etc. (ver los ejemplos más adelante).

present (Mod21) entrega los parámetros estimados del Modelo *Mod21*, junto con la desviación standard del error de estimación de esos parámetros, el error cuadrático de la modelación, etc.

Si se tuviera que la planta está regida por

$$y_1(t) = -a_1\, y_1(t-1) - a_2\, y_1(t-2) + b\, u_1(t) + w(t)$$

entonces $n_a = 2$, $n_b = 1$ y $n_k = 0$.

DATA =[*y u*]
ORDERS =[*na nb nk*]=[2 1 0]

Si conociéramos esta estructura, nuestro modelo sería

$$\hat{y}_1(t) = -\hat{a}_1\, y_1(t-1) - \hat{a}_2\, y_1(t-2) + \hat{b}\, u_1(t)$$

y con

Modelo2= arx([y u],[2 1 0])

se obtendrán las estimaciones de los parámetros y otros datos (ver ejemplos más adelante)

Por otra parte, si la ecuación de la planta es

$$y_1(t) = -a_1\, y_1(t-1) + b_1\, u_1(t-2) + b_2\, u_1(t-3) + w(t)\,,$$

entonces $n_a = 1$, $n_b = 2$ $n_k = 2$, y DATA = [1 2 2], porque

$$\left(b_1 + b_2\, z^{-1}\right) z^{-2}\, u(t) = b_1\, u(t-2) + b_2\, u(t-3)$$

DATA =[*y u*]
ORDERS =[*na nb nk*]=[1 2 2]

Modelo:

$$\hat{y}_1(t) = -\hat{a}_1\, y_1(t-1) + \hat{b}_1\, u_1(t-2) + \hat{b}_2\, u_1(t-3)$$

y en MATLAB,

```
Modelo3= arx([y u],[1 2 2]).
```

Notar que tenemos tres formatos para especificar la estructura del modelo:

- Formato Theta
- Formato XY
- Formato Polonomial
- Formato MATLAB

Modelo con Varias Entradas.

Si un modelo tiene varias entradas, u_1, u_2, ..., u_r, entonces n_b y n_k son vectores fila tales que:

$$n_b = [\text{num\_param\_}u_1 \quad \text{num\_param\_}u_2 \ldots \text{num\_param\_}u_r],$$

$$n_k = [\text{retard\_min\_}u_1 \quad \text{retard\_min\_}u_2 \ldots \text{retard\_min\_}u_r].$$

Por ejemplo, sea el modelo *Mod22* que tiene dos entradas: u_1 y u_2, cuya ecuación se diferencia es

$$y_1(t) = -a_1 y_1(t-1) - a_2 y_1(t-2) + b_{11} u_1(t-1) + b_{21} u_2(t-1) + b_{22} u(t-2) + w(t)$$

En este caso

$n_a = [2]$,
$n_b = [1\ 2]$,
$n_k = [1\ 1]$,

y nuestro modelo es

$$\hat{y}_1(t) = -\hat{a}_1 y_1(t-1) - \hat{a}_2 y_1(t-2) + \hat{b}_{11} u_1(t-1) + \hat{b}_{21} u_2(t-1) + \hat{b}_{22} u(t-2)$$

En genral, ***present (ModZZ)*** entrega los parámetros estimados del Modelo *ModZZ*, junto con la desviación standard del error de estimación de esos parámetros.

Ejemplo.- Estimación de Parámetros de un Modelo con Una Entrada

Modelo del Sistema

$$y_1(t) = a_1\, y_1(t-1) + a_2\, y_1(t-2) + b\, u_1(t-1) + w(t)$$

donde: $a_1 = -1$; $a_2 = 0.16$; $b = 2$.

El modelo a identificar, entonces, es

$$\hat{y}_1(t) = \hat{a}_1\, y_1(t-1) + \hat{a}_2\, y_1(t-2) + \hat{b}\, u_1(t-1)$$

Se supone que estos parámetros son desconocidos y que deben ser encontrados realizando mediciones de $y_1(t)$ y de $u_1(t)$. El ruido $w(t)$ es blanco con media cero.

Predicción de un paso de y(t) dado lo conocido hasta t –1

.

$$E\{\, y_1(t) \,|\, t-1 \,\} = \hat{y}(t \,|\, t-1)$$

Error de Predicción de Un Paso

$$e(t) = y(t) - y(t \,|\, t-1)$$

La predicción a un paso es

$$\hat{y}_1(t \,|\, t-1) = \hat{a}_1\, y_1(t-1) + \hat{a}_2\, y_1(t-2) + \hat{b}\, u_1(t-1)$$

La predicción de y_1 en el caso en que no se tengan mediciones de y es

$$\hat{y}_1(t) = \hat{a}_1\,\hat{y}_1(t-1) + \hat{a}_2\,\hat{y}_1(t-2) + \hat{b}\,u_1(t-1)$$

Este último es el caso de un ***Sensor Virtual*** y se trata de una predicción para varios instantes de muestreo futuros.

Programa escrito en Matlab para estimar los parámetros de este modelo (Mod21)

```
%IDMOD21_6p5_2006.m
% Estimación de parámetros de un sistema dados los datos de entrada y de
salida
% El programa genera datos desde una planta de segundo orden y
% luego los usa para estimar sus parámetros mediante ARX si el ruido es
blanco o ARMAX, si el
% ruido es coloreado.
%Contiene una etapa de validacion con datos dustinatos a los de ajuste
%(entrenamiento) del modelo.

clear w y1 u1 n

TFin=4000% Largo del registro de satos
sig_n=1 %std del ruido blanco w
a1=-1
a2=0.16
polos=roots([1 -1 0.16])%polos de la planta
b=2
ALFA=0.85, %Param. filtro generador de u1
sigu1=0.05
GAMA=0.8;
```

```
c1=0.8;c2=0.5;

%Genera los datos de la planta. En la practica esto equivale a medir datos
%en la planta

y1=zeros(TFin,1);y1sr=zeros(TFin,1);u1=zeros(TFin,1);w=zeros(TFin,1);n=
zeros(TFin,1);


for i=3:TFin
   w(i)=randn(1)*sig_n;%w(t) es ruido blanco
   %**************************
   %Aqui se puede elgir entre ruido blanco y ruido coloreado
   % n ruido blanco
   n(i)=w(i);
   % *****************
   % n  ruido coloreado
   %n(i)=w(i)+c1*w(i-1); %ruido coloreadode promedio movil
   %n(i)=w(i)+c1*w(i-1)+c2*w(i-2);ruido coloreado de promedio movil
   %n(i)=GAMA*n(i-1)+(1-GAMA)*3*w(i);%ruido coloreado
autoregresivo
   %**************************

   u1(i)=ALFA*u1(i-1)+sqrt(((1+ALFA)/(1-
ALFA)))*sigu1*randn(1);%comando
   y1(i)=-a1*y1(i-1)-a2*y1(i-2)+b*u1(i-1)+n(i);%salida
   y1sr(i)=-a1*y1sr(i-1)-a2*y1sr(i-2)+b*u1(i-1);%salida sin ruido. En la
practica no se dispone de ella.

end


grafico=1
if grafico==1
```

```
figure
        plot([y1 5*u1])
        title(' salida y1 y entrada u1')
   legend(' y1','5*u1')
   axis([100 500 -10 10])
   grid
end
%*************************************************************


%IDENTIFICACIÓN

%Estimación de Parámetros del Modelo Caso ARX (ruido blanco)

%ORDERS= N1=[2 1 1]% [na nb nk];
%DATA= z1=[y1 u1];%matriz con datos de salida y de entrada

z1=[y1 u1];N1=[2 1 1];

Mod21=arx(z1,N1);%caso ruido blanco

%Estimación de Parámetros del Modelo Caso ARMAX (ruido coloreado n)

%N1=[2 1 0 1]%  N1=[na nb nc nk]% n ruido blanco
%N1=[2 1 1 1]%  N1=[na nb nc nk] % n ruido coloreado
n(i)=w(i)+GAMA*w(i-1)
%N1=[2 1 2 1]%  N1=[na nb nc nk] % n ruido coloreado n(i)=w(i)+c1*w(i-
1)+c2*w(i-2)
%N1=[2 1 4 1]%  N1=[na nb nc nk] % n ruido coloreado n(i)=w(i)+c1*w(i-
1)+c2*w(i-2)
%z1=[y1 u1];%matriz con datos de salida y de entrada
%z1=[y1 u1];Mod21=armax(z1,N1);%caso ruido coloreado
```

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


%NOTA: En MATLAB 6.5 set(idpoly), indica como acceder a los datos
resultantes de la
%identificacion. En este caso idpoly = Mod21

ah=Mod21.a %En MATLAB 6.5 entrega el vector de parametros a estimados
bh=Mod21.b %En MATLAB 6.5 entrega el vector de parametros b
estimados
ch=Mod21.c %En MATLAB 6.5 entrega el vector de parametros c estimados

a1h=ah(2);  a2h=ah(3);
b1h=bh(2);

%En MATLAB 6.5 las desv. std. de determinacion de los parametros se
%obtienen asi:

stda=Mod21.da
stdb=Mod21.db
stdc=Mod21.dc

siga1=stda(2)
siga2=stda(3)
sigb=stdb(2)

present(Mod21)%Entrega el modelo incluyendo std de los parametros.




%SENSOR VIRTUAL
```

```
%Usa el modelo para generar y1 estimado cuando la entrad es u1 sin usar
mediciones de y(j) para ningún  j.

y1est=idsim(u1,Mod21);

errest=y1-y1est;
RMSEest= sqrt((errest'*errest)/TFin)

figure
plot([y1 y1est])
legend(' y1',' y1SV')
title([' y1 y su estimación y1est como Sensor Virtual  RMSEest=
',num2str(RMSEest)])
ylabel(['  sig_n = ',num2str(sig_n),'      EMCPr1p=',...
     num2str(Mod21.EstimationInfo.LossFcn)])
xlabel(['a1h= ',num2str(a1h),'   a2h= ',num2str(a2h),'   bh= ',num2str(b1h),...
     '   a1= ',num2str(a1),'   a2= ',num2str(a2),'   b= ',num2str(b)])
grid
axis([100 500 -10 10])


%Estimación de los parámetros usando el formato XY no obstante que esto
%es solo válido para ruido blanco

Ly= length(y1);% largo de y1

Y=y1(3:Ly);% y1(t) desde 3 hasta Ly
y_1=y1(2:Ly-1);%y1(t-1) es y1 desde 2 hasta Ly-1
y_2=y1(1:Ly-2);%y1(t-2) es y1 desde 1 hasta Ly-2
u_1=u1(2:Ly-1);% u(t-1) es u1(t) desde 2 hasta Ly-1

X=[y_1 y_2 u_1];%Matriz de entradas
```

```
%Vector de parámetros estimados.
ThXY=inv(X'*X)*X'*Y

% Parámetros estimados usando el formato XY
a1hXY=-ThXY(1);
a2hXY=-ThXY(2);
bhXY=ThXY(3);

%Matriz de covarianza del error de estimación de los parámetros.
%En la diagonal está la estimación de la varianza del error de
%estiamción de los parámtros
VarTH=inv(X'*X)*(std(n))^2;

%Esimación de la Desv. Std. del error de estimación de los parámetros
(diagonal)
siga1XY=sqrt(VarTH(1,1))
siga2XY=sqrt(VarTH(2,2))
sigbXY=sqrt(VarTH(3,3))

disp(' ')
disp(' ')
disp('parámetros verdaderos')
disp('a1     a2       b')
disp([a1  a2  b ])
disp(' ')
disp(' ')

disp('parámetros según arx o armax')
disp(' ')
disp(' a1h     a2h              bh')
disp([a1h  a2h  b1h ])
disp(' ')
disp(' ')
```

```
disp('parámetros según XY')
disp(' ')
disp('    a1hXY      a2hXY      bhXY')
disp([a1hXY  a2hXY  bhXY ])
disp(' ')
disp(' ')

disp('std según arx o armax')
disp(' ')
disp('   siga1      siga2      sigb')
disp([siga1  siga2  sigb])
disp(' ')
disp(' ')

disp('std según XY')
disp(' ')
disp('   siga1XY sigahXY sigbXY')
disp([siga1XY  siga2XY  sigbXY ])
disp(' ')
disp(' ')

present(Mod21)

y1p1p=predict(z1,Mod21,1);
figure;plot([y1 y1p1p])
legend('y1','yp1p')
title('Predicción de un paso')
axis([100 500 -10 10])
grid


ep1p=y1-y1p1p;
Vp1p=var(ep1p)
```

```
%Validacion

TFV=TFin;
y1pv=zeros(TFV,1); u1=zeros(TFV,1);
GenDat=1;

if GenDat==1
        %Genera nuevos  datos de la planta para validación de modelo
        TFV=TFin;
        for i=3:TFin
      w(i)=randn(1)*sig_n;%w(t) es ruido blanco
      %**************************
      % n ruido blanco
      n(i)=w(i);
      % *****************
      % n  ruido coloreado
      %n(i)=w(i)+c1*w(i-1); %ruido coloreadode promedio movil
      %n(i)=w(i)+c1*w(i-1)+c2*w(i-2);ruido coloreadode promedio movil
      %n(i)=GAMA*n(i-1)+(1-GAMA)*3*w(i);%ruido coloreado
autoregresivo
      %**************************
      u1(i)=ALFA*u1(i-1)+sqrt(((1+ALFA)/(1-
ALFA)))*sigu1*randn(1);%comando
      y1pv(i)=-a1*y1pv(i-1)-a2*y1pv(i-2)+b*u1(i-1)+n(i);%salida
      y1sr(i)=-a1*y1sr(i-1)-a2*y1sr(i-2)+b*u1(i-1);%salida sin ruido.

     end

end


 yval=idsim(u1,Mod21);;
 errval=y1pv-yval;
```

```
 erms_val=sqrt((errval'*errval)/TFin);


figure
plot([y1pv yval])
legend(' y1pv','yval')
title(' VALIDACION: y1pv y su predicción yval calculada con idsim.m')
axis([100 500 -10 10])
grid
xlabel(['N1= [',num2str(N1),']','     Error RMS de Validación =
',num2str(erms_val)])



 yval=idsim(u1,Mod21);;
 errval=y1pv-yval;
 erms_val=sqrt((errval'*errval)/TFin);


figure
plot([y1pv yval])
legend(' y1pv','yval')
title(' VALIDACION: y1pv y su predicción yval calculada con idsim.m')
axis([100 500 -10 10])
grid
xlabel(['N1= [',num2str(N1),']','     Error RMS de Validación =
',num2str(erms_val)])
```

Resultados obtenidos en una realización

ah = 1.0000 -0.9072 0.0782
bh = 0 2.1724
ch = 1

stda = 0 0.0298 0.0279
stdb = 0 0.1236
stdc = 0

siga1 = 0.0298 siga2 = 0.0279 sigb = 0.1236

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

A(q) = 1 - 0.9072 (+-0.02985) q^-1 + 0.07821 (+-0.02795) q^-2

B(q) = 2.172 (+-0.1236) q^-1

Estimated using ARX from data set z1
Loss function 1.09553 and FPE 1.10213 (Predicción a un paso)
Sampling interval: 1
Created: 02-May-2007 18:57:05
Last modified: 02-May-2007 18:57:05

RMSEest = 1.9840 (Sensor Virtual)

ThXY =
0.9072
-0.0782
2.1724
siga1XY = 0.0299
siga2XY = 0.0280
sigbXY = 0.1239

parámetros verdaderos

a1	a2	b
-1.0000	0.1600	2.0000

parámetros según arx o armax

a1h	a2h	bh
-0.9072	0.0782	2.1724

parámetros según XY

a1hXY	a2hXY	bhXY
-0.9072	0.0782	2.1724

std según arx o armax

siga1	siga2	sigb
0.0298	0.0279	0.1236

std según XY

siga1XY	sigahXY	sigbXY
0.0299	0.0280	0.1239

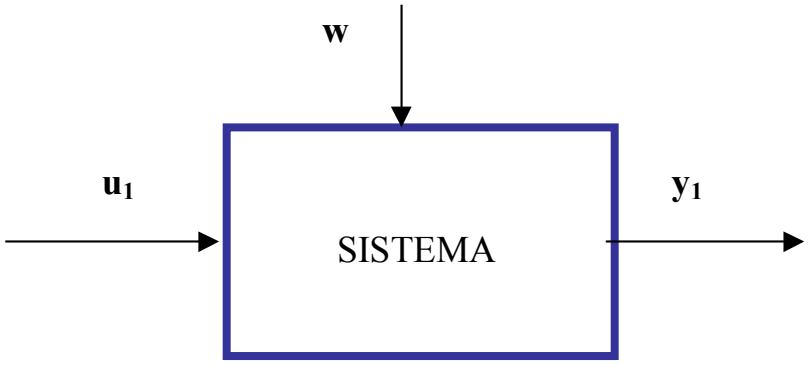
Discrete-time IDPOLY model: A(q)y(t) = B(q)u(t) + e(t)
A(q) = 1 - 0.9072 (+-0.02985) q^-1 + 0.07821 (+-0.02795) q^-2

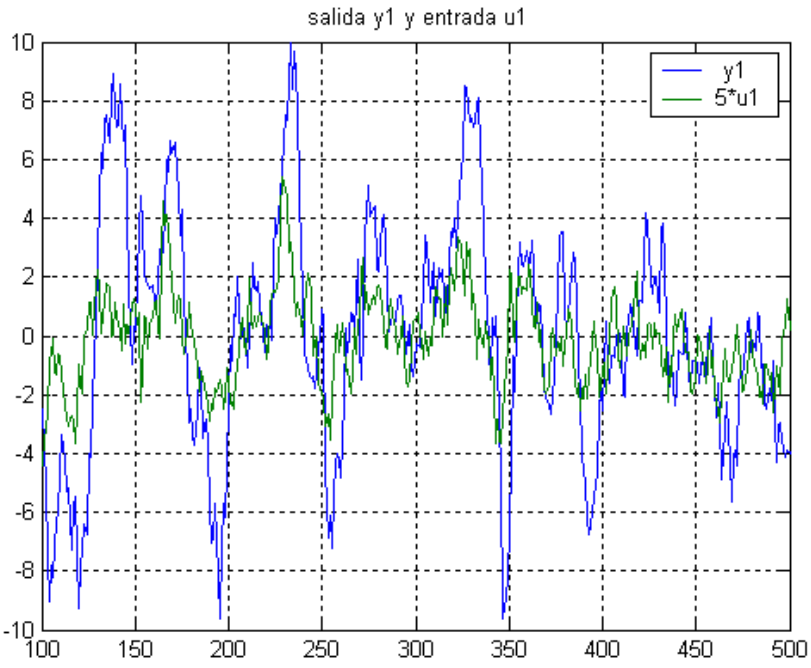
B(q) = 2.172 (+-0.1236) q^-1

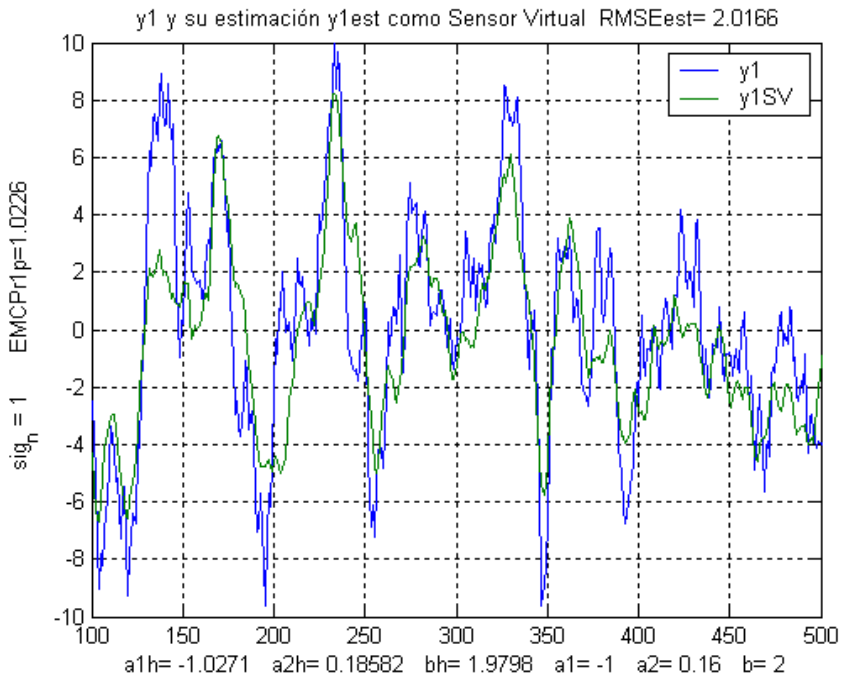
Estimated using ARX from data set z1
Loss function 1.09553 and FPE 1.10213
Sampling interval: 1
Created: 02-May-2007 18:57:05
Last modified: 02-May-2007 18:57:05

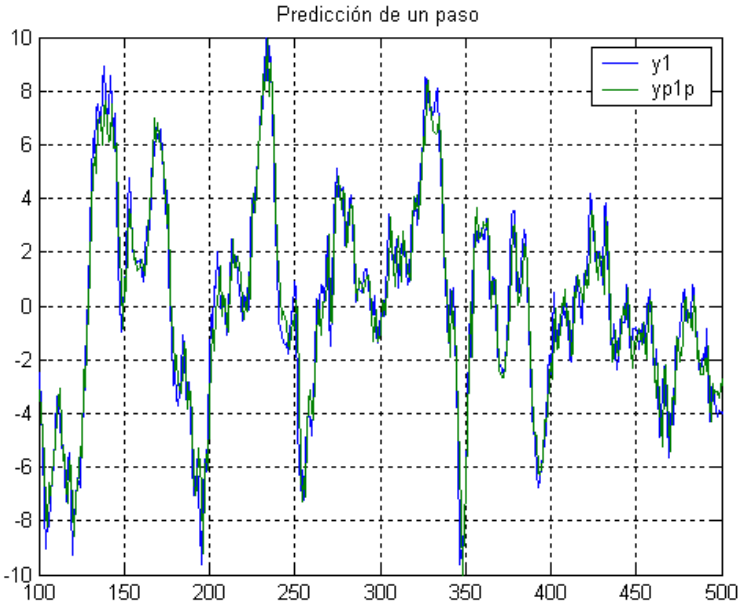
Vp1p = 1.0956

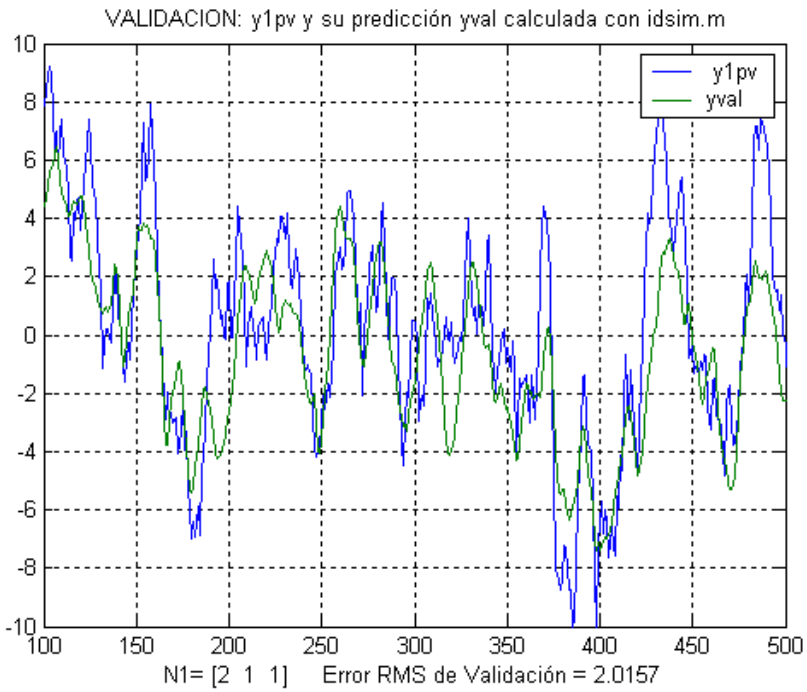
Los parámetros estimados son ***realizaciones de variables aleatorias*** y, por lo tanto, están sujetos error. La desviación standard de ese error está dada por la cantidad que aparece debajo del parámetro correspondiente al ejecutar *present(Mod21);* hay que tener en cuenta que estas desviaciones standard también son estimaciones realizadas con los datos *u(t)* e **y(t)** y que, por lo tanto, también están sujetas a error de estimación.






y1 y su estimación y1est como Sensor Virtual RMSEest= 2.0166
y1
y1SV
sig_n = 1 EMCPr1p=1.0226
a1h= -1.0271 a2h= 0.18582 bh= 1.9798 a1= -1 a2= 0.16 b= 2


Predicción de un paso
y1
yp1p


VALIDACION: y1pv y su predicción yval calculada con idsim.m
y1pv
yval
10
8
6
4
2
0
-2
-4
-6
-8
-10
100
150
200
250
300
350
400
450
500
N1= [2 1 1] Error RMS de Validación = 2.0157

EXTRACCIÓN DE OTROS RESULTADOS DE LA MODELACIÓN.

Para ver todos los datos que se pueden extraer del objeto modelo hacer en Command Window: idprops/ENTER.

Por ejemplo, si se trata del modelo Mod21, el nombre Mod21 es un objeto cuyas propiedades se extraen agregando la propiedad deseada después de un punto a Mod21

a_est = Mod21.a asigna al vector a_est los parámetros estimados del polinomio A del modelo y similarmente para el resto de los parámetros y las desv. std de su estimación (como ya hemos visto)

Mod21.EstimationInfo/ENTER despliega

Status: 'Estimated model (ARX)'
Method: 'ARX'
LossFcn: 0.9415 (error medio cuadrático de modelación)
FPE: 0.9472 (índice FPE de Akaike que se verá en determinación de estructuras)
DataName: 'z1'
DataLength: 1000
DataTs: 1 (intervalo de muestreo)

Por ejemplo:

V = Mod21.EstimationInfo.LossFcn

asigna el valor de LossFcn a la variable V

Para obtener todos los datos que se pueden extraer del objeto modelo en Command Window hacer ***idprops/ENTER***

Ejemplo de Modelo *Mod22* con 2 Entradas

$$y_1(t) = -a_1\, y_1(t-1) - a_2 y(t-2) + b_{11}\, u_1(t-1) + b_{21}\, u_2(t-1) + b_{22} u(t-2) + w(t)$$

En este caso,

DATA = [y1 u1 u2] = z1;
ORDERS = [2 [1 2] [1 1]] = N1;

Mod22=arx(z1,N1).

```
%IDMOD22_6p5_2006.m
% Estimación de parámetros de un sistema dados los datos de dos entradas y de una
salida
% El programa genera datos desde una planta de segundo orden y
% luego los usa para estimar sus parámetros mediante ARX si el ruido es blanco o
ARMAX, si el
% ruido es coloreado.
%Contiene una etapa de validacion con datos dustinatos a los de ajuste
%(entrenamiento) del modelo.
format compact
clear w y1 u1 n
close all

TFin=4000% Largo del registro de satos
sig_n=1 %std del ruido blanco w
a1=-1
a2=0.16
polos=roots([1 -1 0.16])%polos de la planta
b11=2
b21=5
b22=-0.3
ALFA=0.85, %Param. filtro generador de u1
BETA=0.75   %Param. filtro generador de u1
sigu1=0.05
sigu2=0.08
GAMA=0.8;
```

```
c1=0.8;c2=0.5;

%Genera los datos de la planta. En la practica esto equivale a medir datos
%en la planta

y1=zeros(TFin,1);y1sr=zeros(TFin,1);u1=zeros(TFin,1);u2=zeros(TFin,1);w=zeros(TFi
n,1);n=zeros(TFin,1);




for i=3:TFin
   w(i)=randn(1)*sig_n;%w(t) es ruido blanco
   %**************************
   %Aqui se puede elegir entre ruido blanco y ruido coloreado
   % n ruido blanco
   n(i)=w(i);
   % *****************
   % n  ruido coloreado
   %n(i)=w(i)+c1*w(i-1); %ruido coloreadode promedio movil
   %n(i)=w(i)+c1*w(i-1)+c2*w(i-2);ruido coloreado de promedio movil
   %n(i)=GAMA*n(i-1)+(1-GAMA)*3*w(i);%ruido coloreado autoregresivo
   %**************************

   u1(i)=ALFA*u1(i-1)+sqrt(((1+ALFA)/(1-ALFA)))*sigu1*randn(1);%comando
   u2(i)=BETA*u2(i-1)+sqrt(((1+BETA)/(1-BETA)))*sigu2*randn(1);%comando
   y1(i)=-a1*y1(i-1)-a2*y1(i-2)+b11*u1(i-1)+ b21*u2(i-1) +b22*u2(i-2)+n(i);%salida
   y1sr(i)=-a1*y1sr(i-1)-a2*y1sr(i-2)+b11*u1(i-1)+ b21*u2(i-1) +b22*u2(i-2);%salida
sin ruido. En la practica no se dispone de ella.
   %Por supuesto que aqui se usa solo para fines de comparacion
end


grafico=1
if grafico==1

        figure
        plot([y1 5*u1,5*u2])
        title(' salida y1 y entradas u1, u2')
```

```
   xlabel(['a1= ',num2str(a1),'   a2= ',num2str(a2),'   b11= ',num2str(b11),'   b21=
',num2str(b21),'   b22= ',num2str(b22)])
  legend(' y1','5*u1','5*u2')
  axis([100 500 -15 15])
  grid
end
%*********************************************************
```

```
%IDENTIFICACIÓN

%Estimación de Parámetros del Modelo Caso ARX (ruido blanco)

%ORDERS= N1=[2 1 1]% [na nb nk];
%DATA= z1=[y1 u1];%matriz con datos de salida y de entrada

z1=[y1 u1 u2];N1=[2 [1 2] [1 1]];
Mod22=arx(z1,N1);%caso ruido blanco

%Estimación de Parámetros del Modelo Caso ARMAX (ruido coloreado n)

%N1=[2 1 0 1]%  N1=[na nb nc nk]% n ruido blanco
%N1=[2 1 1 1]%  N1=[na nb nc nk] % n ruido coloreado n(i)=w(i)+GAMA*w(i-1)
%N1=[2 1 2 1]%  N1=[na nb nc nk] % n ruido coloreado n(i)=w(i)+c1*w(i-1)+c2*w(i-
2)
%N1=[2 1 4 1]%  N1=[na nb nc nk] % n ruido coloreado n(i)=w(i)+c1*w(i-1)+c2*w(i-
2)
%z1=[y1 u1];%matriz con datos de salida y de entrada
%z1=[y1 u1];Mod22=armax(z1,N1);%caso ruido coloreado

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

%NOTA: En MATLAB 6.5 set(idpoly), indica como acceder a los datos resultantes de
la
%identificacion. En este caso idpoly = Mod22
```

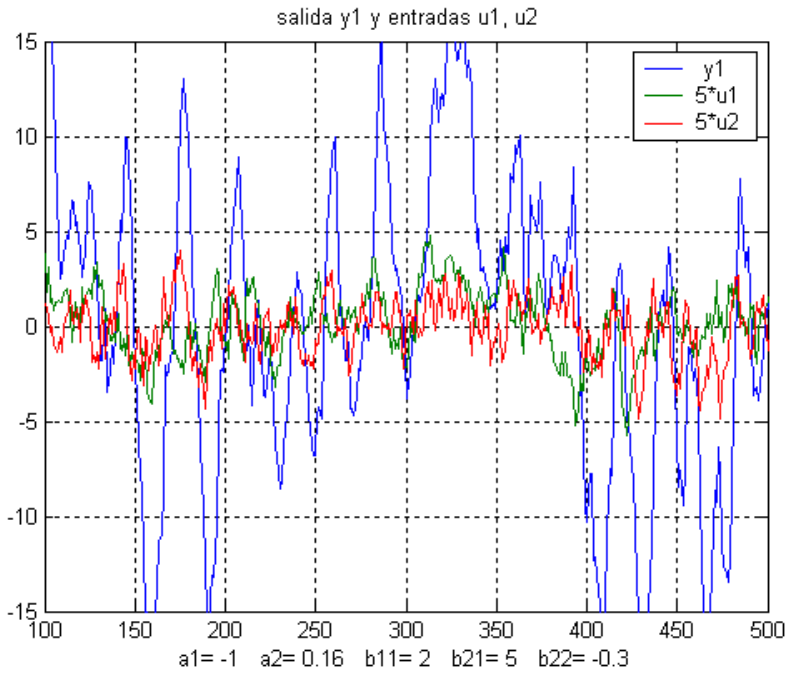
```
ah=Mod22.a %En MATLAB 6.5 entrega el vector de parametros a estimados
bh=Mod22.b %En MATLAB 6.5 entrega el vector de parametros b estimados
ch=Mod22.c %En MATLAB 6.5 entrega el vector de parametros c estimados

a1h=ah(2);  a2h=ah(3);
b11h=bh(1,2)
b21h=bh(2,2)
b22h=bh(2,3)




%En MATLAB 6.5 las desv. std. de determinacion de los parametros se
%obtienen asi:

stda=Mod22.da
stdb=Mod22.db
stdc=Mod22.dc

siga1=stda(2)
siga2=stda(3)
sigb11=stdb(1,2)
sigb21=stdb(2,2)
sigb22=stdb(2,3)

present(Mod22)%Entrega el modelo incluyendo std de los parametros.


%idsim usa el modelo para generar y1 estimado cuando las entradas son u1, u2

y1est=idsim([u1 u2],Mod22);

errest=y1-y1est;
RMSEest= sqrt((errest'*errest)/TFin)

figure
plot([y1 y1est])
legend(' y1',' y1est')
title([' y1 y su estimación y1est como Sensor Virtual  RMSEest= ',num2str(RMSEest)])
ylabel(['  sig_n = ',num2str(sig_n),'      EMCPr1p=',...
```

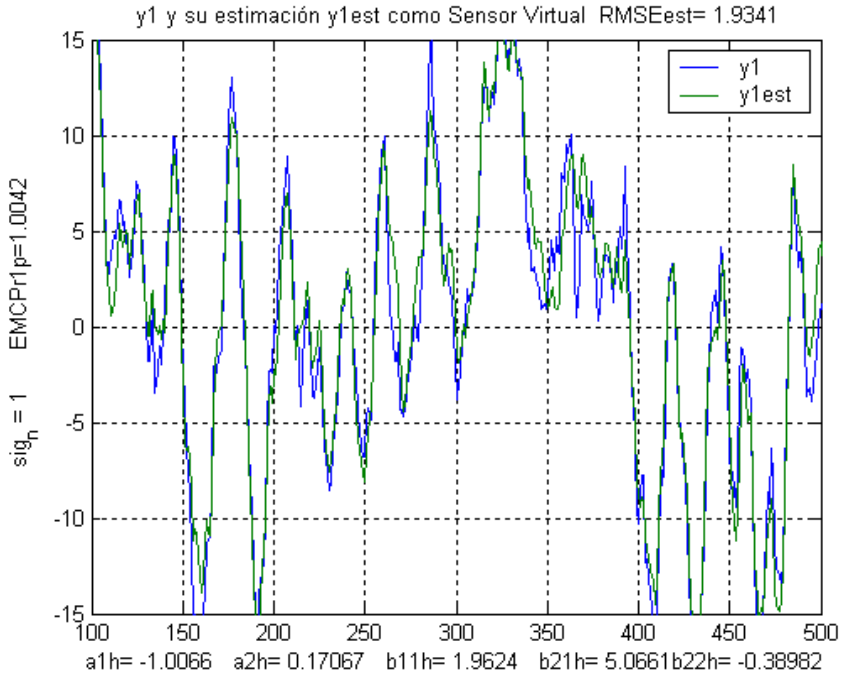
```
    num2str(Mod22.EstimationInfo.LossFcn)])
xlabel(['a1h= ',num2str(a1h),'   a2h= ',num2str(a2h),'   b11h= ',num2str(b11h),'   b21h=
',num2str(b21h),   'b22h= ',num2str(b22h)])
grid
axis([100 500 -15 15])



disp(' ')
disp(' ')
disp('parámetros verdaderos')
disp(' a1          a2          b11      b21      b22')
disp([a1  a2  b11  b21  b22 ])
disp(' ')
disp(' ')

disp('parámetros según arx o armax')
disp(' ')
disp('  a1h      a2h      b11h      b21h      b22h')
disp([a1h  a2h  b11h b21h  b22h])
disp(' ')
disp(' ')


disp('std según arx o armax')
disp(' ')
disp('   siga1      siga2      sigb11      sigb21      sigb22')
disp([siga1  siga2  sigb11 sigb21  sigb22 ])
disp(' ')
disp(' ')




y1p1p=predict(z1,Mod22,1);
figure;plot([y1 y1p1p])
legend('y1','yp1p')
title('Predicción de un paso')
axis([100 500 -15 15])
```

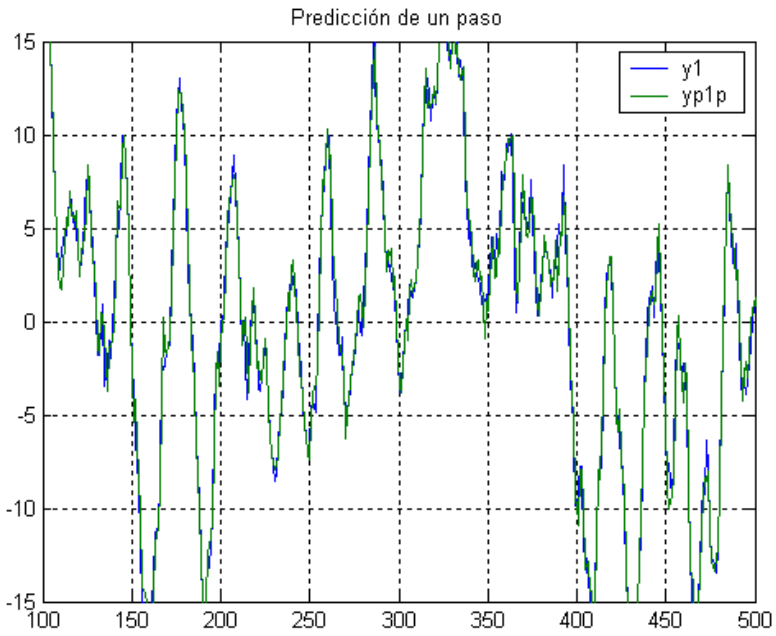
```
grid


ep1p=y1-y1p1p;
Vp1p=var(ep1p)

%VALIDACION

TFV=TFin;
y1v=zeros(TFV,1); u1=zeros(TFV,1);y1vsr=zeros(TFV,1);
GenDat=1;

if GenDat==1
        %Genera nuevos  datos de la planta para validación de modelo
        TFV=TFin;
        for i=3:TFin
    w(i)=randn(1)*sig_n;%w(t) es ruido blanco
    %*************************
    % n ruido blanco
    n(i)=w(i);
    % *****************
    % n  ruido coloreado
    %n(i)=w(i)+c1*w(i-1); %ruido coloreadode promedio movil
    %n(i)=w(i)+c1*w(i-1)+c2*w(i-2);ruido coloreadode promedio movil
    %n(i)=GAMA*n(i-1)+(1-GAMA)*3*w(i);%ruido coloreado autoregresivo
    %*************************
    u1(i)=ALFA*u1(i-1)+sqrt(((1+ALFA)/(1-ALFA)))*sigu1*randn(1);%comando
  u2(i)=BETA*u2(i-1)+sqrt(((1+BETA)/(1-BETA)))*sigu2*randn(1);%comando
  y1v(i)=-a1*y1v(i-1)-a2*y1v(i-2)+b11*u1(i-1)+ b21*u2(i-1) +b22*u2(i-
2)+n(i);%salida
  y1vsr(i)=-a1*y1vsr(i-1)-a2*y1vsr(i-2)+b11*u1(i-1)+ b21*u2(i-1) +b22*u2(i-
2);%salida sin ruido. En la practica no se dispone de ella.
  %Por supuesto que aqui se usa solo para fines de comparacion
   end

end


 y1hval=idsim([u1 u2],Mod22);
 errval=y1v-y1hval;
```

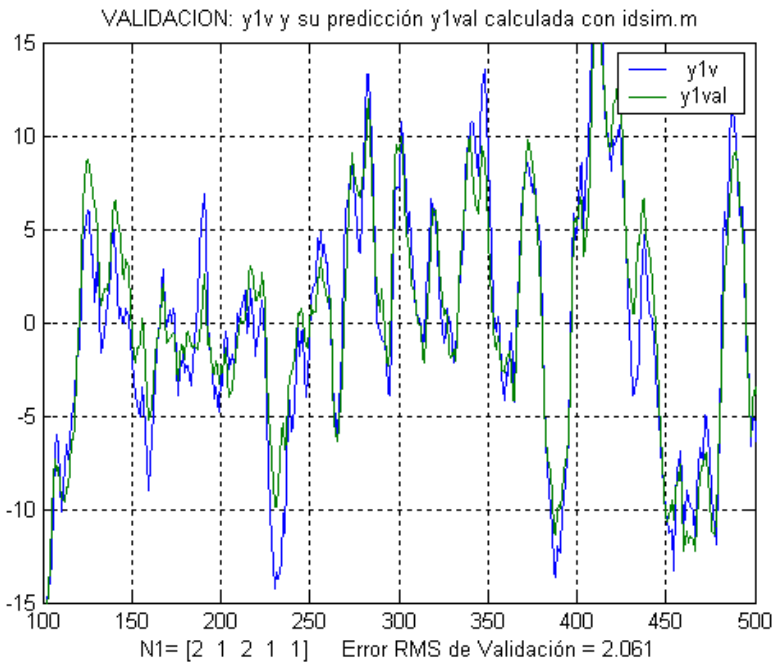
```
 erms_val=sqrt((errval'*errval)/TFin);


figure
plot([y1v y1hval])
legend(' y1v','y1val')
title(' VALIDACION: y1v y su predicción y1val calculada con idsim.m')
axis([100 500 -15 15])
grid
xlabel(['N1= [',num2str(N1),']','     Error RMS de Validación = ',num2str(erms_val)])

%Para obtener todos los datos que se pueden extraer del objeto modelo:
idprops/ENTER')

Mod22.EstimationInfo
V=Mod22.EstimationInfo.LossFcn
FPE=Mod22.EstimationInfo.FPE
par=1:8
AIC=V*(1+2.*par/TFin);

figure;plot(AIC);legend('AIC')
```

Resultados de Mod22






Predicción de un paso
y1
yp1p
15
10
5
0
-5
-10
-15
100
150
200
250
300
350
400
450
500


VALIDACION: y1v y su predicción y1val calculada con idsim.m
y1v
y1val
15
10
5
0
-5
-10
-15
100
150
200
250
300
350
400
450
500
N1= [2 1 2 1 1] Error RMS de Validación = 2.061

```
                                    ah =
    1.0000   -1.0066    0.1707
bh =
         0    1.9624         0
         0    5.0661   -0.3898
ch =
     1

b11h =
    1.9624

b21h =
    5.0661

b22h =
   -0.3898

stda =
         0    0.0145    0.0123

stdb =
         0    0.0555         0
         0    0.0748    0.1108
stdc =
     0
siga1 =
    0.0145

siga2 =
    0.0123

sigb11 =
    0.0555
```

```
sigb21 =
   0.0748

sigb22 =
   0.1108
```

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

A(q) = 1 - 1.007 (+-0.01446) q^-1 + 0.1707 (+-0.0123) q^-2

B1(q) = 1.962 (+-0.05548) q^-1

B2(q) = 5.066 (+-0.0748) q^-1 - 0.3898 (+-0.1108) q^-2

Estimated using ARX from data set z1
Loss function 1.00415 and FPE 1.00667
Sampling interval: 1
Created: 17-Oct-2006 16:18:31
Last modified: 17-Oct-2006 16:18:31

```
RMSEest =
   1.9341
```

parámetros verdaderos

a1	a2	b11	b21	b22
-1.0000	0.1600	2.0000	5.0000	-0.3000

parámetros según arx o armax

a1h	a2h	b11h	b21h	b22h
-1.0066	0.1707	1.9624	5.0661	-0.3898

std según arx o armax

siga1	siga2	sigb11	sigb21	sigb22
0.0145	0.0123	0.0555	0.0748	0.1108

Vp1p =
1.0043

Modelo ARMAX

Ene este caso el ruido es coloreado.

Notación de Matlab y Ljung para Modelos ARMAX

$$A(z^{-1},\theta)y(t) = B(z^{-1},\theta)u(t) + C(z^{-1},\theta)n(t)$$

donde *n(t)* es ruido blanco y

$$A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_{n_a} z^{-n_a}$$
$$B(z^{-1}) = b_1 + b_2 z^{-1} + b_3 z^{-2} + \ldots + b_{n_b} z^{-n_b+1}$$
$$C(z^{-1}) = 1 + c_1 z^{-1} + c_2 z^{-2} + \ldots + c_{n_c} z^{-n_c}$$

Se ve que para el caso de ruido blanco $C(z^{-1}) = 1$.

Como antes, se ha usado la notación de MATLAB para facilitar el uso del TOOLBOX *System Identification.*

En este caso (paramétrico) el modelo está descrito por:

- (i) el orden de los polinomios ***A***, ***B, C***, que definen su estructura y por el retardo n_k de u, y además,
- (ii) por el valor de los parámetros incluidos en el vector $\boldsymbol{\theta}$.

Notación de Matlab y Ljung para modelo general PEM

$$y(t) = \frac{B(z^{-1},\theta)}{A(z^{-1},\theta)}u(t) + \frac{C(z^{-1},\theta)}{D(z^{-1},\theta)}n(t)$$

donde

$$A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_{n_a} z^{-n_a}$$

$$B(z^{-1}) = b_1 + b_2 z^{-1} + b_3 z^{-2} + \ldots + b_{n_b} z^{-n_b+1}$$

$$C(z^{-1}) = 1 + c_1 z^{-1} + c_2 z^{-2} + \ldots + c_{n_c} z^{-n_c}$$

$$D(z^{-1}) = 1 + d_1 z^{-1} + d_2 z^{-2} + \ldots + d_{n_d} z^{-n_d}$$

Se ve que para el caso de ruido blanco $C = D = 1$.