Mínimos Cuadrados ARX

Vídeo en http://personales.upv.es/asala/YT/V/arxml.html (secc. 1)

Vídeo en http://personales.upv.es/asala/YT/V/arxreml.html (secc. 2)

Table of Contents

1.- no recursivo, parte inicial, vídeo [arxml]

Vamos a generar unos datos de un proceso:

```
s=tf('s');
G=9/(s^2+3*s+1); Ts=0.1;
Gd=c2d(G,Ts,'zoh');
Gd
```

```
Gd =

    0.04078 z + 0.0369
  ----------------------
  z^2 - 1.732 z + 0.7408

Sample time: 0.1 seconds
Discrete-time transfer function.
```

```
truenumerador_params=Gd.num{:}(2:end)
```

```
truenumerador_params =
    0.0408    0.0369
```

```
truedenom_params=Gd.den{:}(2:end)
```

```
truedenom_params =
   -1.7322    0.7408
```

```
%generamos datos
Nmuestras=8000;
rng(1234);%siempre mismo ruido, por repetir/comparar diferentes algoritmos
u=randn(Nmuestras,1)*2;
desvtipexcit=0.2;
```

```
ruido=randn(Nmuestras,1)*desvtipexcit;

ruidoarx=1;ruidoaditivo=0;
if(ruidoarx)
  Gd2=[Gd tf(1,Gd.den,Ts)];
  y=lsim(Gd2,[u ruido]);
end
if(ruidoaditivo)
  ylimpia=lsim(Gd,u);
  y=ylimpia+ruido;% añadimos ruido de medida
end
```

Implementación directa no recursiva

```
nb = 2; na = 2; nk = 1;
yd=adddelayedvars(y,na);
%y
%yd
ud=adddelayedvars(u,nb);
size(ud)
```

```
ans =
        7998           3
```

```
RegresorX=[-yd(:,2:end) ud(:,2:end)];
[mm,nn]=size(RegresorX)
```

```
mm = 7998
nn = 4
```

```
ParametrosEstimados=pinv(RegresorX)*yd(:,1)
```

```
ParametrosEstimados =
   -1.7301
    0.7386
    0.0406
    0.0378
```

```
[truedenom_params truenumerador_params] %comparemos con "auténticos"
```

```
ans =
   -1.7322    0.7408    0.0408    0.0369
```

```
residuoencadaecuacion=yd(:,1)-RegresorX*ParametrosEstimados;
VarianzaResiduo=sum(residuoencadaecuacion.^2)/(mm-4)
```

```
VarianzaResiduo = 0.0401
```

```
DesvTipResiduo=sqrt(VarianzaResiduo)
```

```
DesvTipResiduo = 0.2004
```

```
cosa=inv(RegresorX'*RegresorX);
VzaParametrosEstimados=VarianzaResiduo*cosa
```

```
VzaParametrosEstimados =
   1.0e-04 *
    0.3811   -0.3793    0.0013    0.0168
   -0.3793    0.3809   -0.0013   -0.0167
    0.0013   -0.0013    0.0125   -0.0001
    0.0168   -0.0167   -0.0001    0.0133
```

```
DesvTipParametrosEstimados=sqrt(diag(VzaParametrosEstimados))'
```

```
DesvTipParametrosEstimados =
    0.0062    0.0062    0.0011    0.0012
```

Calculemos el modelo identificado no recursivo usando la system ID toolbox

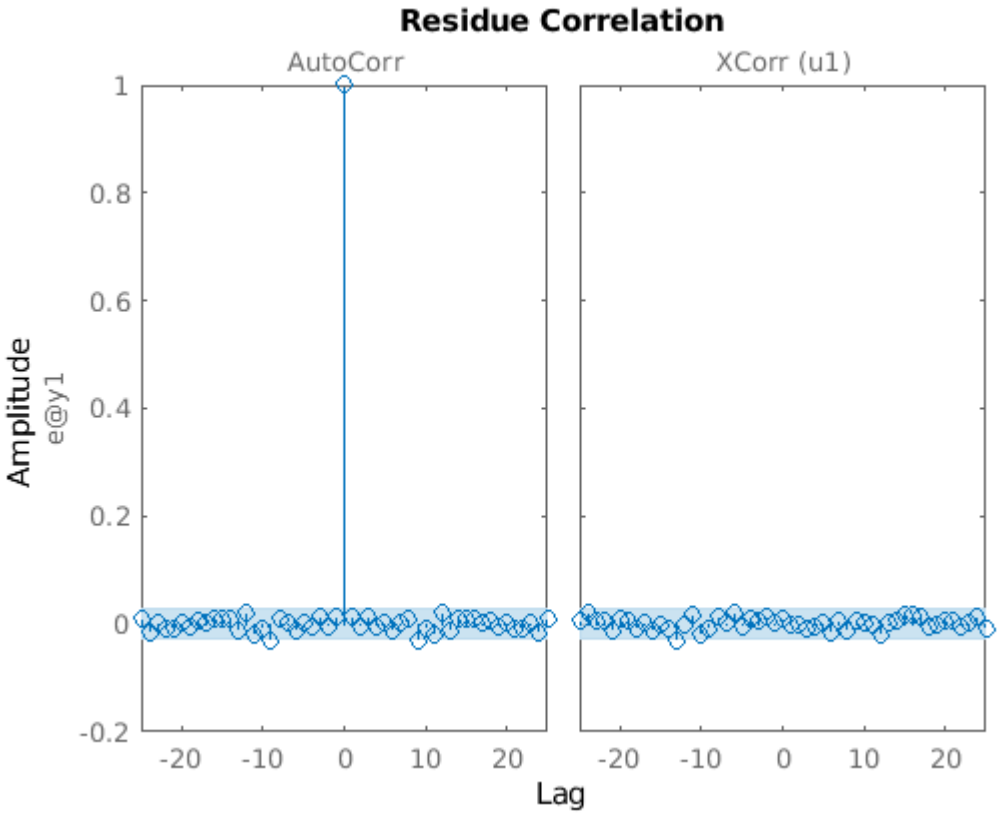
```
todoslosdatos=iddata(y,u,Ts);
th=arx(todoslosdatos,[na nb nk]);
%present(th);
th.cov
```

```
ans =
   1.0e-04 *
    0.3811   -0.3793    0.0013    0.0168
   -0.3793    0.3809   -0.0013   -0.0167
    0.0013   -0.0013    0.0125   -0.0001
    0.0168   -0.0167   -0.0001    0.0133
```

```
th.NoiseVariance
```

```
ans = 0.0401
```

```
resid(todoslosdatos,th)
```




```
%para gráficas de luego.
dsvtipparams=sqrt(diag(th.cov))';
indiceNUM=[3 4]; indiceDEN=[1 2];
norecursivoparsnumerador=th.b(2:end);
norecursivoparsdenominador=th.a(2:end);
dsvtipnum=dsvtipparams(indiceNUM);
dsvtipden=dsvtipparams(indiceDEN);
```

2.- Recursivo, vídeo [arxreml]

Implementación recursiva

Create a System object for online estimation of model with known initial polynomial coefficients.

```
num_inicial = [0 0.03 0.03]; den_inicial = [1 -1.8 0.9];
estimador = recursiveARX([na nb nk],den_inicial,num_inicial);
estimador.InitialParameterCovariance = [2 2 2 2].^2;
estimador.ForgettingFactor = 1;%-5e-3;

plotmedia=zeros(4,Nmuestras);
plotIntervConf=zeros(8,Nmuestras);
for i=1:Nmuestras
  [p1,p2,yhat]=estimador(y(i),u(i));

  %guardemos datos para gráficas posteriores
  covv=estimador.ParameterCovariance;
  %parameter covariance se calcula suponiendo
  % que el resíduo (ruido de proceso arx) es de varianza 1.
  %corregimos:
  covv=covv*desvtipexcit^2;
  %esto es complicado de saber en proceso práctico! pero la covarianza se usa poco.
  desvtippars=sqrt(diag(covv));
  xa=[p1(2:end)';p2(2:end)'];%el primero es 0 en num, 1 en den, no lo guardamos.
  plotmedia(:,i)=xa;  plotIntervConf(:,i)=[xa+2*desvtippars; xa-2*desvtippars];
end
```

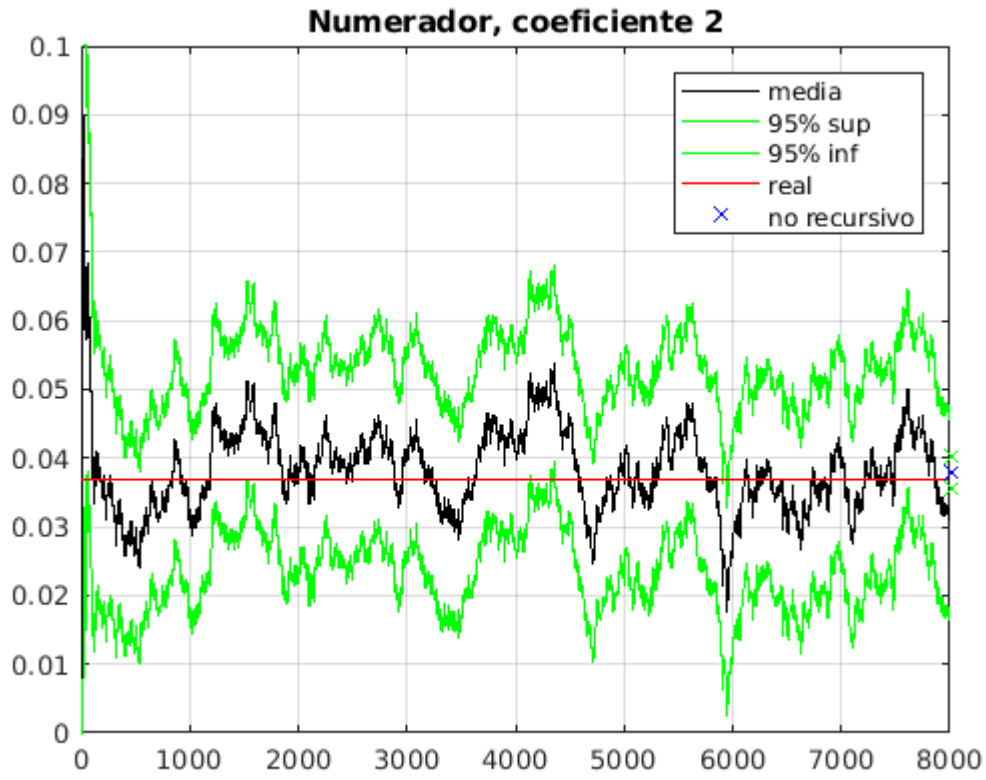
Graficas de resultados

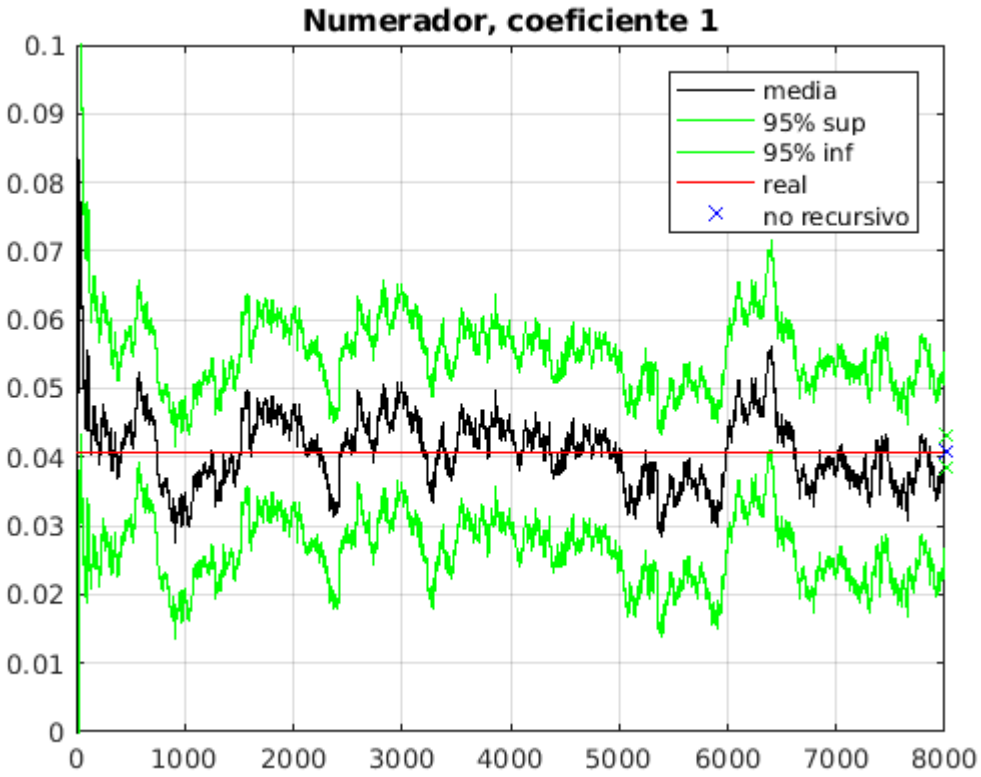
Dibujemos los parámetros estimados y su intervalo de conf...

NUMERADOR

```
%dibujemos estimado de parámetros del numerador
for coefic=1:2
    figure()
    plot(plotmedia(indiceNUM(coefic),:)','k')
    hold on
    plot(plotIntervConf([indiceNUM(coefic) indiceNUM(coefic)+4],:)','g')
    plot(ones(Nmuestras,1)*truenumerador_params(coefic),'r')

    %plot del resultado NO recursivo
    plot(Nmuestras,norecursivoparsnumerador(coefic),'xb')
    plot(Nmuestras,norecursivoparsnumerador(coefic)+dsvtipnum(coefic)*2,'xg')
    plot(Nmuestras,norecursivoparsnumerador(coefic)-dsvtipnum(coefic)*2,'xg')
```

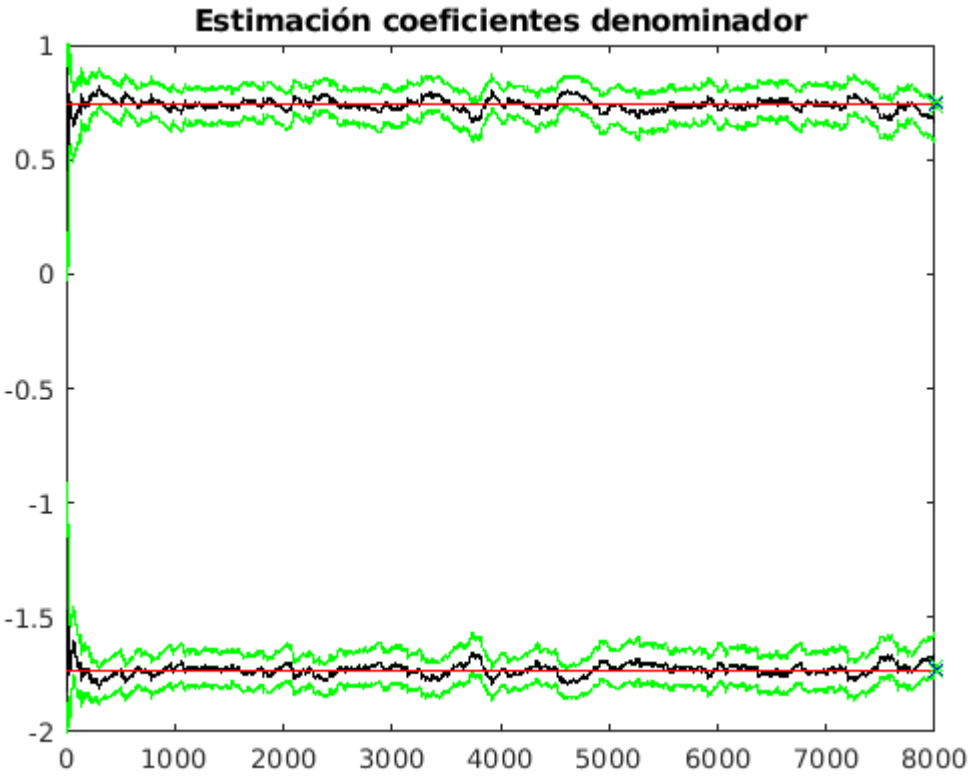
```
    hold off
    axis([0 Nmuestras -.0 .1])
    legend('media','95% sup','95% inf','real','no recursivo')
    title(sprintf('Numerador, coeficiente %d',coefic))
    grid on

end
```





Denominador:

```
plot(plotmedia(indiceDEN,:)','k')
hold on
plot(plotIntervConf([indiceDEN indiceDEN+4],:)','g')
plot(ones(Nmuestras,2)*diag(truedenom_params),'r')
plot(Nmuestras,norecursivoparsdenominador,'xb')
    plot(Nmuestras,(norecursivoparsdenominador+dsvtipden*2)','xg')
    plot(Nmuestras,(norecursivoparsdenominador-dsvtipden*2)','xg')
hold off
title('Estimación coeficientes denominador')
axis([0 Nmuestras -2 1.])
```


Estimación coeficientes denominador
1
0.5
0
-0.5
-1
-1.5
-2
0
1000
2000
3000
4000
5000
6000
7000
8000