Respuesta temporal sistema 2o orden (oscilatorio) ante rampa truncada

© 2022, Antonio Sala Piqueras, Universitat Politècnica de València. Todos los derechos reservados.

Presentaciones en vídeo: http://personales.upv.es/asala/YT/V/dlyramp.html

Este código ejecutó sin errores en Matlab **R2022b** (Linux)

Objetivo: comprender el cálculo de respuesta temporal por transformada de Laplace inversa usando la descomposición en fracciones simples (partial fraction expansion) de la Symbolic Math Toolbox de Matlab, en un caso muy común en la práctica de "*transición en rampa desde un punto de operación a otro*".

Tabla de Contenidos

Modelo del sistema

Modelo (repr. interna)

Consideremos un sistema en representación variable de estado normalizada

$$\frac{dv}{dt} = -q/2 - v/2 + u,$$

$$\frac{dq}{dt} = v - u$$

siendo $y(t) = q(t)$ la salida de interés, $u(t)$ la variable de entrada, y siendo $v(t)$, $q(t)$ las variables de estado.

Obviamente, en forma matricial normalizada, el modelo se representaría con

$$\dot{x} = Ax + Bu, \qquad y = Cx + Du,$$

siendo x=[v;q], y las matrices:

```
A=[-.5 -.5;1 0]; B=[1;-1]; C=[0 1]; D=0;
```

Modelo como EDO de orden 2

Derivando la segunda ecuación tendremos:

$$\frac{d^2q}{dt^2} = \frac{dv}{dt} - \frac{du}{dt} = -q/2 - v/2 + u - \frac{du}{dt} = -q/2 - \frac{1}{2}(\frac{dq}{dt} + u) + u - \frac{du}{dt}$$

Por tanto, como $y = q$, operando podemos escribir:

$$\frac{d^2y}{dt} + \frac{1}{2}\frac{dy}{dt} + \frac{1}{2}y = \frac{1}{2}u - \frac{du}{dt}$$

que es una EDO de 2º orden, que podríamos considerar "equivalente" a las dos EDO de 1er orden de arriba... pero las condiciones iniciales y(0), y'(0) junto con u(0) necesarias para Laplace tienen un sentido "físico" diferente.

Nótese que hemos tenido que "calcular derivadas" de ecuaciones; eso no ocurre trabajando en Laplace, donde todas las operaciones son "algebraicas" (sumas, restas, multiplicaciones y divisiones).

Obtención de la función de transferencia del sistema

*Sólo necesitamos FdT porque se plantea con condiciones iniciales nulas.

FdT con Control Systems Toolbox

En "control toolbox" obtendríamos la FdT con el comando "`tf`" (transfer function):

```
sys=ss(A,B,C,D);
tf(sys)
```

```
ans =

      -s + 0.5
  -----------------
  s^2 + 0.5 s + 0.5

Continuous-time transfer function.
```

Pero el control toolbox no puede obtener la "fórmula" de la respuesta ante las entradas, y sólo tiene de forma "fácil" la respuesta ante entrada escalón (`step`) o senoidal (`bode`). Podríamos preparar con

`lsim` la simulación ante la rampa de entrada pero no lo haremos por brevedad, dado que en este material buscamos la "fórmula" de la salida mediante Laplace. Por ello, no vamos a usar las rutinas del Control toolbox, centrándonos en las manipulaciones con la Symbolic Toolbox, más cercanas a las necesarias en un examen de "papel y bolígrafo".

FdT con Symbolic Toolbox a partir de repr. interna normalizada

En symbolic toolbox (objeto de este vídeo), haremos transformada de Laplace de las EDO del modelo y despejaremos; con cond. iniciales nulas se reduce a:

```
syms s v q y u
v0=0; q0=0;
ModeloLapl=[ s*v - v0 == -q/2 - v/2 + u;
             s*q - q0 == v - u;
                    y == q ]
```

```
ModeloLapl =
```

$$\begin{pmatrix} s\,v = u - \frac{q}{2} - \frac{v}{2} \\ q\,s = v - u \\ y = q \end{pmatrix}$$

```
sol=solve(ModeloLapl,{v,q,y},ReturnConditions=true)
```

```
sol = struct with fields:
              v: (u + 2*s*u)/(2*s^2 + s + 1)
              q: (u - 2*s*u)/(2*s^2 + s + 1)
              y: (u - 2*s*u)/(2*s^2 + s + 1)
    parameters: [1×0 sym]
    conditions: 2*s^2 + s ~= -1
```

```
FdT=simplify(sol.y/u) %coincide con Control Toolbox
```

```
FdT =
```

$$-\frac{2\,s-1}{2\,s^2+s+1}$$

Nota: en problemas con varias entradas lo de "dividir por la entrada" estaría mal: "lo que multiplica a una entrada específica" sería la "derivada parcial" respecto a ella:

```
diff(sol.y,u)
```

```
ans =
```

$$-\frac{2\,s-1}{2\,s^2+s+1}$$

Por supuesto, la fórmula teórica obtenida con la resolución de la ecuación de estado matricial normalizada en Laplace:

$$s \cdot X(s) - x(0) = AX(s) + Bu(s),\ (sI - A) \cdot X(s) = Bu(s) + x(0),\ X(s) = (sI - A)^{-1}(B \cdot u(s) + x(0))$$

y luego la ecuación de salida:

$$y(s) = CX(s) + Du(s) = (\,C(sI - A)^{-1}B + D\,) \cdot u(s) + C(sI - A)^{-1} \cdot x(0)$$

implica que la función de transferencia es:

$$G(s) = C(sI - A)^{-1}B + D$$

Obviamente, Matlab simbólico dará el mismo resultado:

```
simplify(C*inv(s*eye(2)-A)*B+D)
```

```
ans =
```

$$-\frac{2\,s-1}{2\,s^2+s+1}$$

FdT a partir de la EDO de 2º orden

A partir de aquí:

$$\frac{d^2y}{dt} + \frac{1}{2}\frac{dy}{dt} + \frac{1}{2}y = \frac{1}{2}u - \frac{du}{dt}$$

la FdT se reduce casi a "copiar coeficientes", porque su transformada de Laplace con cond. iniciales nulas es:

$$s^2 \cdot Y + 0.5s \cdot Y + 0.5 \cdot Y = 0.5 \cdot U - s \cdot U$$

o sea

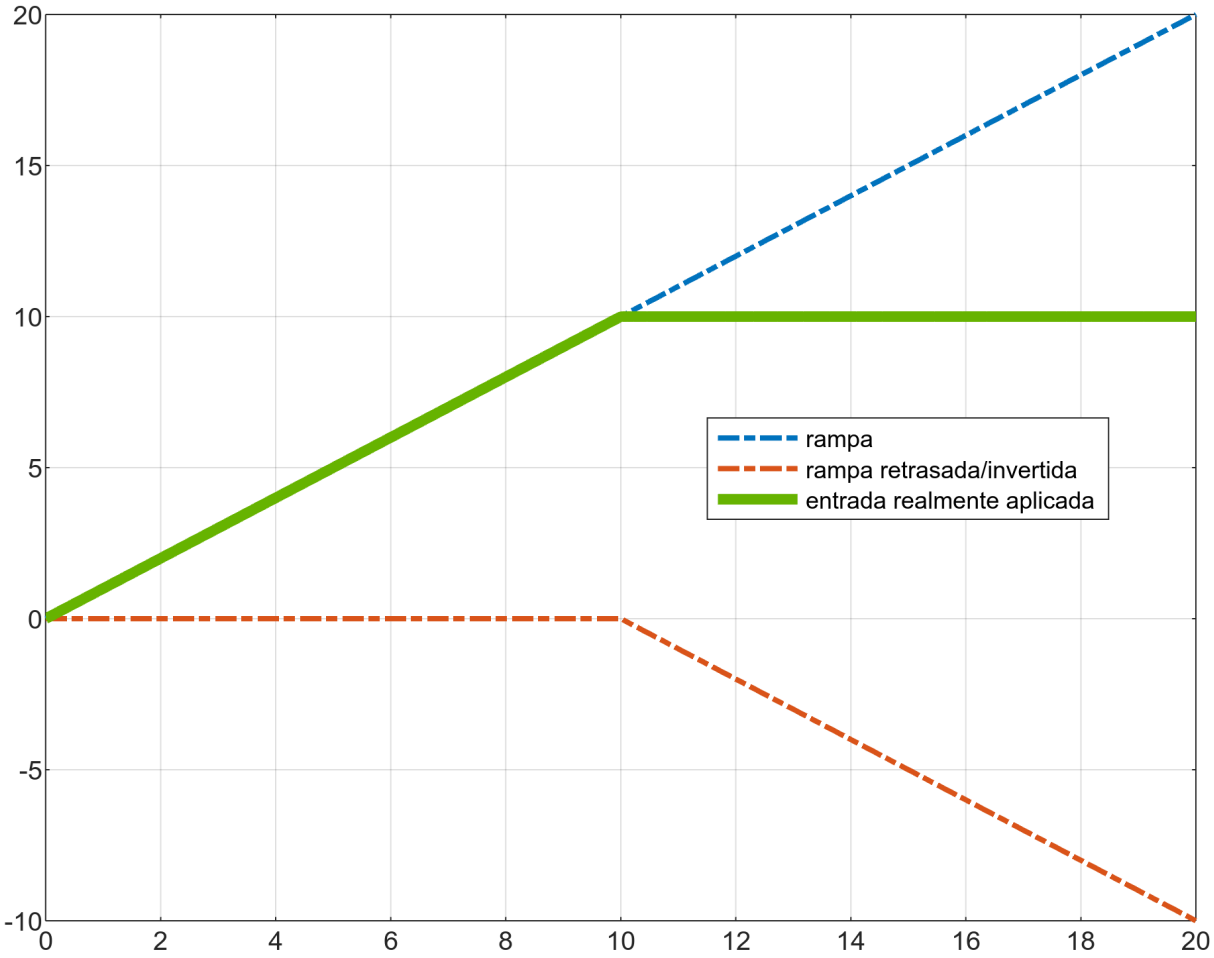
$$(s^2 + 0.5s + 0.5) \cdot Y = (0.5 - s) \cdot U$$

resultando inmediatamente la misma FdT calculada antes por otros métodos.

Entrada para calcular respuesta temporal

El objetivo será calcular la salida ante condiciones iniciales **nulas** y la entrada $u(t)$ en forma rampa truncada:

```
syms t real
u1=heaviside(t)*t;
u2=-subs(heaviside(t)*t, t, t-10);
u_time=u1+u2;
fplot([u1;u2],[0 20],LineWidth=2,LineStyle='-.'), grid on, hold on
fplot(u_time,[0 20],LineWidth=4,Color=[.4 .7 0]), grid on, hold off
legend("rampa","rampa retrasada/invertida","entrada realmente aplicada", Location='best
```



Transformada de Laplace de la entrada

```
U=laplace(u_time)
```

```
U =
```

$$\frac{1}{s^2} - \frac{e^{-10\,s}}{s^2}$$

Resolución de la respuesta temporal

Método 1 (recomendado): solucionar "fragmentos" sin retardo y luego retrasar y superponer

La solución ante rampa $u_1(t) = t,\ U_1(s) = 1/s^2$ será:

```
U1=laplace(u1)
```

```
U1 =
```

$$\frac{1}{s^2}$$

```
Y1_s=FdT*U1
```

```
Y1_s =
```

$$-\frac{2\,s-1}{s^2\,(2\,s^2+s+1)}$$

Como el sistema tiene polos complejos:

```
[N,D]=numden(Y1_s)
```

```
N = 1 − 2 s
D = s^2 (2 s^2 + s + 1)
```

```
solve(D==0)
```

```
ans =
```

$$\begin{pmatrix} 0 \\ 0 \\ -\frac{1}{4}-\frac{\sqrt{7}\,\mathrm{i}}{4} \\ -\frac{1}{4}+\frac{\sqrt{7}\,\mathrm{i}}{4} \end{pmatrix}$$

Por tanto, deberíamos hacer:

$$Y_1(s) = \frac{0.5 - s}{s^2(s^2 + 0.5s + 0.5)} = \frac{A}{s} + \frac{B}{s^2} + \frac{Cs + D}{s^2 + 0.5s + 0.5}$$

```
partfrac(Y1_s)
```

```
ans =
```

$$\frac{1}{s^2}-\frac{3}{s}+\frac{6\,s+1}{2\,s^2+s+1}$$

o mejor, para obtener directamente en tablas cada fracción simple, expresando el factor de 2º orden como: $s^2 + 0.5s + 0.5 = (s + \frac{1}{4})^2 + \frac{7}{16}$

deberíamos calcular la descomposición:

$$Y_1(s) = \frac{0.5 - s}{s^2(s^2 + 0.5s + 0.5)} = \frac{A}{s} + \frac{B}{s^2} + \frac{\bar{C} \cdot (s + \frac{1}{4})}{(s + \frac{1}{4})^2 + \frac{7}{16}} + \frac{\bar{D} \cdot \frac{\sqrt{7}}{4}}{(s + \frac{1}{4})^2 + \frac{7}{16}}$$

donde obviamente $\bar{C} = C$ y $\bar{D}$ se obtendría fácilmente de C y D.

Entonces, la transformada inversa sería:

$$Y_1(t) = A + B \cdot t + \bar{C} \cdot e^{-\frac{1}{4}t} \cos(\frac{\sqrt{7}}{4}t) + \bar{D} \cdot e^{-\frac{1}{4}t} \sin(\frac{\sqrt{7}}{4}t)$$

Con las adecuadas operaciones más o menos tediosas para sacar los 4 coeficientes de las fracciones simples y tablas, resultará en:

```
Y1_t=ilaplace(Y1_s)
```

```
Y1_t =
```

$$t + 3\,\mathrm{e}^{-\frac{t}{4}}\left(\cos\left(\frac{\sqrt{7}\,t}{4}\right) - \frac{\sqrt{7}\,\sin\left(\frac{\sqrt{7}\,t}{4}\right)}{21}\right) - 3$$

La salida ante $u_2(t) = \begin{cases} 0 & t < 10 \\ -(t-10) & t \geq 10 \end{cases}$, $U_2(s) = \frac{1}{s^2} \cdot e^{-10s}$ será la misma que ante u_1 pero retrasada y cambiada de signo.

En definitiva, la salida pedida será $y(t) = y_1(t) + y_2(t)$, esto es:

$$y(t) = y_1(t) + \begin{cases} 0 & t < 10 \\ -y_1(t-10) & t \geq 10 \end{cases}$$

con lo que habríamos completado la solución lápiz y papel.

Con Matlab, para graficar cosas:

```
Y2_t=-simplify(subs(heaviside(t)*Y1_t, t, t-10));
Y_metodo1=simplify(Y1_t+Y2_t)
```

```
Y_metodo1 =
```

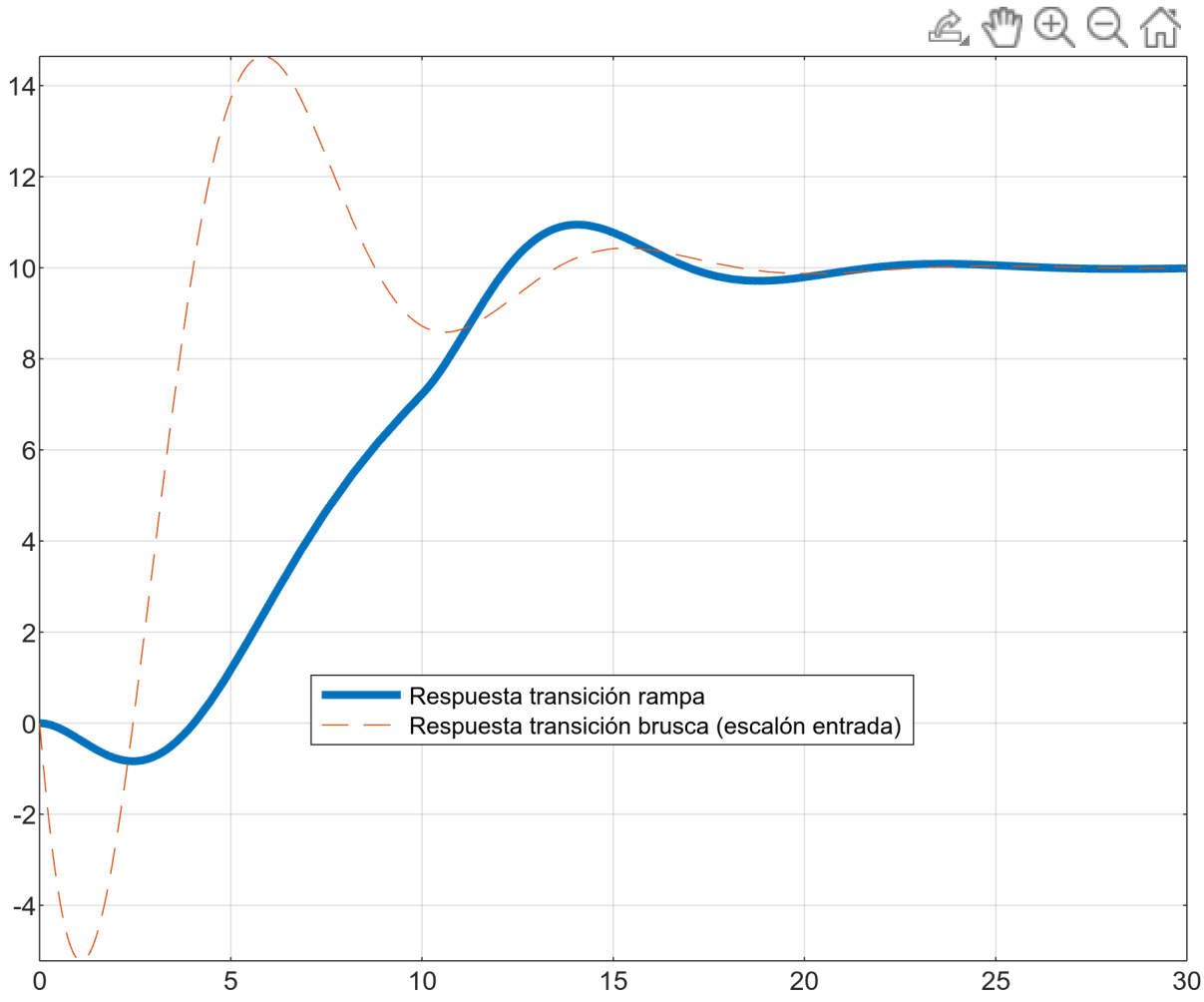
$$t - \text{heaviside}(t-10)\left(t + 3\,e^{\frac{5}{2}-\frac{t}{4}}\left(\cos(\sigma_1) - \frac{\sqrt{7}\,\sin(\sigma_1)}{21}\right) - 13\right) + 3\,e^{-\frac{t}{4}}\left(\cos\left(\frac{\sqrt{7}\,t}{4}\right) - \frac{\sqrt{7}\,\sin\left(\frac{\sqrt{7}\,t}{4}\right)}{21}\right)$$

where

$$\sigma_1 = \frac{\sqrt{7}\,(t-10)}{4}$$

Vamos a representarlo junto a la respuesta ante escalón "brusco", para entender físicamente el significado de la "subida gradual en rampa":

```
fplot(Y_metodo1,[0 30],LineWidth=3), grid on
hold on
fplot(ilaplace(FdT*10/s),[0 30],LineStyle="--"), hold off %step response
legend("Respuesta transición rampa","Respuesta transición brusca (escalón entrada)",Loc
```



Método 2: Laplace con operador retardo (exp(-ds))

Ahora por Laplace, poco a poco a mano:

```
Y=simplify(FdT*U,70)
```

```
Y =
```

$$\frac{(2\,s-1)\,(\mathrm{e}^{-10\,s}-1)}{s^2\,(2\,s^2+s+1)}$$

Por tanto, deberíamos descomponer:

$$Y_1(s)=\frac{(1-e^{-10s})s+0.5(1-e^{-10s})}{s^2(s^2+0.5s+0.5)}=\frac{A}{s}+\frac{B}{s^2}+\frac{Cs+D}{s^2+0.5s+0.5}=\frac{A}{s}+\frac{B}{s^2}+\frac{\bar{C}\cdot(s+\frac{1}{4})}{(s+\frac{1}{4})^2+\frac{7}{16}}+\frac{\bar{D}\cdot\frac{\sqrt{7}}{4}}{(s+\frac{1}{4})^2+\frac{7}{16}}$$

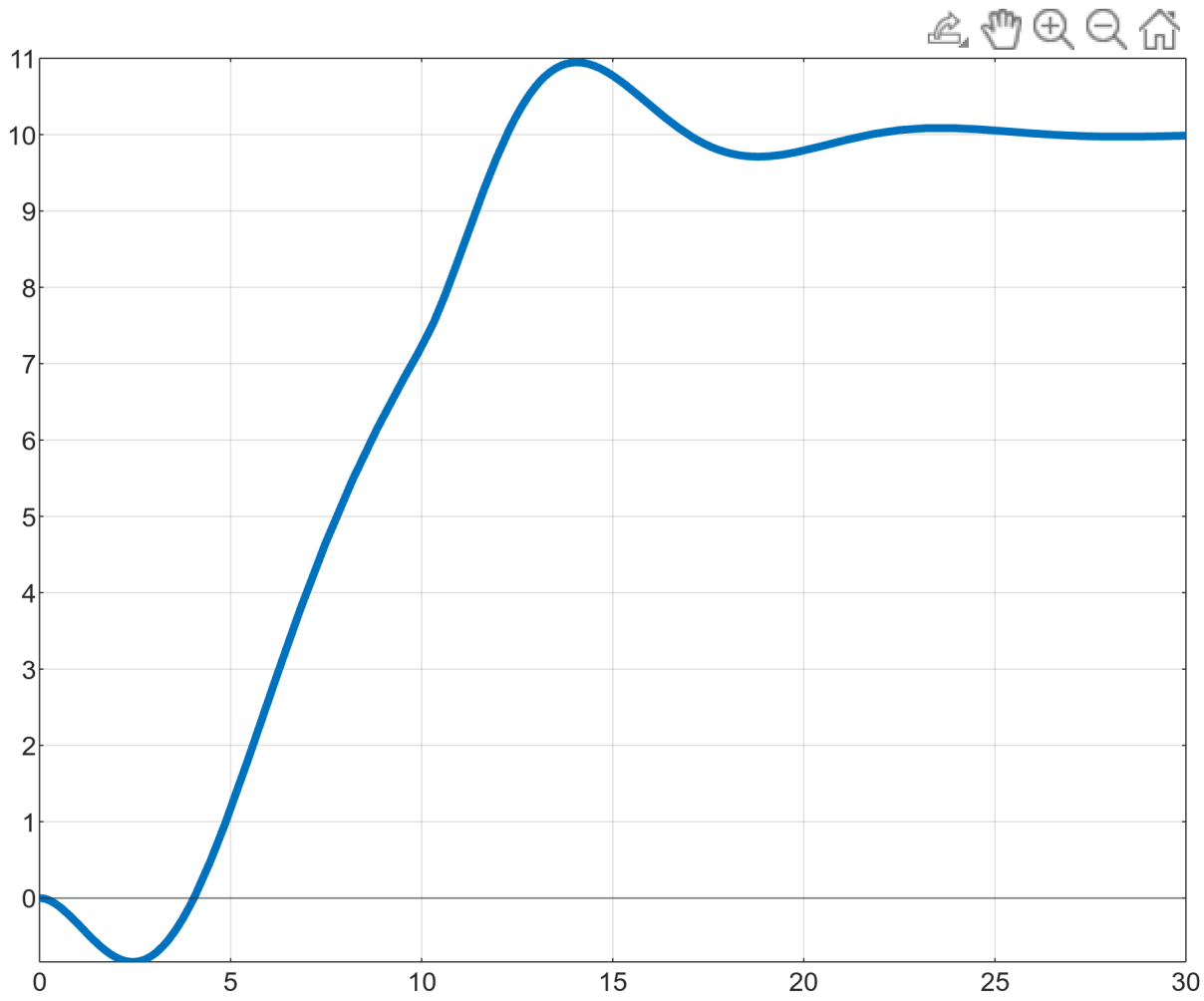
donde ahora, claro, los elementos A, B, C, D serán de tipo $A_1(1-e^{-10s})$... bueno, saldrán los mismos números que arriba, claro, multiplicados por $(1-e^{-10s})$... por eso recomiendo el primer método, para no asustarse con exponenciales en "s" en las fracciones simples cuando realmente estamos haciendo lo mismo de forma diferente.

```
partfrac(Y)
```

```
ans =
```

$$\frac{3\,\mathrm{e}^{-10\,s}-3}{s}-\frac{\mathrm{e}^{-10\,s}+s\,(6\,\mathrm{e}^{-10\,s}-6)-1}{2\,s^2+s+1}-\frac{\mathrm{e}^{-10\,s}-1}{s^2}$$

```
Y_t=ilaplace(Y);
fplot(Y_t,[0 30],LineWidth=3), grid on, yline(0) %sale lo mismo que antes
```


11
10
9
8
7
6
5
4
3
2
1
0
0
5
10
15
20
25
30