# Identificación experimental de respuestas escalón no oscilatorias como procesos de primer orden



Presentaciones en vídeo:

--- http://personales.upv.es/asala/YT/V/ord1teo.html

--- http://personales.upv.es/asala/YT/V/ord1exg.html

--- http://personales.upv.es/asala/YT/V/ord1exmal.html

Este código funcionó sin errores con Matlab **R2022b** (Linux)

**Objetivo:** comprender la **teoría** y la identificación **experimental** rápida con el "criterio del 63%" de sistemas de 1er orden ante escalón... con ejemplos de cuando *funciona* y cuando *no funciona* tan bien porque el proceso físico no es "lineal de 1er orden".

**Tabla de Contenidos**



## Revisión teórica

### Respuesta escalón de sistemas de 1er orden

Un sistema de primer orden lineal dado por la EDO:

$$\tau\frac{dy}{dt}+y=K\cdot u$$

con condiciones iniciales nulas, representado en el dominio de Laplace con $Y(s)=G(s)U(s)$, siendo

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

su función de transferencia, tiene una respuesta temporal ante $u(t) = A$, o sea $U(s) = A \cdot \frac{1}{s}$ en dominio de Laplace, dada por la fórmula:

$$y(s) = \frac{K \cdot A}{s \cdot (\tau s + 1)} = \frac{A \cdot K/\tau}{s \cdot (s + 1/\tau)} = \frac{A \cdot K}{s} - \frac{A \cdot K}{s + 1/\tau}$$

$$y(t) = A \cdot K - A \cdot K \cdot e^{-\frac{1}{\tau}t} = A \cdot K \cdot \left(1 - e^{-\frac{1}{\tau}t}\right)$$

En concreto:

$$\text{Valor final} = \lim_{t \to \infty} y(t) = A \cdot K$$

$$y(\tau) = A \cdot K \cdot (1 - e^{-1}), \qquad y(3\tau) = A \cdot K \cdot (1 - e^{-3}), \qquad y(4\tau) = A \cdot K \cdot (1 - e^{-4})$$

y sustituyendo valores numéricos de la exponencial,

```
[1-exp(-1), 1-exp(-3), 1-exp(-4)]
```

```
ans = 1×3
    0.6321    0.9502    0.9817
```

queda:

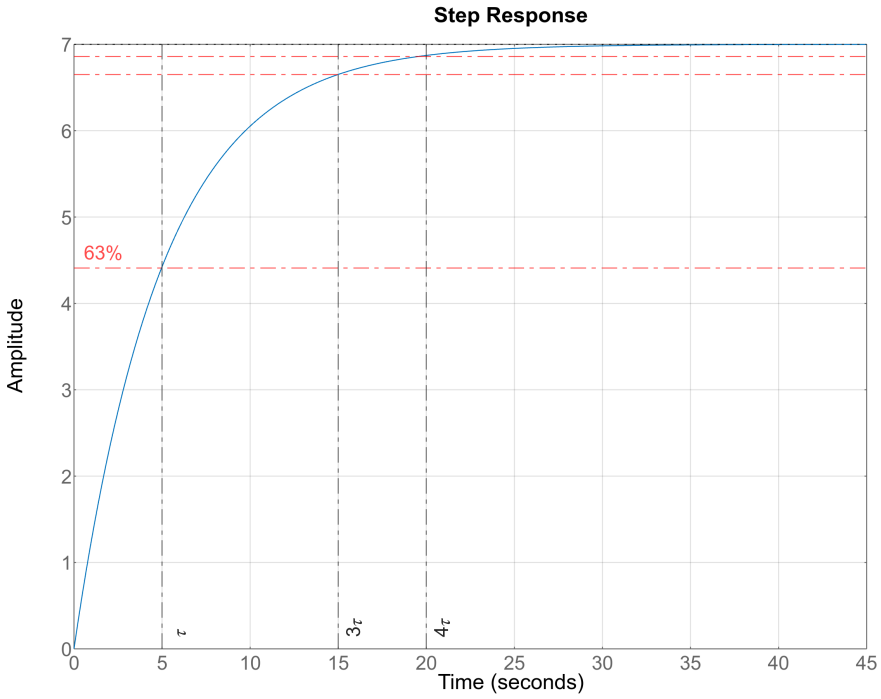
$$\text{Valor final} = \lim_{t \to \infty} y(t) = A \cdot K$$

$$y(\tau) \approx 0.63 \cdot A \cdot K, \qquad y(3\tau) \approx 0.95 \cdot A \cdot K, \qquad y(4\tau) \approx 0.98 \cdot A \cdot K$$

**NOTA:** teóricamente, el valor final se alcanza en "infinito" tiempo... pero en la práctica, las cosas se paran: un coche en un semáforo acaba parado "del todo" en segundos, nuestra habitación "grosso modo" se calienta en 40 minutos al encender la calefacción, la batería se carga en 1 hora, etc... Decir que el transitorio tarda "*aproximadamente tres o cuatro veces* $\tau$" en terminar es más cercano a nuestro "*feeling en la práctica*" que el pensar en los límites $t \to \infty$ de las fórmulas que es lo que "en rigor" habría que hacer. Los sistemas "prácticos" no tardan "infinito" en pararse, o al menos no para mis sentidos que son incapaces de discernir subidas extra de 0.2 grados de temperatura... una vez estamos al 95% o 98% del total, usualmente no hay diferencia "perceptible" con el 100% $A \cdot K$. Por eso la "conveniencia práctica" de lo de arriba.

**Ejemplo:**

```
s=tf('s');
step(7/(5*s+1)), grid on %A=1   K=7, tau=5
RayasHoriz(7)
TresRayasVert(5)
```




*Trivialmente, si me dan el sistema en la forma $\frac{dy}{dt} + ay = bu, \quad G(s) = \frac{b}{s+a}$ lo puedo escribir como

$$\frac{1}{a} \cdot \frac{dy}{dt} + y = \frac{b}{a} u, \qquad G(s) = \frac{b/a}{\frac{1}{a} \cdot s + 1}$$

con lo que con $K = b/a$, $\tau = 1/a$ podría aplicar lo de arriba.

### Sistemas 1er orden con retardo

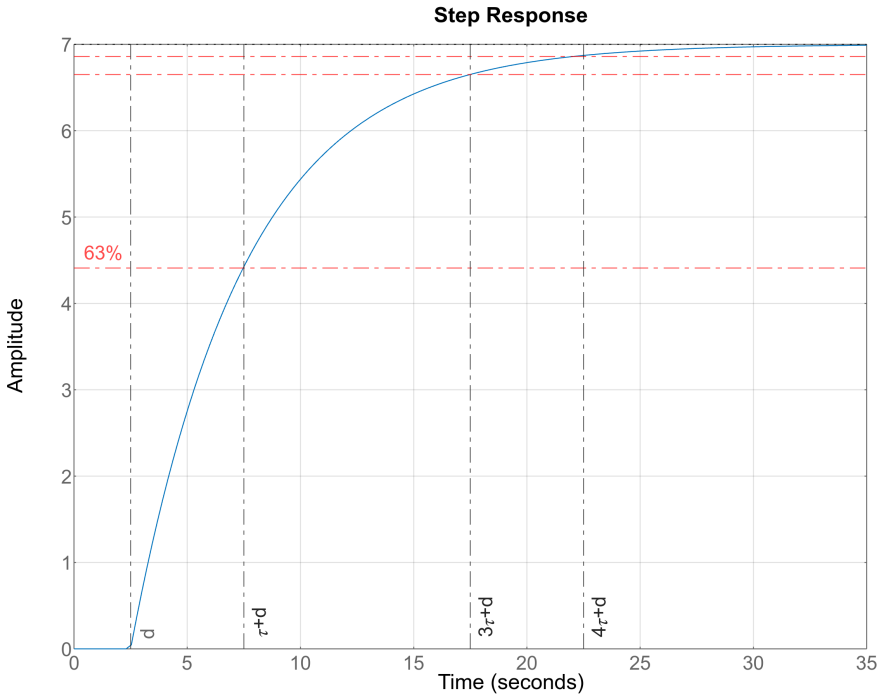
Si el sistema tiene retraso de $d$ segundos (por ejemplo, transporte de masa o energía en tuberías en proceso químico, retardo de cómputo o de comunicación por red,...) entonces todo ocurre "$d$" segundos más tarde:

$$\tau \frac{dy}{dt} + y(t) = K \cdot u(t-d), \qquad G(s) = \frac{K}{\tau s + 1} \cdot e^{-ds}$$

$$y(t) = \begin{cases} 0 & t < d \\ A \cdot K \cdot \left(1 - e^{-\frac{1}{\tau}(t-d)}\right) & t \geq d \end{cases}$$

$$y(d+\tau) \approx 0.63 \cdot A \cdot K, \qquad y(d+3\tau) \approx 0.95 \cdot A \cdot K, \qquad y(d+4\tau) \approx 0.98 \cdot A \cdot K$$

```
s=tf('s');
step(7/(5*s+1)*exp(-2.5*s)), grid on %A=1    K=7, tau=5
RayasHoriz(7)
TresRayasVert(5,2.5)
```




## Utilidad práctica del resultado

### 1.-Descripción "rápida" de la respuesta temporal ante escalón

Ante un incremento brusco (escalón) de entrada, la salida se incrementa $K$ veces el incremento de entrada, con transitorio no oscilatorio, alcanzando el nuevo equilibrio en "3 o 4 veces $\tau$" tiempo... Bueno, si hay retraso, tarda $d$ en empezar a notarse el cambio y, obviamente, todo se produce $d$ segundos más tarde.

A un "lego" en la materia le diremos que "*la temperatura sube 4ºC por cada 1% que abras la válvula de vapor, la temperatura tarda unos 10 minutos en estabilizarse alrededor del nuevo equilibrio*". No necesita saber Matlab, ni EDO, ni Laplace para entender eso.

### 2.-Identificación experimental "caja negra" de forma rápida

Si observo un transitorio no oscilatorio que se parece a las simulaciones de 1er orden, puedo ajustar una EDO o FdT a dicho experimento sin realmente saber la física de lo que hay dentro, simplemente mirando el valor final y el tiempo en el que alcanza el 63% de dicho valor... Se hace en "segundos", sin necesidad más que de una calculadora básica.

En la práctica, el ajuste suele ser "aproximado" en bastantes casos... si las cosas no cuadran demasiado, podemos recurrir a optimización para minimizar el error cuadrático entre simulación y experimento, pero eso ya es "harina de otro costal", y requiere Matlab, toolboxes varias... mientras que lo del 63%, con un poco de suerte, se hace "a ojo", sin teclear nada en computador directamente sobre una gráfica.

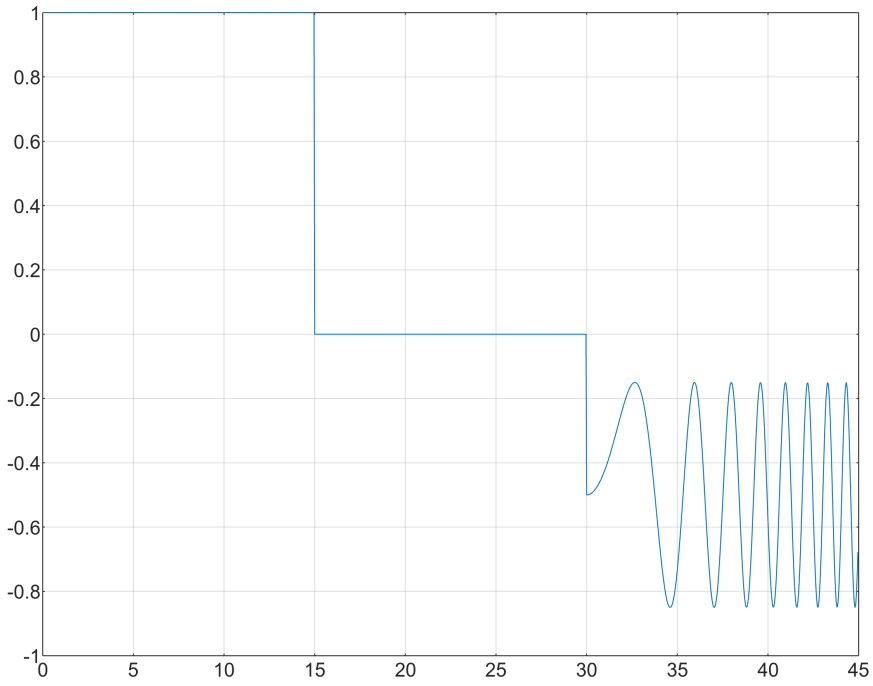
## Ejemplos identificación experimental: carga de datos de laboratorio

```
load experimentos.mat
```

**NOTA sobre escalado y variables incrementales**

Todas las entradas han sido escalón UNITARIO, y luego un escalón descendente de otra amplitud + onda tipo "chirp" para validar.

```
plot(T1,U1), grid on
```



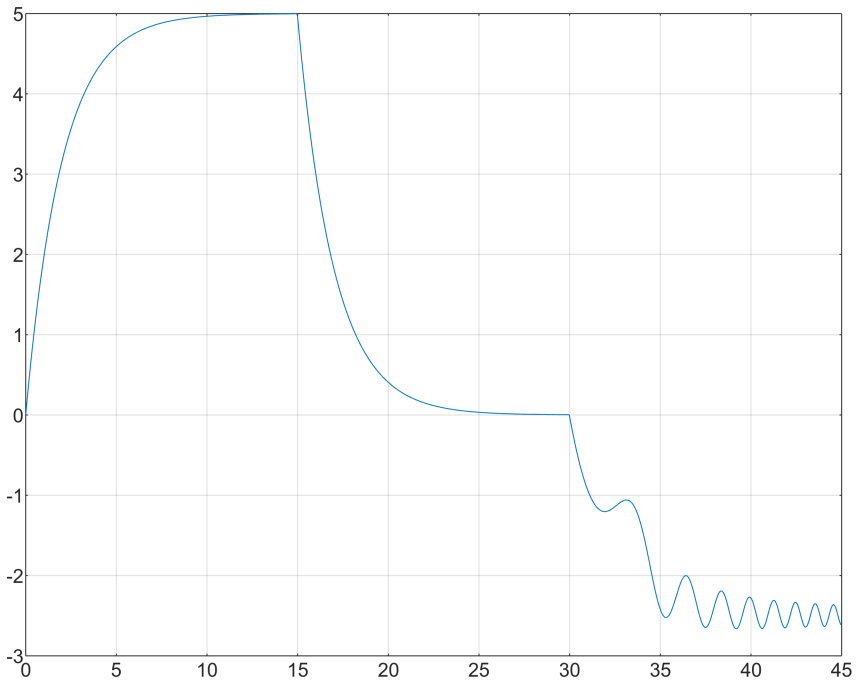

**Variables incrementales:** en un experimento real, si entrada o salida son "incremento" desde algo que no es cero, entonces hay que restar ese punto de operación para que todo empiece en cero.

**Escalado:** si el experimento tuviera un incremento de entrada real de "2 unidades", bueno, pues por linealidad, la gráfica que manejaría en los ejemplos de abajo sería la "Y_experimental dividida por 2".

Por ejemplo, si mi experimento "abriese una válvula de agua caliente del 40% al 45%" y registrase un "incremento de temperatura de salida de intercambiador de 55ºC a 70ºC", entonces yo trasladaría todo a cero (entrada de 0 a 5%, salida de 0 a 15ºC, incrementales), y escalaría posteriormente dividiendo por 5 (si el proceso es lineal, incremento de entrada de 0 a 1% incrementaría salida de 0 a 3ºC) para hacer los ajustes a continuación. La "ganancia estática" sería de "3ºC por unidad porcentual de apertura de válvula", y refinamientos sobre dinámica y transitorio ya requerirían el análisis a continuación.

## Caso 1: "clavado"

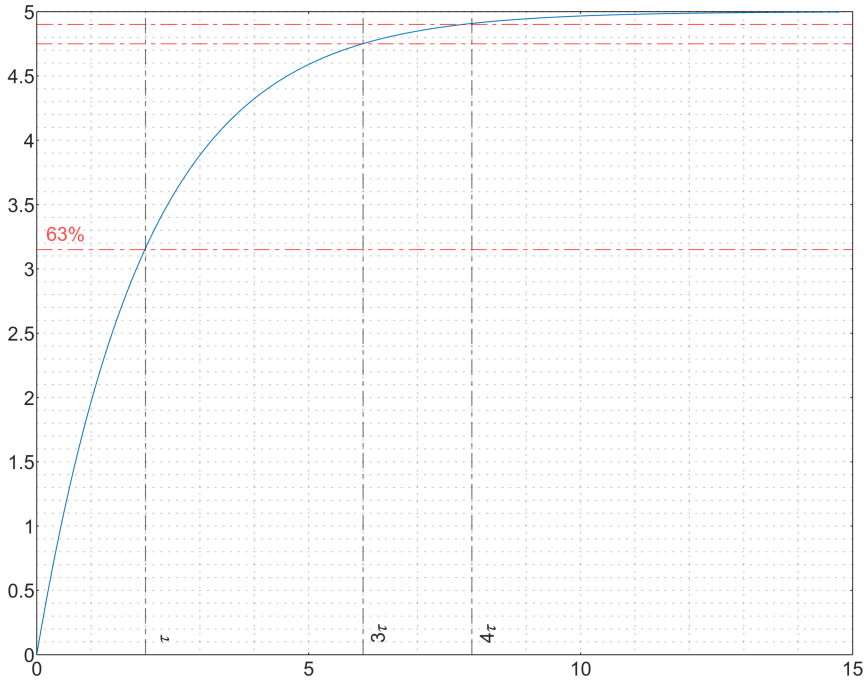
```
plot(T1,Y1), grid on
```




```
Nesc=370;
plot(T1(1:Nesc),Y1(1:Nesc)), grid minor
ValFin=5;
RayasHoriz(ValFin)
tau=2
```

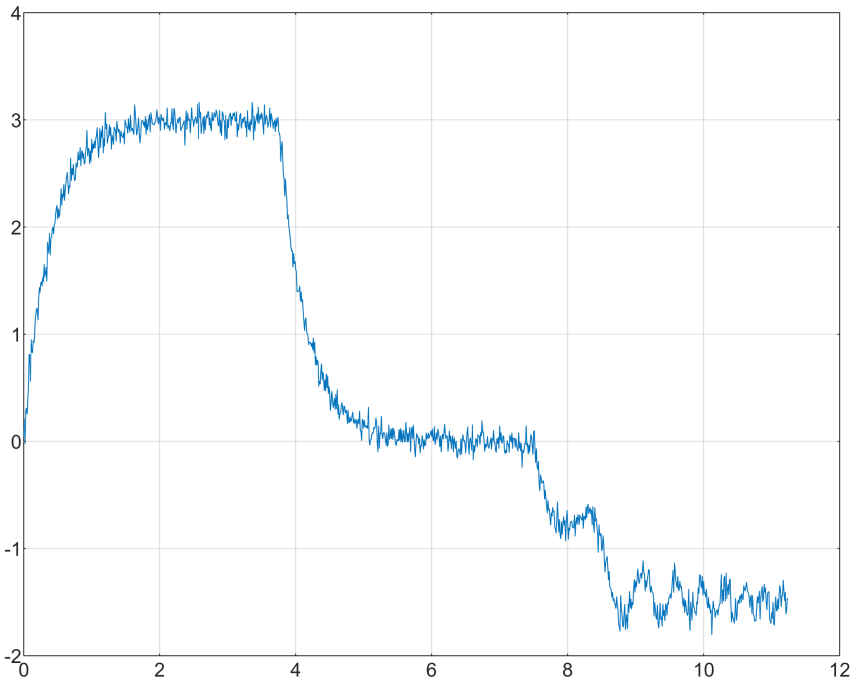
```
tau = 2
```

```
TresRayasVert(tau)
```

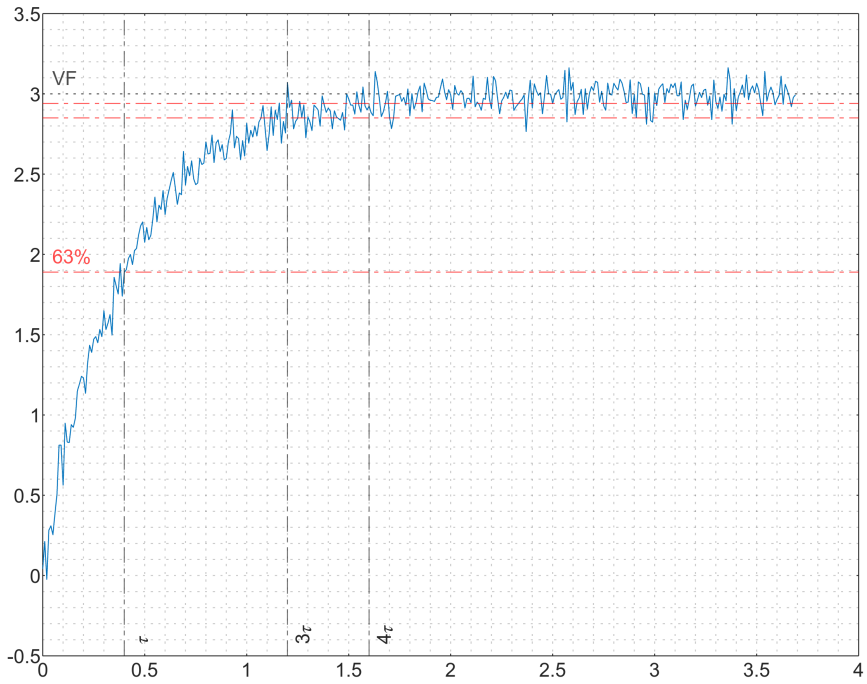


## Caso 2: clavado pero con ruido

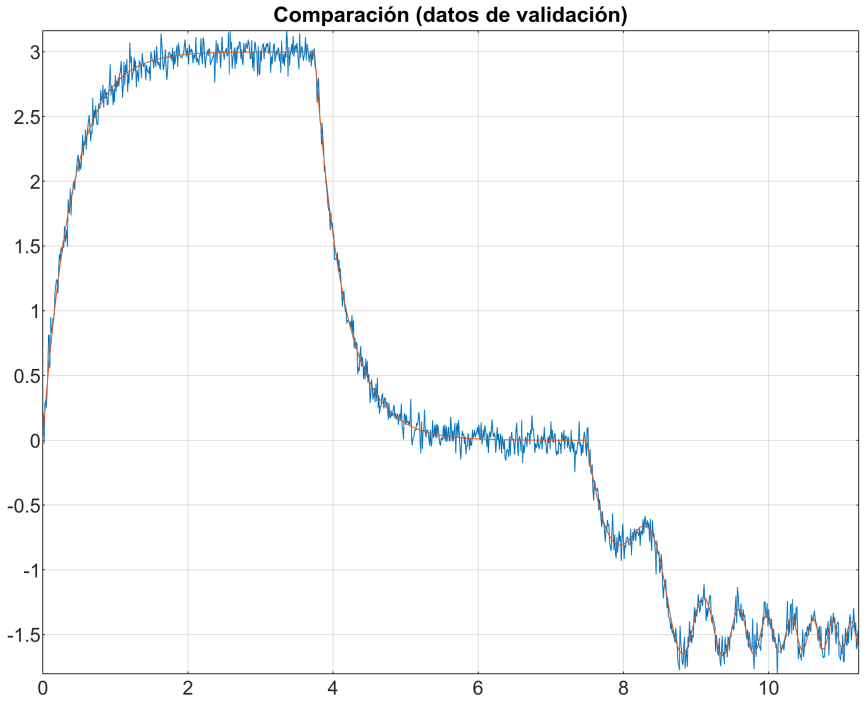
```
plot(T2,Y2), grid on
```




```
plot(T2(1:Nesc),Y2(1:Nesc)), grid minor
ValFin=3;
RayasHoriz(ValFin)
tau=0.4;
TresRayasVert(tau)
```

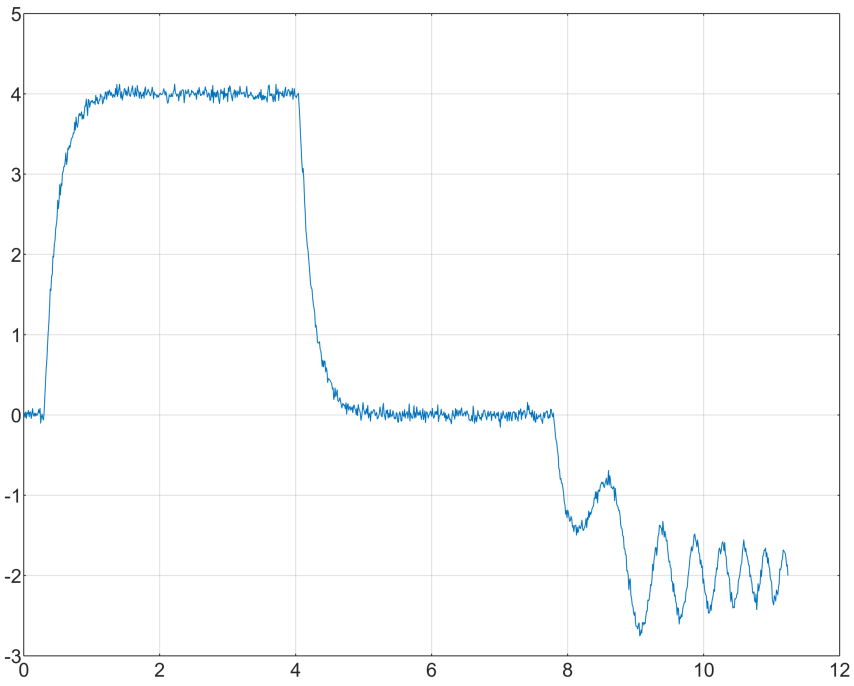



```
plot(T2,Y2), grid on
simula(tau,0,ValFin,U2,T2)
axis tight, title("Comparación (datos de validación)")
```

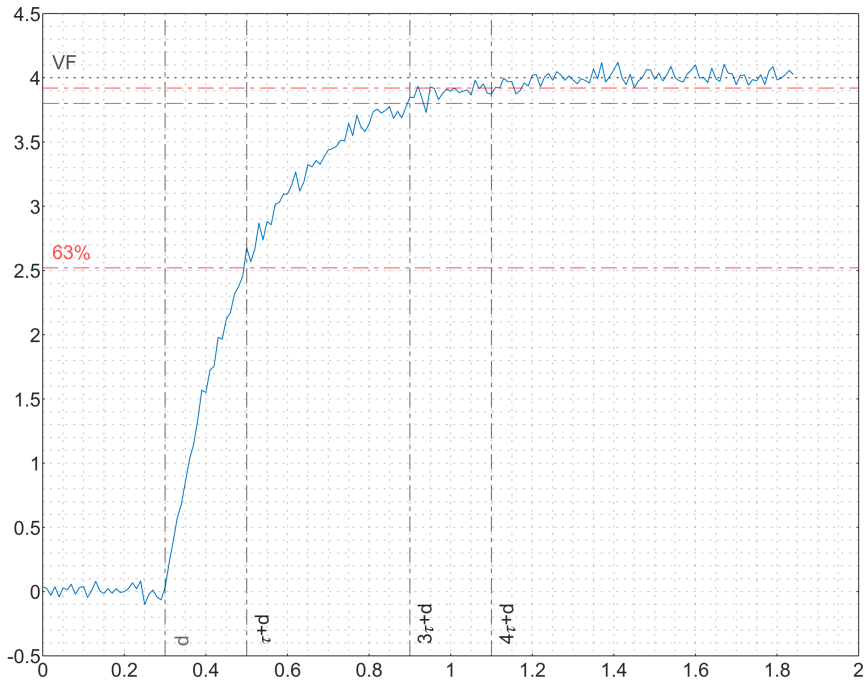



## Caso 3: "clavado" con retardo

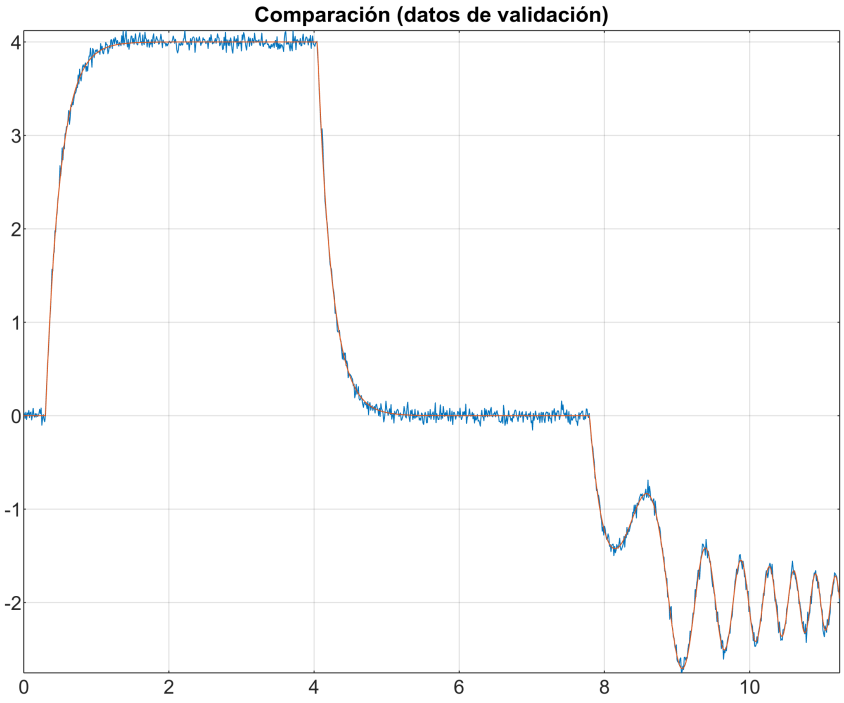
```
plot(T3,Y3), grid on
```




```
plot(T3(1:Nesc/2),Y3(1:Nesc/2)), grid minor
ValFin=4;
RayasHoriz(ValFin)
tau=0.2;retard=0.3;
TresRayasVert(tau,retard)
```

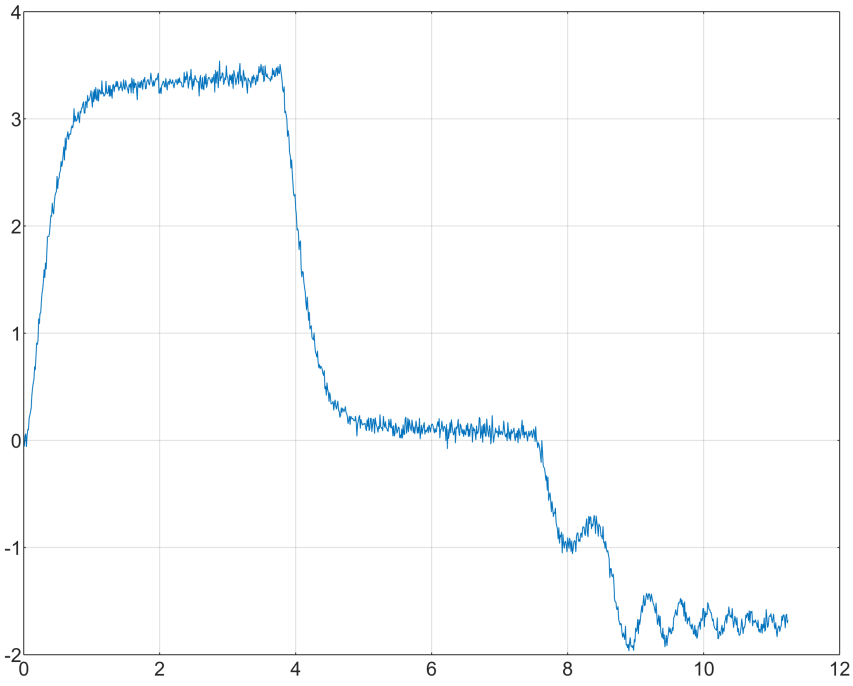



```
plot(T3,Y3), grid on
simula(tau,retard,ValFin,U3,T3)
axis tight, title("Comparación (datos de validación)")
```



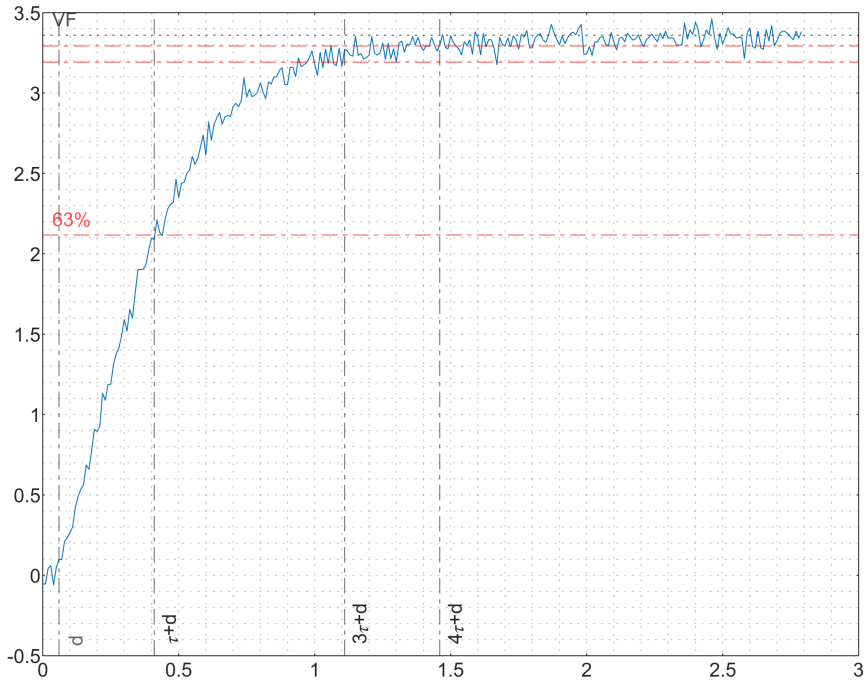

## Caso 4: esto ya va aceptable, pero no "clavado"

```
plot(T4,Y4), grid on
```

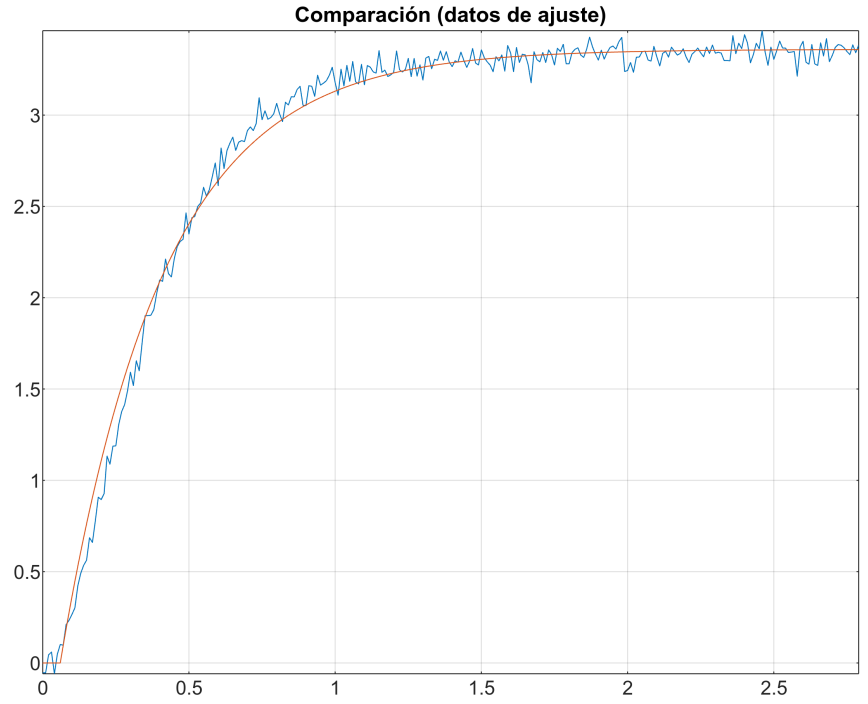



```
plot(T4(1:280),Y4(1:280)), grid minor
ValFin=3.36;
RayasHoriz(ValFin)
tau=0.35;retard=0.06;
```

```
TresRayasVert(tau,retard)
```

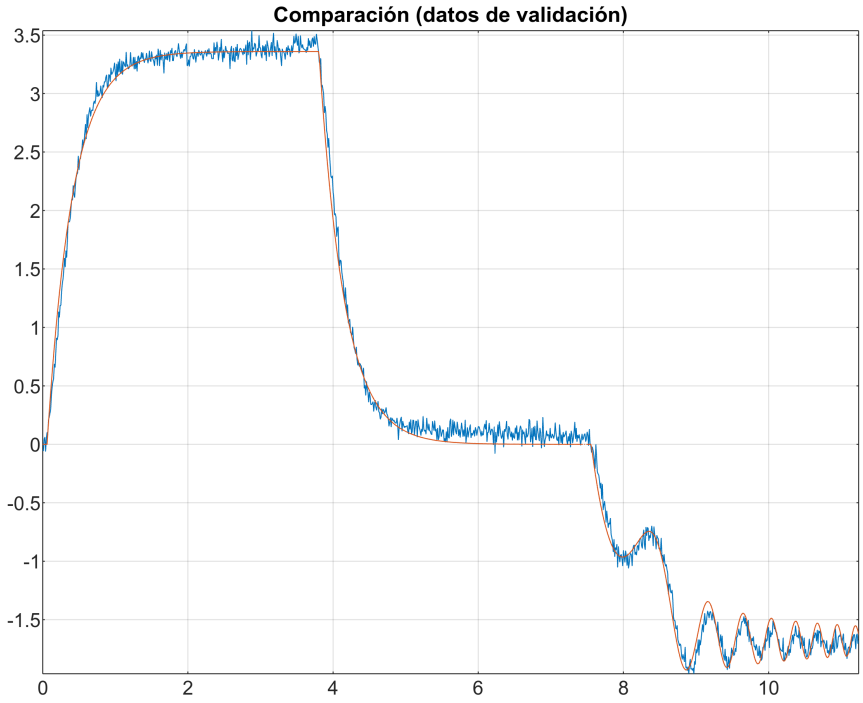



```
plot(T4(1:280),Y4(1:280)), grid on
simula(tau,retard,ValFin,U4(1:280),T4(1:280))
axis tight, title("Comparación (datos de ajuste)")
```




```
plot(T4,Y4), grid on
simula(tau,retard,ValFin,U4,T4)
```

```
axis tight, title("Comparación (datos de validación)")
```




### Ajuste con optimización (system ID toolbox, procest)

Vamos a probar por "fuerza bruta", que Matlab encuentre algo... para comparar con el "manual"

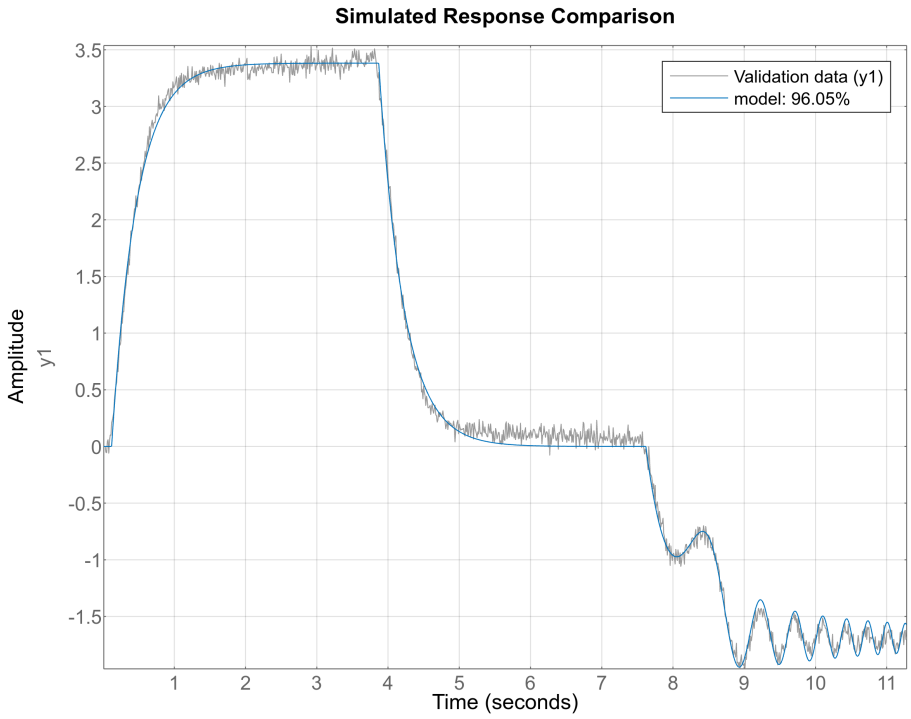
```
U=ones(size(T4));
Ts4=T4(2)-T4(1)
```

```
Ts4 = 0.0100
```

```
IDd=iddata([0;0;0;Y4], [0;0;0;U4'], Ts4);  %para que tenga claro que
empezamos en cero
model=procest(IDd,'P1D',procestOptions(InitialCondition="zero"));
tf(model)
```

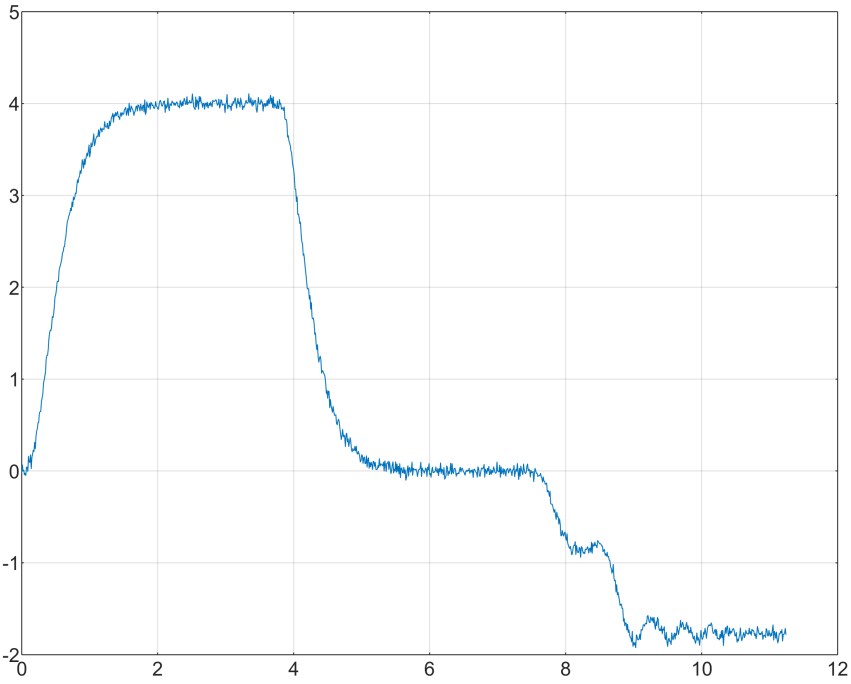
```
ans =

  From input "u1" to output "y1":
                    3.382
  exp(-0.0795*s) * -----------
                   0.348 s + 1

Continuous-time transfer function.
```

```
compare(IDd,model,compareOptions(InitialCondition="zero")), grid on
```



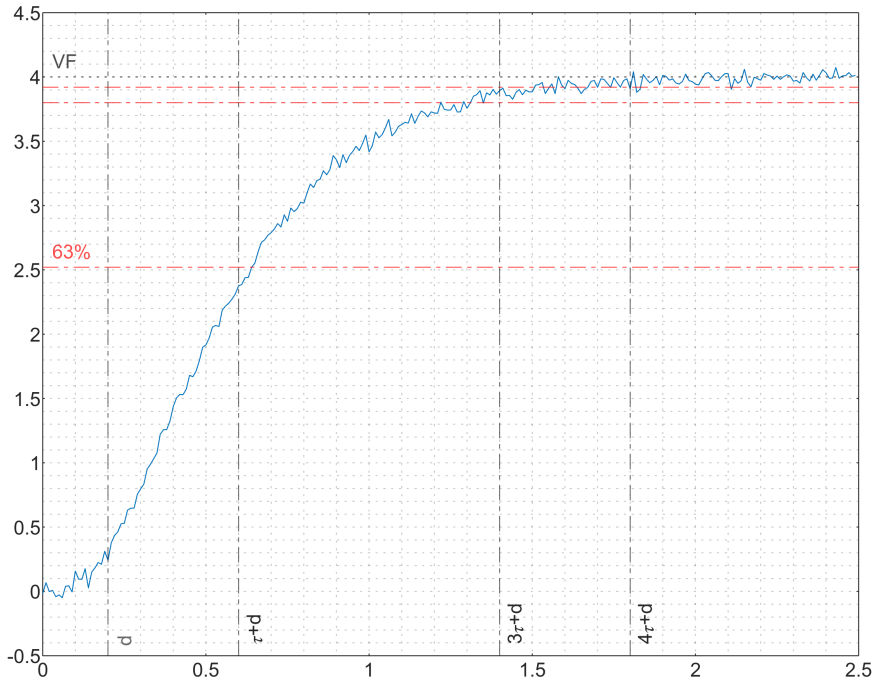

## Caso 5: esto ya no va bien, sobre todo validación

```
plot(T5,Y5), grid on
```

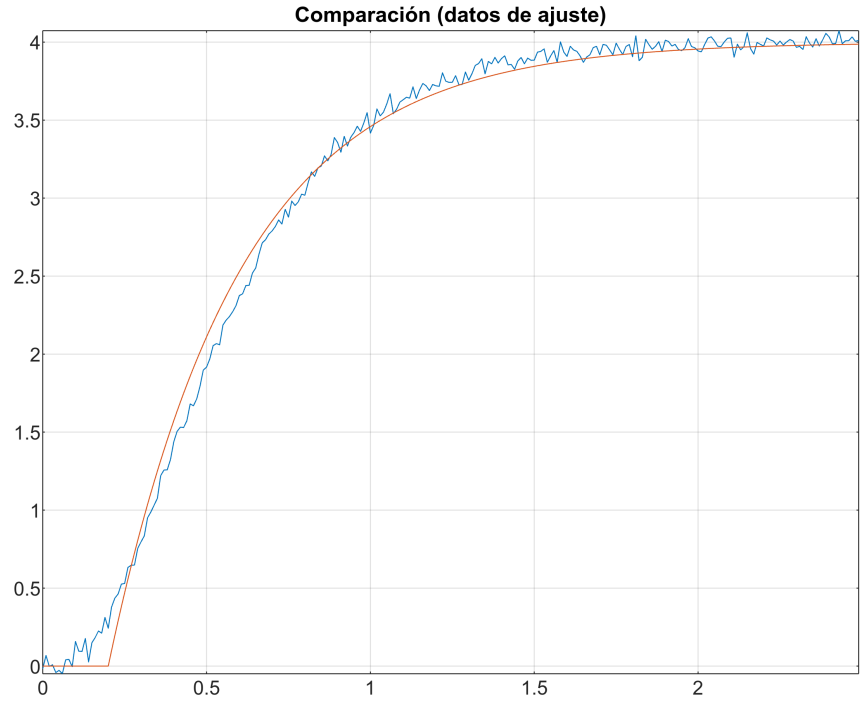


```
plot(T5(1:250),Y5(1:250)), grid minor
ValFin=4;
RayasHoriz(ValFin)
tau=0.4;retard=0.2;
```

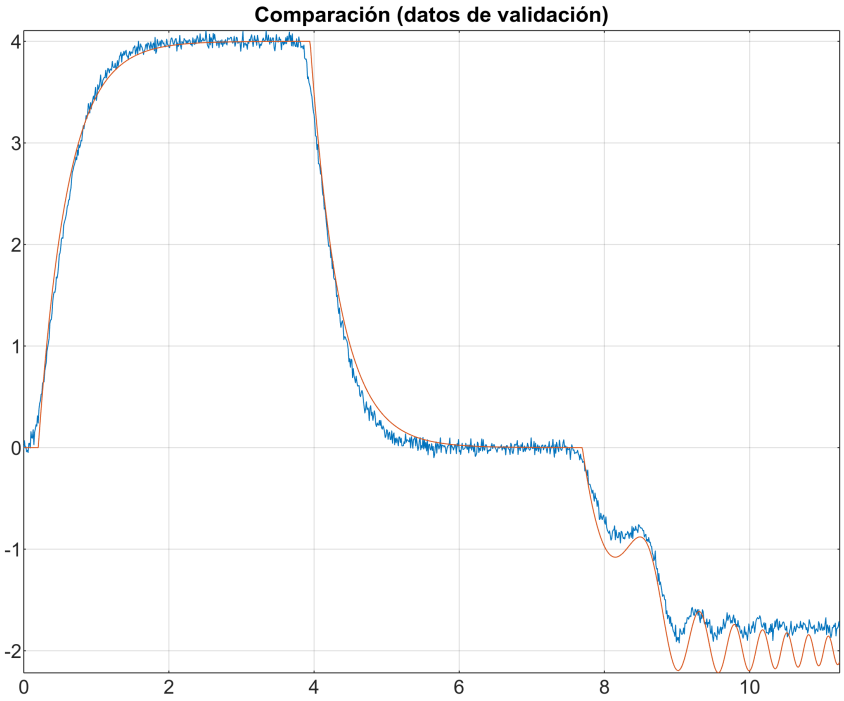
```
TresRayasVert(tau,retard)
```




```
plot(T5(1:250),Y5(1:250)), grid on
simula(tau,retard,ValFin,U5(1:250),T5(1:250))
axis tight, title("Comparación (datos de ajuste)")
```




```
plot(T5,Y5), grid on
simula(tau,retard,ValFin,U5,T5)
axis tight, title("Comparación (datos de validación)")
```




### Ajuste con optimización (system ID toolbox, procest)

Vamos a probar por "fuerza bruta", que Matlab encuentre algo...

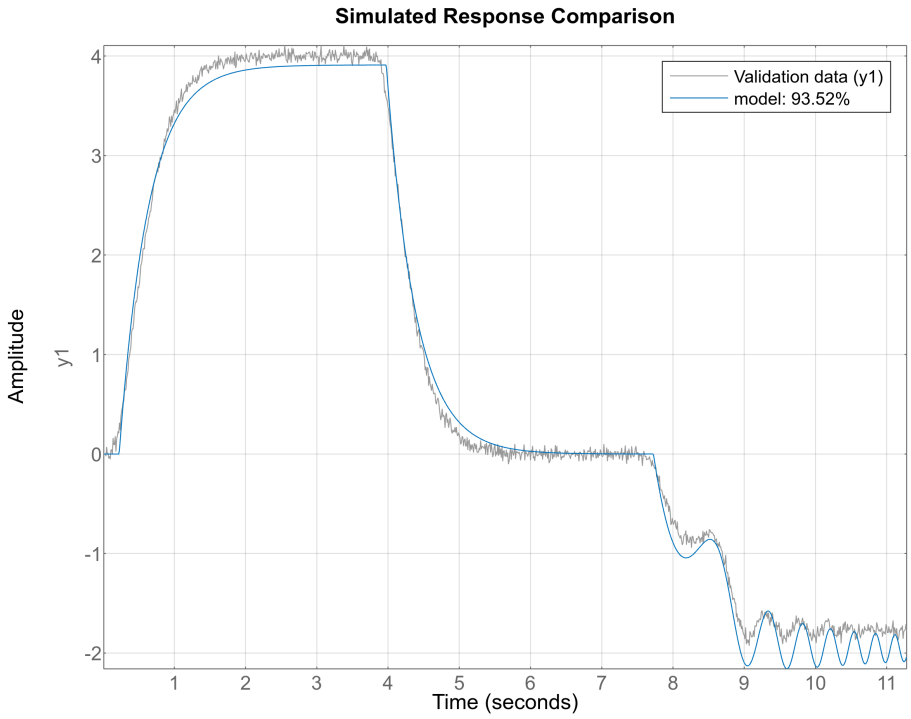
```
U=ones(size(T5));
Ts5=T5(2)-T5(1)
```

```
Ts5 = 0.0100
```

```
IDd=iddata([0;0;0;Y5],[0;0;0;U5'],Ts5); %para que tenga claro que
empezamos en cero
model=procest(IDd,'P1D',procestOptions(InitialCondition="zero"));
tf(model)
```

```
ans =

  From input "u1" to output "y1":
                    3.91
  exp(-0.186*s) * ------------
                  0.4087 s + 1

Continuous-time transfer function.
```

```
compare(IDd,model,compareOptions(InitialCondition="zero")), grid on
```




Podríamos proponer ajustar modelos más complicados incluso no lineales o, bueno, depende de la precisión de la aplicación, conformarnos...

## Conclusiones

El ajuste "a mano" con lo del 63%, 95%, 98% sirve para hacerse una idea rápida sin "system id toolbox" ni "optimización con, por ejemplo, algoritmos genéticos", etc. con una "gráfica" de un experimento para ajustar una dinámica de 1er orden... En experimentos reales, habrá veces donde entrenamiento y validación ajusten bien, otras veces no tanto.

## Apéndice (funciones auxiliares)

```
function RayasHoriz(VF) %argumento de entrada es el valor final VF
%hace líneas horiz. en 63%, 95% y 98% del valor final para comprobar las
fórmulas de teoría.
 yline(VF,':k',Label="VF",LabelHorizontalAlignment="left") %Valor final
 yline(0.63*VF,'-.r',Label="63%",LabelHorizontalAlignment="left") % 63%
 yline(0.95*VF,'-.r') % 95%
 yline(0.98*VF,'-.r') % 98%
end
```

```
function TresRayasVert(tau,retard)
%hace 3 líneas en tau, 3tau y 4tau para comprobar las fórmulas de teoría.
arguments
    tau
```

```
    retard=0 %si no pongo retardo, lo hago cero
end
 textod="";
 if(retard>0)
    xline(retard,'-.',Label="d",LabelVerticalAlignment="bottom")
%marcamos cuándo se supone que un 1er orden empezará a subir
    textod="+d";
 end
 d=retard;
 %marcamos cuándo sería el 63%, 95% y 98% de VF.
 xline(tau+d,'-.',label="\tau"+textod,LabelVerticalAlignment="bottom"),
 xline(3*tau+d,'-.',label="3\tau"+textod,LabelVerticalAlignment="bottom"),
 xline(4*tau+d,'-.',label="4\tau"+textod,LabelVerticalAlignment="bottom")
end
```

```
function simula(tau,retard,ValFin,U,T)
%superpone la simulación de un modelo de 1er orden a la gráfica actual,
para
%comparar
 s=tf('s');
 G=ValFin/(tau*s+1)*exp(-retard*s); %1er orden con retardo
 [Y,T]=lsim(G,U,T);
 hold on
 plot(T,Y)
 hold off
end
```