Identificación "no paramétrica" de respuestas ante impulso de un sistema lineal invariante en el tiempo: regularización.

Presentación en vídeo: http://personales.upv.es/asala/YT/V/impulidreg.html

Este código funcionó correctamente con Matlab **R2020b**

Objetivo: Identificar la respuesta ante impulso (la respuesta ante la entrada $\{1,0,0,\dots\}$, esto es, una cierta secuencia $\{h_0,h_1,h_2,\dots\}$) de un sistema lineal discreto invariante en el tiempo, usando la fórmula de convolución. Comprobar, mediante datos de validación, la longitud de respuesta recomendable para el método (**regularización**).

Tabla de Contenidos

1.- Modelado y simulación para conseguir datos de entrenamiento/validación..... 1
2.- Identificación de respuesta impulsional por mínimos cuadrados sin regularizar.....3
3a.- Regularización (selección de número de elementos de la respuesta a estimar)..... 6
1.- Por error sobre datos de validación..... 6
2.- Por criterio de información de Akaike..... 7
3b.- Regularización (ponderar hacia "cero" los parámetros, ridge regression)..... 9
Conclusiones..... 11

1.- Modelado y simulación para conseguir datos de entrenamiento/ validación

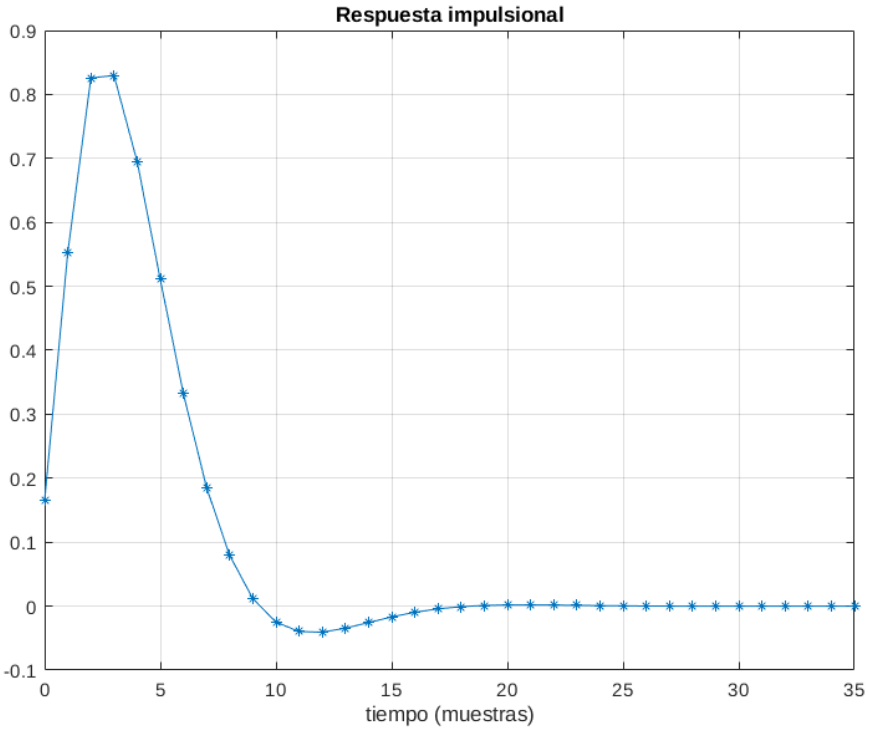
```
[b,a]=butter(2,.15);
G=tf(b*4,a,1)
```

```
G =

  0.165 z^2 + 0.33 z + 0.165
  --------------------------
    z^2 - 1.349 z + 0.514

Sample time: 1 seconds
Discrete-time transfer function.
```

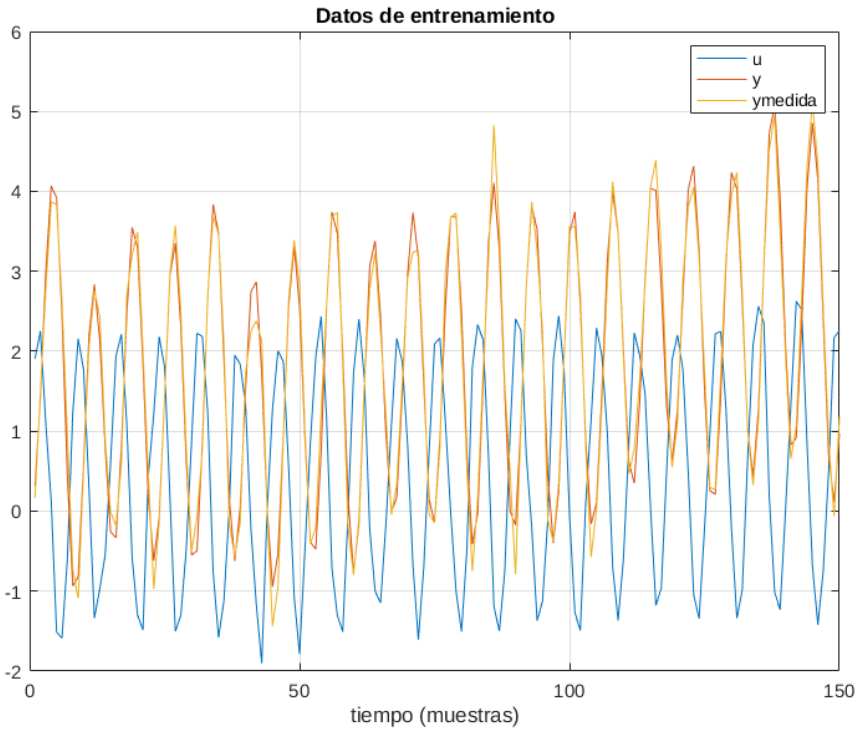
Calculemos la respuesta ante un impulso $\{u_k\}=\{1,0,0,\dots\}$:

```
orden=36;
thperfecto=impulse(G,orden-1);
```

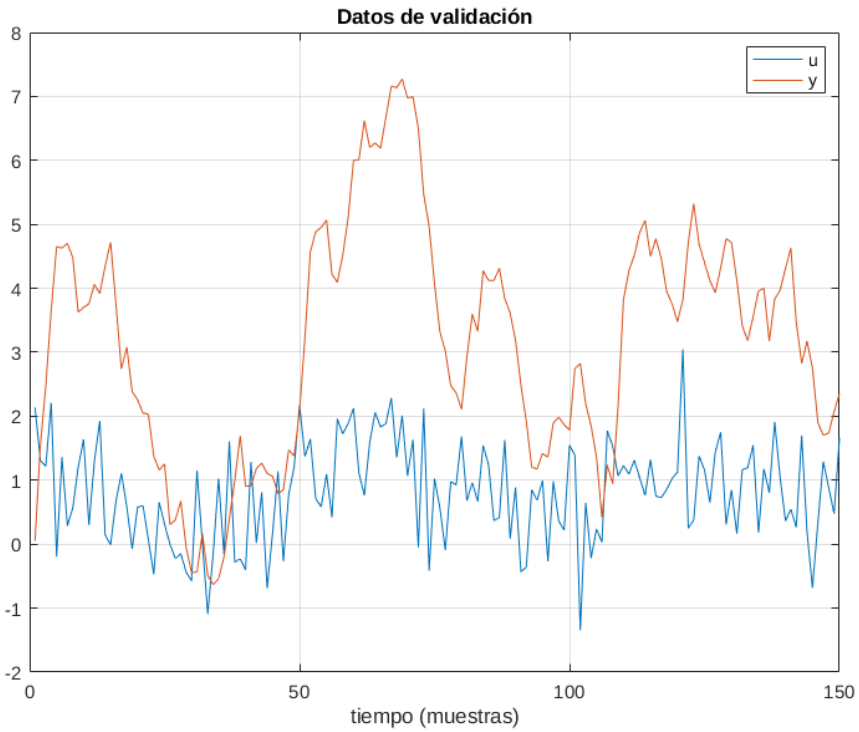
```
ejeTh=0:(orden-1);
plot(ejeTh,thperfecto,'Marker','*'), grid on, title("Respuesta impulsional"), xlabel("t
```




```
N=150;
u=1.9*sin(0.85*(1:N))'+0.2*randn(N,1)+0.3*(1:N)'/N+0.3;
uvalid=0.5*cos(0.1*(1:N))'+0.75*randn(N,1)+0.7;
dtruido=0.25;
ylimpia=lsim(G,u);
y=ylimpia+dtruido*randn(N,1);
yvalid=lsim(G,uvalid)+dtruido*randn(N,1);
plot([u ylimpia y]), legend('u','y','ymedida'), grid on,
title("Datos de entrenamiento"), xlabel("tiempo (muestras)")
```




```
plot([uvalid yvalid]), legend('u','y'), grid on, title("Datos de validación"), xlabel('
```



2.- Identificación de respuesta impulsional por mínimos cuadrados sin regularizar

Si la respuesta ante $\{1,0,0,\dots\}$ es $\{h_0,h_1,h_2,\dots\}$, denominada *respuesta impulsional*, entonces, por linealidad (superposición), la respuesta ante $\{u_0,u_1,u_2,u_3,\dots\}$y condiciones iniciales nulas es $\{y_0=u_0h_0, y_1=u_0h_1+u_1h_0, y_2=u_0h_2+u_1h_1+u_2h_0, y_3=u_0h_3+u_1h_2+u_2h_1+u_3h_0,\dots\}$, que se denomina **fórmula de convolución discreta**.

Escribiendo la última expresión en forma de matriz:

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \end{pmatrix} = \begin{pmatrix} u_0 & 0 & 0 & \dots & 0 \\ u_1 & u_0 & 0 & \dots & 0 \\ u_2 & u_1 & u_0 & \dots & 0 \\ \vdots & & \vdots & & \vdots \end{pmatrix} \begin{pmatrix} h_0 \\ h_1 \\ h_2 \\ \vdots \end{pmatrix} = T \cdot \mathbf{h}$$

ello permite estimar $\mathbf{h}$ dados datos de entrada $\{u_0,u_1,u_2,u_3,\dots\}$ y salida $\{y_0,y_1,y_2,y_3,\dots\}$ por mínimos cuadrados. Como en un sistema estable los elementos de la respuesta impulsional tienden a cero, se limita el número de elementos de $\mathbf{h}$ (y de columnas de T) a un tamaño "razonable" N.

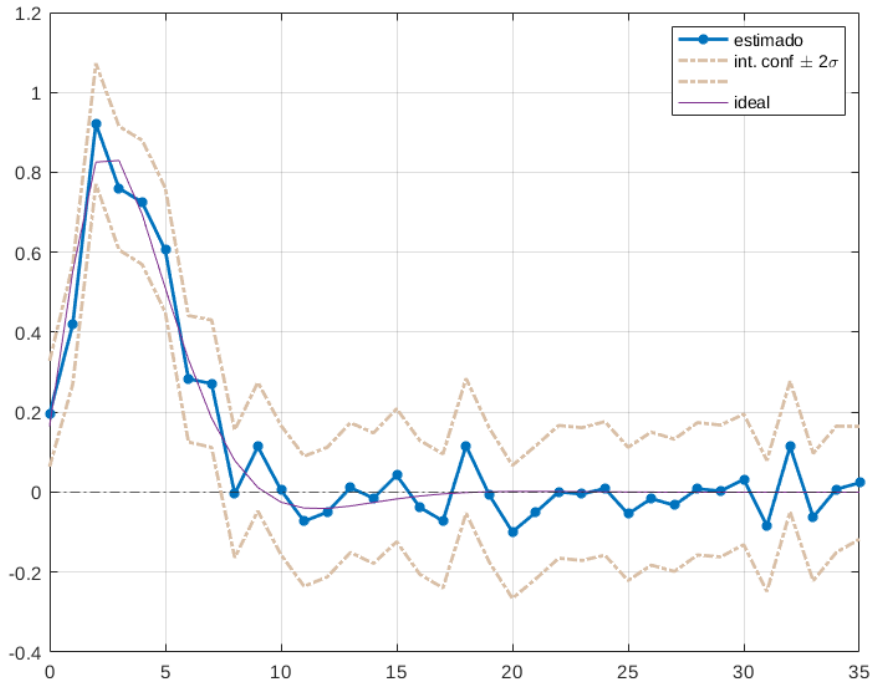
```
T=toeplitz(u,[u(1) zeros(1,orden-1)])
```

```
T = 150×36
    1.9036         0         0         0         0         0         0         0 ···
    2.2543    1.9036         0         0         0         0         0         0
    1.0960    2.2543    1.9036         0         0         0         0         0
    0.1320    1.0960    2.2543    1.9036         0         0         0         0
   -1.5138    0.1320    1.0960    2.2543    1.9036         0         0         0
   -1.5868   -1.5138    0.1320    1.0960    2.2543    1.9036         0         0
   -0.5921   -1.5868   -1.5138    0.1320    1.0960    2.2543    1.9036         0
    1.2329   -0.5921   -1.5868   -1.5138    0.1320    1.0960    2.2543    1.9036
    2.1581    1.2329   -0.5921   -1.5868   -1.5138    0.1320    1.0960    2.2543
    1.7726    2.1581    1.2329   -0.5921   -1.5868   -1.5138    0.1320    1.0960
     ⋮
```

```
th_noreg=pinv(T)*y;
```

Este es, obviamente, el modelo que mejor ajusta los datos de *entrenamiento* (en términos de error cuadrático)

```
std_err=sqrt(sum((y-T*th_noreg).^2/(N-orden))); %desv. típica error
varth=std_err^2*inv(T'*T);
std_th=sqrt(diag(varth));
plotIncierto(ejeTh,th_noreg,std_th)
hold on
plot(ejeTh,thperfecto)
hold off, yline(0,'-.')
legend('estimado','int. conf \pm 2\sigma','','ideal')
```



El error sobre los datos de validación será:

```
Tvalid=toeplitz(uvalid,[uvalid(1) zeros(1,orden-1)]);
err_valid_noregularizado=sum((yvalid-Tvalid*th_noreg).^2)% error validación
```

```
err_valid_noregularizado = 17.7877
```

Pruebo con ARX:

```
zz=zeros(orden,1);
modelo=arx(iddata([zz; y],[zz; u],1),[0 orden 0],arxOptions('InitialCondition','zero'))
```

```
modelo =
Discrete-time FIR model: y(t) =  B(z)u(t) + e(t)

  B(z) = 0.1959 + 0.42 z^-1 + 0.9217 z^-2 + 0.7607 z^-3 + 0.725 z^-4 + 0.6056 z^-5 + 0.2837 z^-6 + 0.2716
         - 0.07265 z^-11 - 0.04973 z^-12 + 0.0114 z^-13 - 0.01483 z^-14 + 0.0425 z^-15 - 0.03792 z^-16 -
         -19 - 0.0992 z^-20 - 0.05089 z^-21 + 0.001039 z^-22 - 0.004627 z^-23 + 0.009443 z^-24 - 0.05429
                                              + 0.003115 z^-29 + 0.03214 z^-30 - 0.08435 z^-31 + 0.1146 z^-3

Sample time: 1 seconds

Parameterization:
   Polynomial orders:   nb=36   nk=0
   Number of free coefficients: 36
   Use "polydata", "getpvec", "getcov" for parameters and their uncertainties.

Status:
Estimated using ARX on time domain data.
Fit to estimation data: 87.73%
FPE: 0.08454, MSE: 0.03783
```

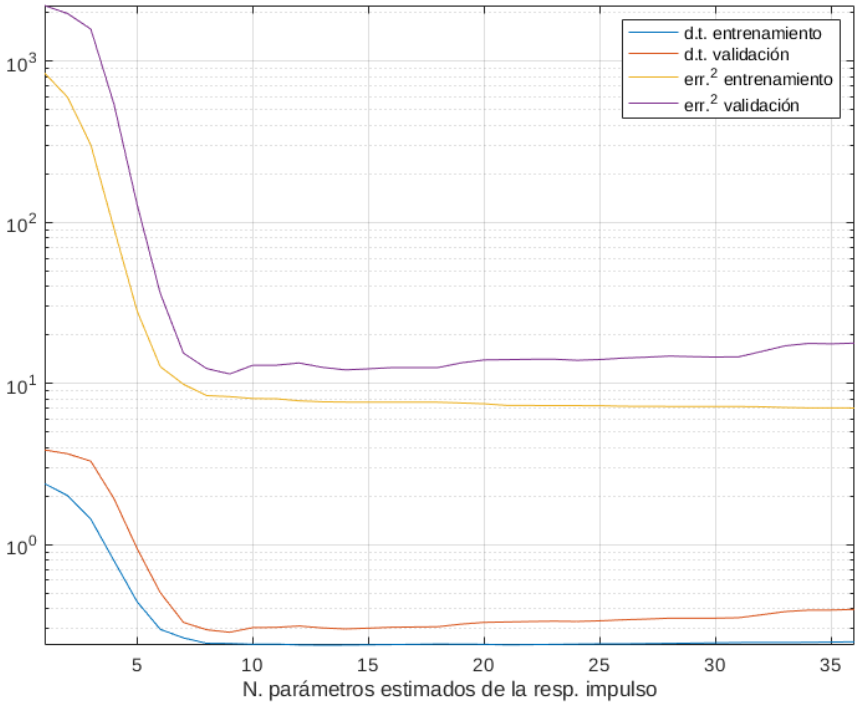
3a.- Regularización (selección de número de elementos de la respuesta a estimar)

1.- Por error sobre datos de validación

Hay muchos elementos de la respuesta ante impulso cuyo intervalo de confianza toca el cero: puede que sean "verdaderamente cero" o al menos "no hay suficientes datos para asegurar que no sean cero". Si estimamos menos parámetros, tendremos menos varianza (pero más "sesgo").

Limitaremos el número de parámetros a estimar, viendo el ajuste sobre datos de validación:

```
ord_max=36;
dt_test=zeros(1,ord_max);
dt_val=zeros(1,ord_max);
error_test=zeros(1,ord_max);
error_val=zeros(1,ord_max);
akaikeIC=zeros(1,ord_max);
for orden=1:ord_max
    %hacemos regresión
    T=toeplitz(u,[u(1) zeros(1,orden-1)]);
    Tvalid=toeplitz(uvalid,[uvalid(1) zeros(1,orden-1)]);
    thv=pinv(T)*y;
    %determinamos estadísticos del ajuste
    error_test(orden)=sum((y-T*thv).^2);
    error_val(orden)=sum((yvalid-Tvalid*thv).^2);
    dt_test(orden)=sqrt(error_test(orden)/(N-orden)); %desv. típica error validación
    dt_val(orden)=sqrt(error_val(orden)/(N-orden)); %desv. típica error validación
    akaikeIC(orden)=AkaikeInformationCrit(error_test(orden),N,orden); %crit. akaike.
end
semilogy(1:ord_max,[dt_test;dt_val;error_test;error_val]), grid on, axis tight
legend("d.t. entrenamiento","d.t. validación","err.^2 entrenamiento","err.^2 validación
xlabel("N. parámetros estimados de la resp. impulso")
```



2.- Por criterio de información de Akaike

Este criterio no necesita datos de validación. Equilibra núm. de parámetros con capacidad de reducir error de ajuste sobre entrenamiento. Grosso modo, añadir un parámetro más debe mejorar el error cuadrático en un factor $e^{-2/N}$ para merecer la pena.

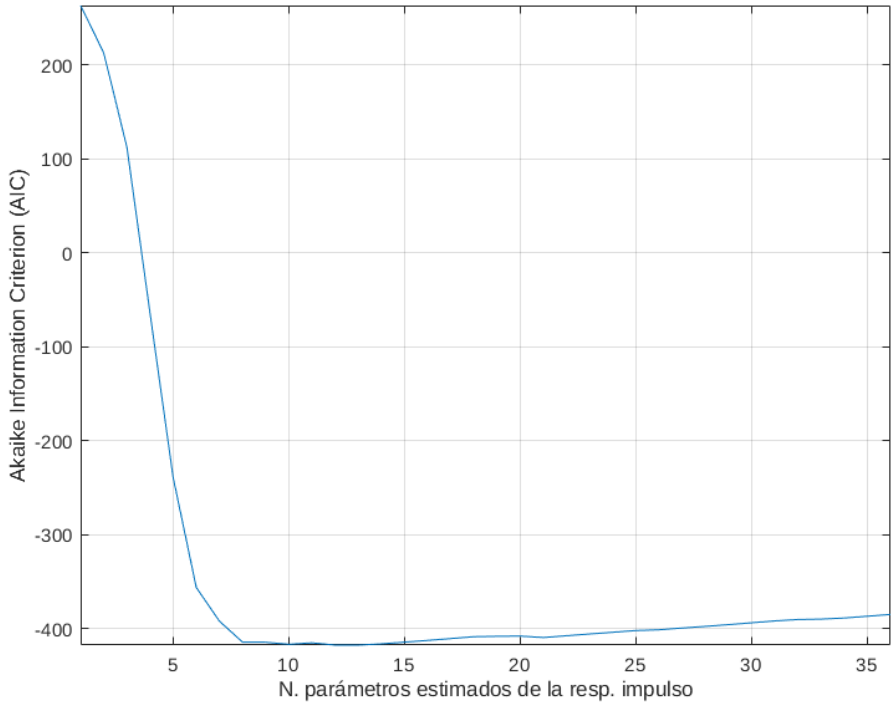
```
exp(-2/N)
```

```
ans = 0.9868
```

```
1-ans
```

```
ans = 0.0132
```

```
plot(1:ord_max,akaikeIC), grid on, axis tight
%ylabel("log error"), legend("d.t. entrenamiento","d.t. validación","err.^2 entrenamier
xlabel("N. parámetros estimados de la resp. impulso"), ylabel("Akaike Information Crite
```



Los datos parecen indicar que con un numero de parámetros de aproximadamente ocho o nueve tenemos un buen compromiso...

```
[~,orden]=min(dt_val)
```

```
orden = 9
```

```
T=toeplitz(u,[u(1) zeros(1,orden-1)]);
Tvalid=toeplitz(uvalid,[uvalid(1) zeros(1,orden-1)]);
th=pinv(T)*y;
error_test=sum((y-T*th).^2)
```

```
error_test = 8.2817
```

```
std_err=sqrt(error_test/(N-orden)) %desv. típ. error
```

```
std_err = 0.2424
```

```
error_val=sum((yvalid-Tvalid*th).^2)
```

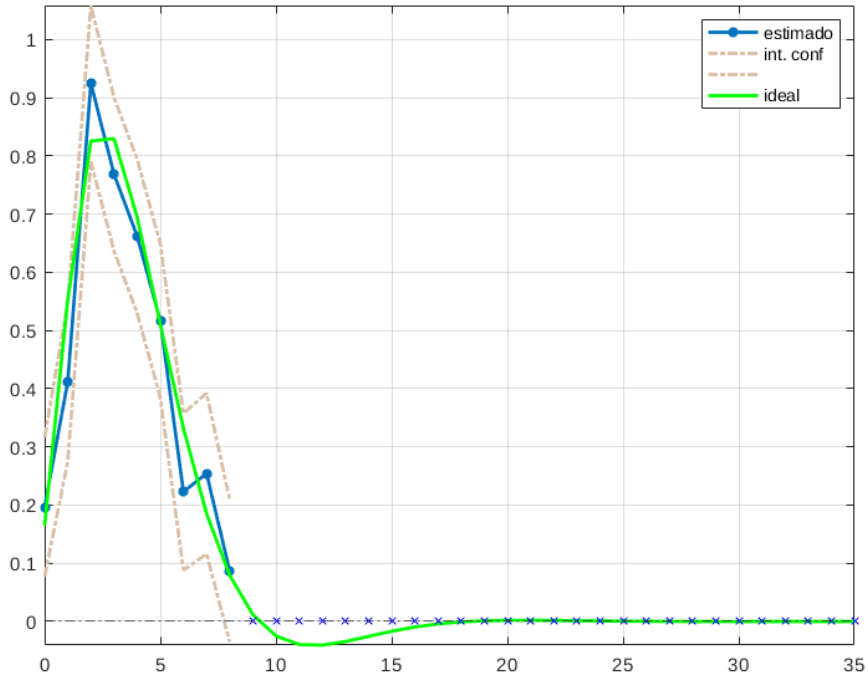
```
error_val = 11.4433
```

```
varth=std_err^2*inv(T'*T);
std_th=sqrt(diag(varth));
max(std_th)
```

```
ans = 0.0690
```

```
plotIncierto(0:(orden-1),th,std_th)
hold on
```

```
L=length(thperfecto);
plot(ejeTh,thperfecto,'g','LineWidth',2)
plot(orden:(L-1),zeros(L-orden,1),'xb')
hold off, yline(0,'-.'), axis tight
legend('estimado','int. conf','','ideal')
```



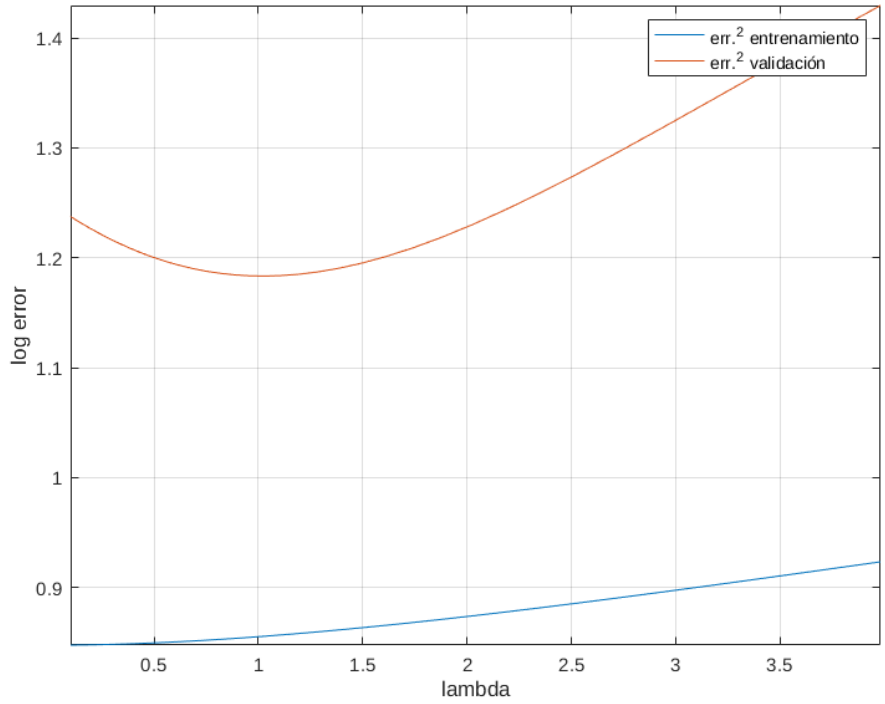
3b.- Regularización (ponderar hacia "cero" los parámetros, ridge regression)

Resolveremos ahora el problema "ridge regression" de minimizar $J = \|y - T \cdot \theta\|_2^2 + \lambda \cdot \|\theta\|_2^2$, de modo que los parámetros "poco útiles" para minimizar error son "empujados" hacia cero. El resultado es

$$\widehat{\theta} = (T^T T + \lambda I)^{-1} \cdot T^T \cdot y.$$

```
orden=36;
error_test=zeros(1,ord_max);
error_val=zeros(1,ord_max);
Ll=50;%Num puntos a explorar
lambdarg=logspace(-1,0.6,Ll);
T=toeplitz(u,[u(1) zeros(1,orden-1)]);
Tvalid=toeplitz(uvalid,[uvalid(1) zeros(1,orden-1)]);
for i=1:Ll
    thv=(T'*T+lambdarg(i)*eye(orden))\(T'*y);
    error_test(i)=sum((y-T*thv).^2);
    error_val(i)=sum((yvalid-Tvalid*thv).^2);
```

```
end
plot(lambdarg,log10([error_test;error_val])), grid on, axis tight
ylabel("log error"), legend("err.^2 entrenamiento","err.^2 validación"), xlabel("lambda
```




```
[~,iopt]=min(error_val)
```

```
iopt = 32
```

```
lamopt=lambdarg(iopt)
```

```
lamopt = 1.0286
```

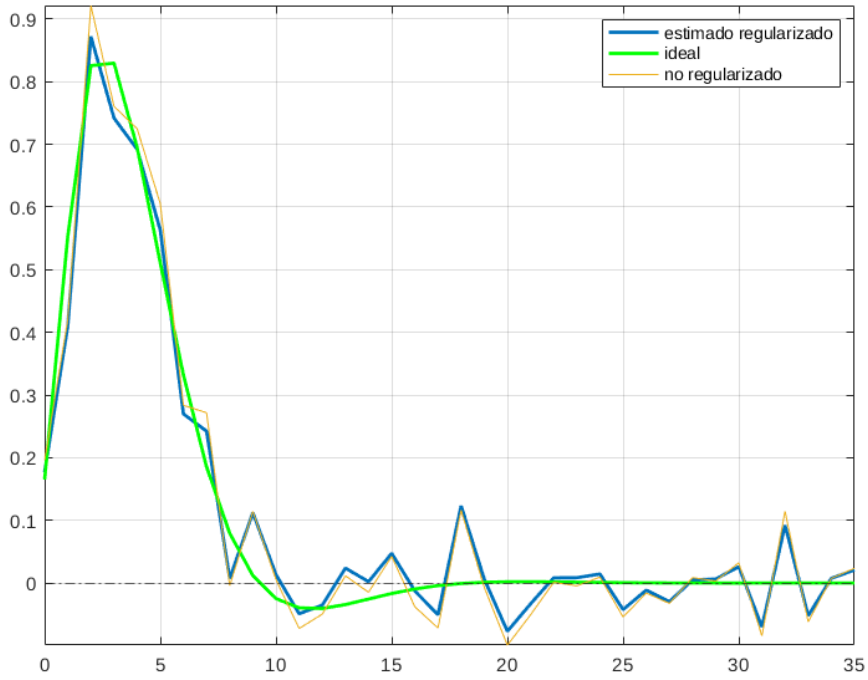
```
error_test(iopt)
```

```
ans = 7.1700
```

```
error_val(iopt)
```

```
ans = 15.2523
```

```
thL=(T'*T+lamopt*eye(orden))\(T'*y);
plot(ejeTh,thL,'LineWidth',2)
hold on
L=length(thperfecto);
plot(ejeTh,thperfecto,'g','LineWidth',2)
plot(ejeTh,th_noreg)
hold off, yline(0,'-.'), axis tight, grid on
legend('estimado regularizado','ideal','no regularizado')
```



Conclusiones

Aunque la identificación directa de la respuesta impulsional (la respuesta ante la entrada $\{1, 0, 0, \dots\}$, una secuencia $\{h_0, h_1, h_2, \dots\}$) basada en datos de entrada-salida es usualmente catalogada como un método "no paramétrico" en muchos libros, en este caso utiliza las mismas fórmulas de mínimos cuadrados clásicos: los elementos de la respuesta impulsional pueden ser considerados como los parámetros a estimar.

La respuesta tiene teóricamente "duración infinita" pero, como cuantos más elementos de la misma se estiman mayor es la varianza del estimado debida al ruido, debe "regularizarse", por ejemplo comprobando el ajuste ante datos de validación independientes de los de entrenamiento, para determinar el número razonable de parámetros que podría ser identificado con confianza en una situación práctica (dependiendo del número y calidad de los datos).

```
function plotIncierto(X,Y,DesvTip)
plot(X,Y,'LineWidth',2,'Marker','*')
hold on
plot(X,Y+2*DesvTip,'-.','Color',[.85 .75 .65],'LineWidth',2)
plot(X,Y-2*DesvTip,'-.','Color',[.85 .75 .65],'LineWidth',2)
```

```
hold off, grid on
end

function AIC=AkaikeInformationCrit(err,N,numpars)
%err es el error cuadrático, N el número de datos, y numpars el número de
%parámetros estimado (la fórmula añade 1 porque también se estima la
%"varianza del error de predicción", resulta:
vzaerr=err/N;
AIC=N*log(vzaerr)+2*(numpars+1);
%https://projects.ncsu.edu/crsc//reports/ftp/pdf/crsc-tr17-09.pdf
end
```