Fondamenti di Automatica

Trasformate di Laplace in Matlab
(con Symbolic Math e Control Systems toolbox)

Prof. Marcello Bonfè

Dipartimento di Ingegneria - Università di Ferrara

Tel. +39 0532 974839

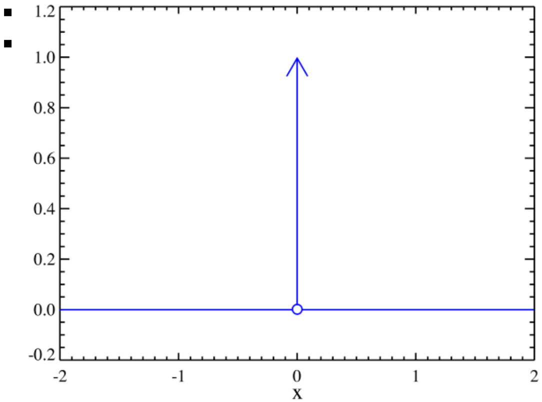
E-mail: marcello.bonfe@unife.it

Matlab: trasformate e antitrasformate di Laplace

- La trasformata e l'antitrasformata di Laplace sono strumenti matematici importanti per l'Ingegneria, soprattutto per l'Automatica
- In ambiente Matlab, le operazioni di Laplace sono supportate da due toolbox alternativi (e, aimè, ad oggi non intercambiabili):
  - **Symbolic Toolbox**: `laplace() / ilaplace()`
  - **Control Systems Toolbox**: sistemi modellati con `tf()` (Transfer Function)

Matlab: trasformate e antitrasformate di Laplace

- Operazioni di base con **Symbolic Toolbox**:
  - Trasformata dell'impulso di **Dirac**:



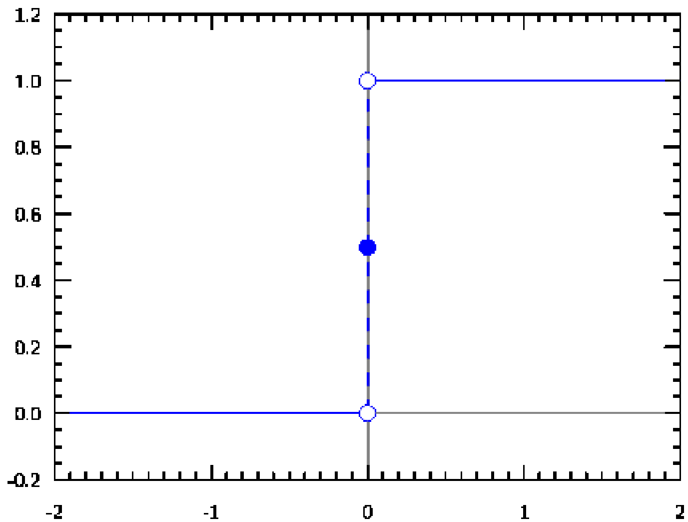

```
>> syms t s
>> D(s) = laplace(dirac(t),t,s)
 D(s) =
 1
```

NOTA: `t` = argomento della funzione da trasformare
`s` = argomento della funzione trasformata

Si possono omettere in quanto di default `laplace()` considera i simboli `t` ed `s`, purchè già definiti nel workspace

Matlab: trasformate e antitrasformate di Laplace

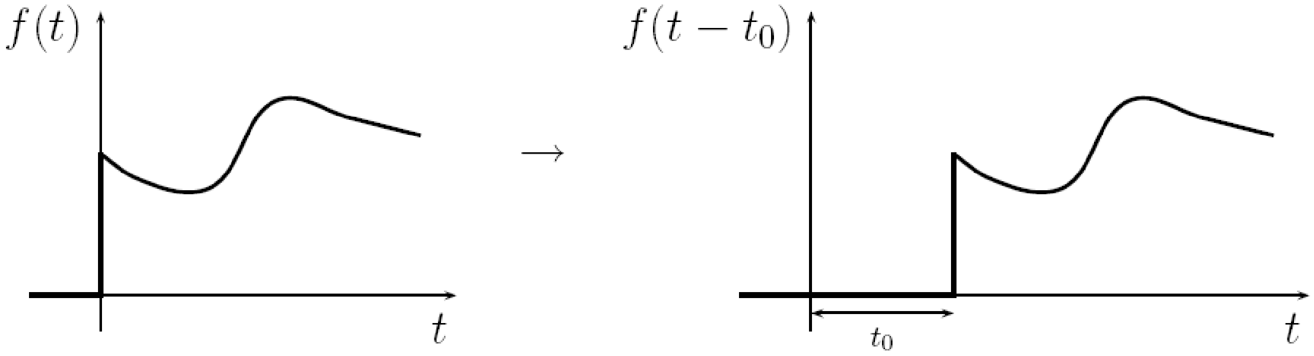
- Operazioni di base con **Symbolic Toolbox**:
  – Trasformata del gradino (i.e. funzione di **Heaviside**):




```
>> H(s) = laplace(heaviside(t))
 H(s) =
 1/s
```

Matlab: trasformate e antitrasformate di Laplace

- Operazioni di base con **Symbolic Toolbox**:
  - Traslazione nel tempo




```
>> H(s) = laplace(heaviside(t-3))
 H(s) =
 exp(-3*s)/s
```

Matlab: trasformate e antitrasformate di Laplace

- Operazioni di base con **Symbolic Toolbox**:
  - Trasformata della derivata di una funzione:

```
>> syms f(t)
>> Df=diff(f,t)
 Df(t) =
 diff(f(t), t)
>> Ds=laplace(Df)
 Ds =
 s*laplace(f(t), t, s) - f(0)
```

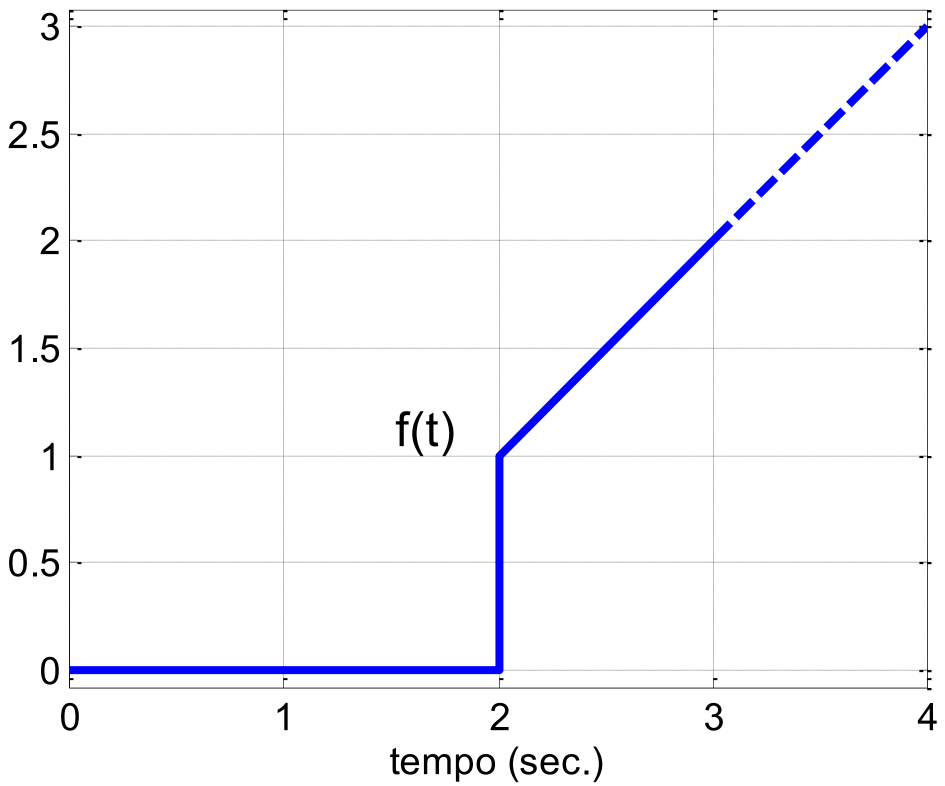
Matlab: trasformate e antitrasformate di Laplace

- Operazioni di base con **Symbolic Toolbox**:
  - Trasformate di altre funzioni notevoli:

```
>> syms a t s w
>> E(s)=laplace(exp(a*t))
 E(s) =
 -1/(a - s)
>> C(s)=laplace(cos(w*t))
 C(s) =
 s/(s^2 + w^2)
>> C(s)=laplace(sin(w*t))
 C(s) =
 w/(s^2 + w^2)
```

Matlab: trasformate e antitrasformate di Laplace

- Operazioni di base con **Symbolic Toolbox**:
  - Trasformata di segnale composito (primo es.)



Matlab: trasformate e antitrasformate di Laplace

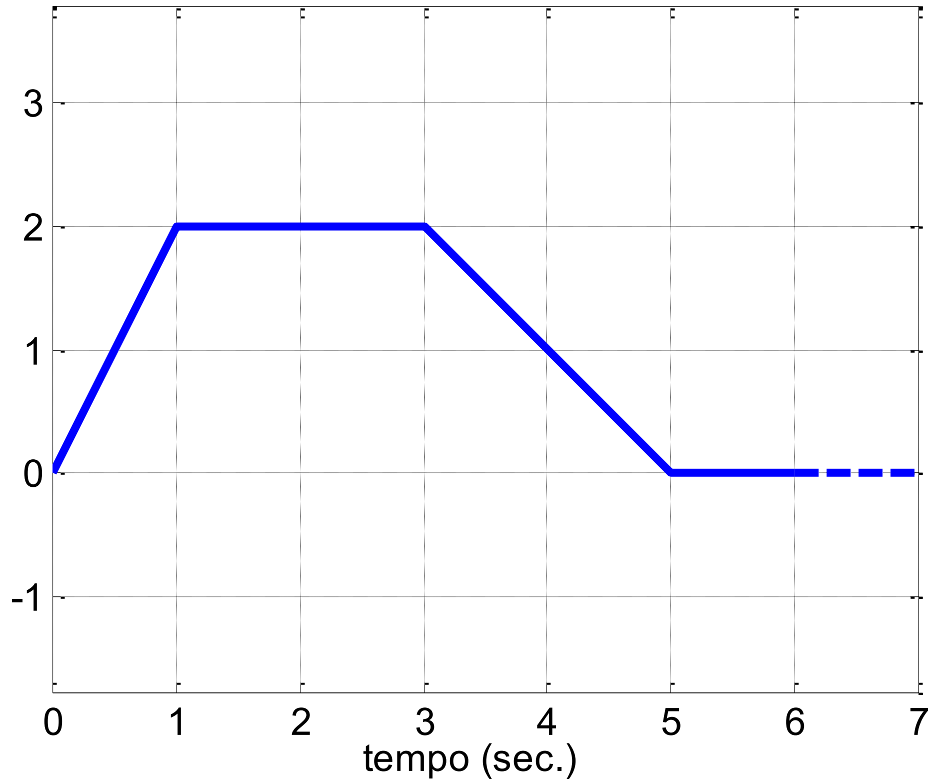
- Operazioni di base con **Symbolic Toolbox**:
  – Trasformata di segnale composito (primo es.)

```
>> F=laplace(heaviside(t-2)*(1 + (t-2)))
 F =
 (exp(-2*s)*(s + 1))/s^2
```

NOTA: anche il termine corrispondente alla rampa traslata di due secondi è moltiplicato per la funzione di Heaviside. Questo è necessario per mantenere la definizione del segnale coerente con l'intervallo di integrazione di Laplace (i.e. da 0 a $+\infty$)

Matlab: trasformate e antitrasformate di Laplace

- Operazioni di base con **Symbolic Toolbox**:
  - Trasformata di segnale composito (secondo es.)



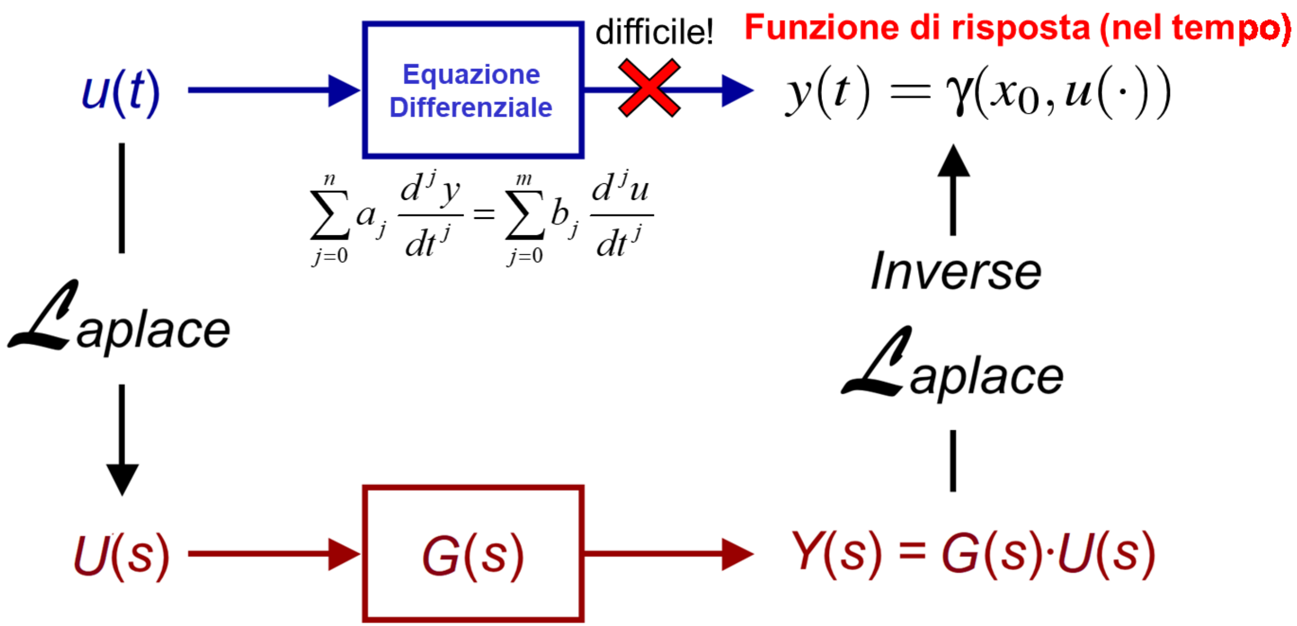
Matlab: trasformate e antitrasformate di Laplace

- Operazioni di base con **Symbolic Toolbox**:
  – Trasformata di segnale composito (secondo es.)

```
>> F=laplace(heaviside(t)*2*t + ...
          heaviside(t-1)*(-2)*(t-1) + ...
          heaviside(t-3)*(-1)*(t-3) + ...
          heaviside(t-5)*(t-5))
F =
exp(-5*s)/s^2 - exp(-3*s)/s^2 -
            (2*exp(-s))/s^2 + 2/s^2
```

Matlab: Laplace ed equazioni differenziali

- Si è visto nella parte 1 del corso che (con stato iniziali =0), la funzione di risposta è la convoluzione tra l'ingresso e la funzione di risposta impulsiva: $W(t) = Ce^{At}B$
- Con **Laplace**:



NOTA: considereremo solo sistemi puramente dinamici per limitare l'analisi a funzioni di risposta non impulsive a loro volta (i.e. se D≠0, W(t) include δ(t))

Matlab: Laplace ed equazioni differenziali

- Si considerino le matrici:

$$A = \begin{bmatrix} -3 & 0 \\ 1 & -6 \end{bmatrix} \quad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 1 \end{bmatrix}$$

- In Matlab:

```
>> A=[-3 0; 1 -6]
>> B=[1; 1]
>> C=[1 1]
```

Matlab: Laplace ed equazioni differenziali

- Con **Symbolic Toolbox:**

```
>> eA=expm(A*t)
eA =
[                    exp(-3*t),            0]
[ exp(-3*t)/2 - exp(-6*t)/2, exp(-6*t)]
>> W=C*eA*B
W =
(4*exp(-3*t))/3 + (2*exp(-6*t))/3
>> G=laplace(W)
>> G=collect(G)
G =
(2*s + 10)/(s^2 + 9*s + 18)
```

Matlab: Laplace ed equazioni differenziali

- Come descritto nelle dispense principali del corso (FdA-2.1-FunzioniTrasferimento), l'applicazione della trasformata di Laplace ai modelli differenziali nello spazio degli stati permette di ottenere la **matrice di trasferimento** **(o funzione di trasferimento, FdT, per sistemi SISO):**

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

$$\mathcal{L}[e^{At}] = (sI - A)^{-1}$$

NOTA: D = 0 nei sistemi puramente dinamici

Matlab: Laplace ed equazioni differenziali

- Con **Symbolic Toolbox**:

```
>> syms s
>> sA=inv(s*eye(2) - A)
```
🡐 eye(2)= identità 2x2..

```
sA =
[             1/(s + 3),           0]
[ 1/((s + 3)*(s + 6)), 1/(s + 6)]
>> G1=C*sA*B
G1 =
1/(s + 3) + 1/(s + 6) + 1/((s + 3)*(s + 6))
>> G1=collect(G1)
G1 =
(2*s + 10)/(s^2 + 9*s + 18)
```

Matlab: Laplace ed equazioni differenziali

- **Ovviamente,** l'operazione di antitrasformazione conferma la relazione tra le **rappresentazioni ingresso-uscita** (*tempo→ W(t) vs. Laplace→ G(s)*):

```
>>  W1=ilaplace(G1)
W1 =
(4*exp(-3*t))/3 + (2*exp(-6*t))/3
```

o anche:

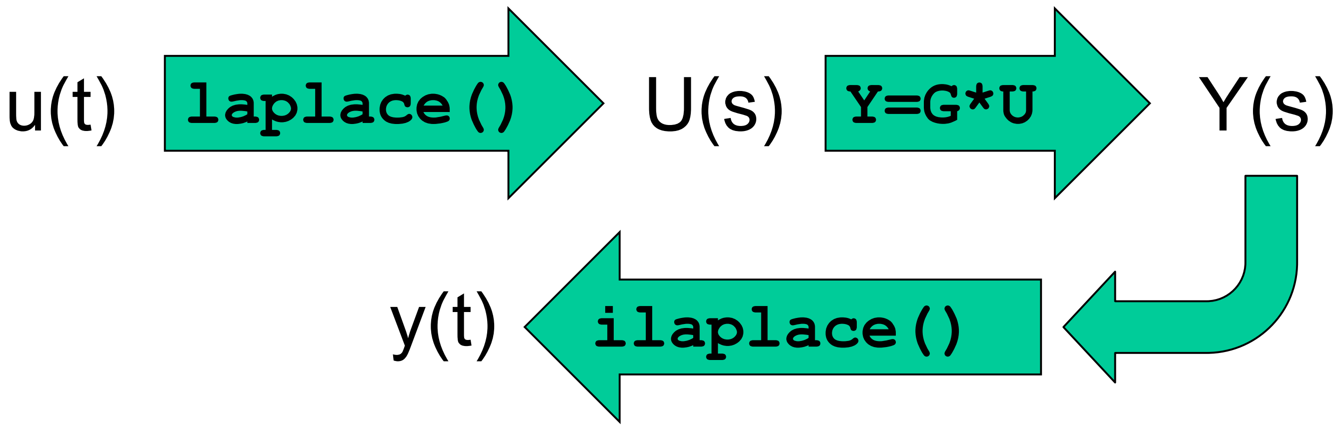
```
>>  W2=ilaplace(G)
W2 =
(4*exp(-3*t))/3 + (2*exp(-6*t))/3
```

Matlab: Laplace ed equazioni differenziali

- **NOTA:** il denominatore della funzione di trasferimento è certamente di grado inferiore a quello del denominatore (**funzione razionale strettamente propria**) per sistemi puramente dinamici!!
- Si ricordi inoltre che **sistemi fisicamente realizzabili** avranno funzioni di trasferimento proprie (i.e. grado del numeratore al più uguale a quello del denominatore)

Matlab: Laplace e risposta del sistema

- Gli strumenti di trasformazione e anti-trasformazione permettono di calcolare **l'espressione analitica della risposta** di un sistema rispetto a qualunque segnale, senza svolgere integrali di convoluzione (necessari invece nel dominio del tempo):



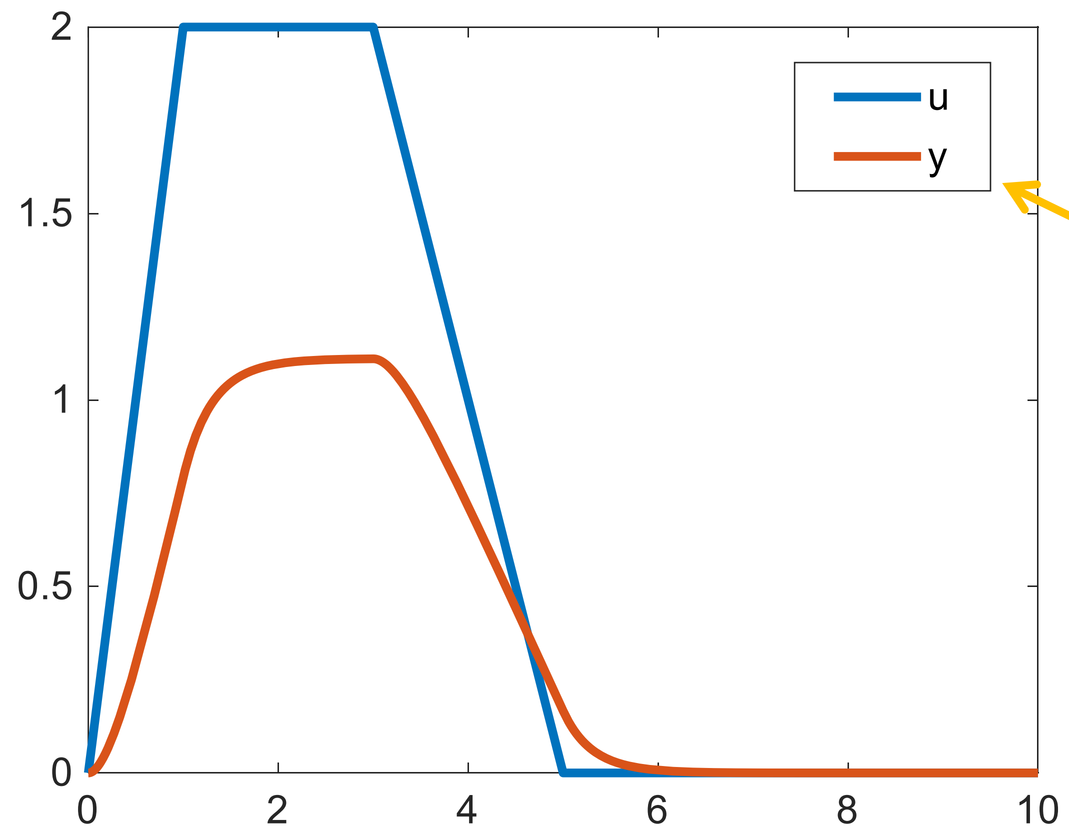
Matlab: Laplace e risposta del sistema

- Con **Symbolic Toolbox**:

```
>> u=heaviside(t)*2*t + ...
      heaviside(t-1)*(-2)*(t-1) + ...
      heaviside(t-3)*(-1)*(t-3) + ...
      heaviside(t-5)*(t-5)
>> U=laplace(u)
>> Y=G*U
>> y=ilaplace(Y)
>> fplot(u,[0 10])
>> hold on
>> fplot(y,[0 10])
```

Matlab: Laplace e risposta del sistema

- Grafico ottenuto con `fplot()` (estensione di `plot()` per il **Symbolic Toolbox**):



`legend('u','y')`

Matlab: Laplace e risposta del sistema

- **NOTA**: **`fplot()`** opera direttamente sull'espressione simbolica, mentre **`plot()`** opera solo su vettori numerici
- Per ottenere un grafico numerico, occorre sostituire nell'espressione simbolica un opportuno insieme di valori per le variabili indipendenti e convertire in **`double`**:

```
>> vals=subs(u,t,0:0.1:10)
>> vals=double(vals)
>> plot(vals)
```

Matlab: Laplace nel Control Systems Toolbox

- La rappresentazione del modello di un sistema con funzioni complesse di variabili complesse (i.e. *funzioni di trasferimento* → *transfer functions*) è parte integrante delle operazioni di analisi contenute nel **Control Systems Toolbox**, a fianco di quelle per sistemi LTI con modello nello spazio degli stati:

`ss(A,B,C,D)` ⟷ **`tf(NUM,DEN)`**

Matlab: Laplace nel Control Systems Toolbox

- Con **Control Systems Toolbox**:

```
>> sys=ss(A,B,C,0)
```
← D=0, necessario quarto parametro..

```
>> G=tf(sys)
Transfer function:
   2 s + 10
--------------
s^2 + 9 s + 18
```

Oppure anche, calcolando i coefficienti di numeratore e denominatore della FdT:

```
>> [Num,Den]=ss2tf(A,B,C,0)
Num = 0      2     10
Den = 1      9     18
>> G=tf(Num,Den)
```

Matlab: Laplace nel Control Systems Toolbox

- Oltre al passaggio alla funzione **`tf(num,den)`** dei due vettori contenenti i coefficienti della FdT, esiste un'alternativa comoda per definire la FdT con la struttura del **Control Systems Toolbox**:

```
>> s=tf('s')
```
← **"definisce" la variabile di Laplace**

```
>> Gc=10*(1+s)^2/s/(1+s/0.1)/(1+s/100)
```

NOTA: in questo caso `s` NON è una variabile simbolica, ma una vera e propria FdT rappresentata con la struttura dati corrispondente del Control Systems Toolbox..

Matlab: risposta della FdT con Control Systems Tlbx

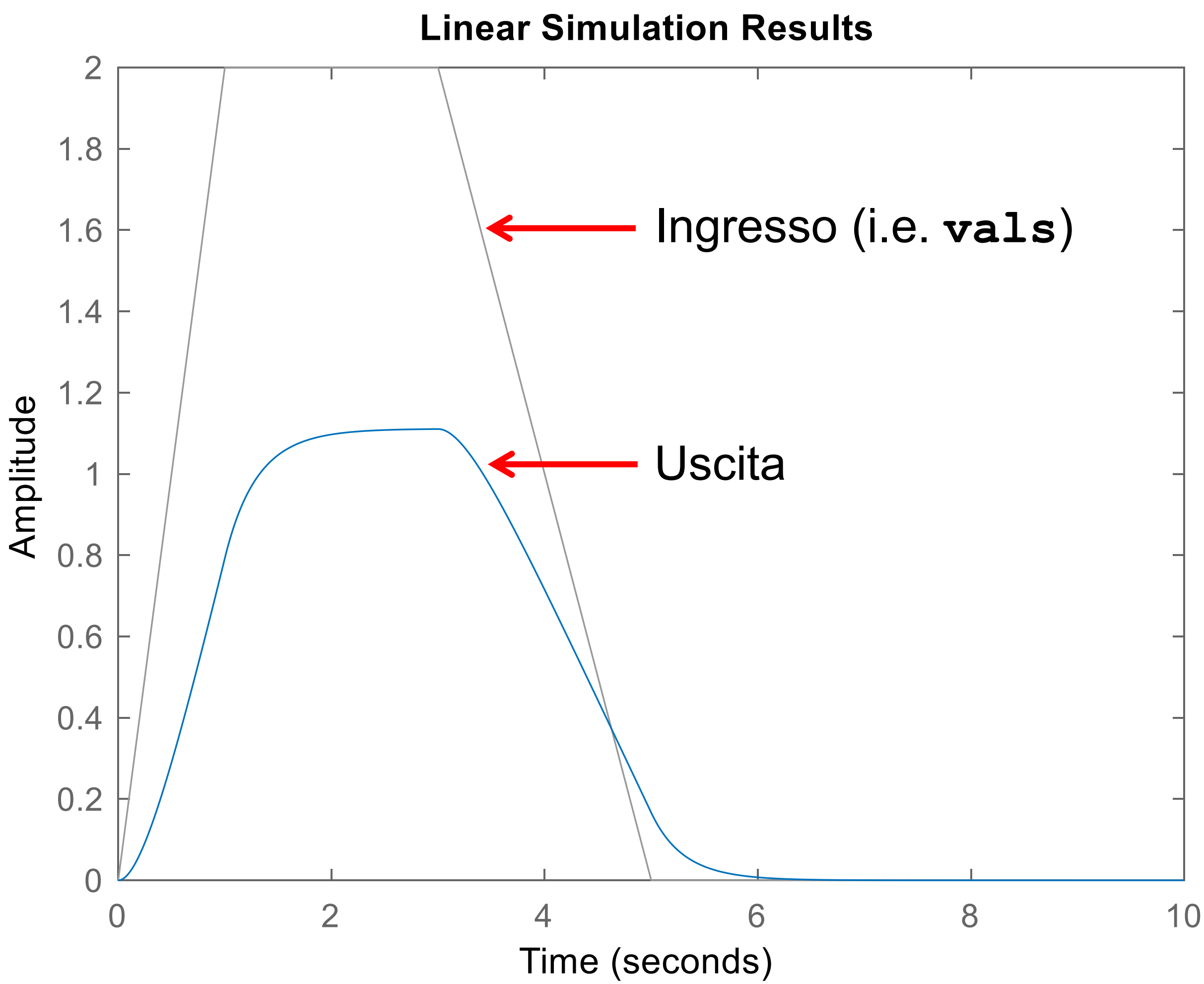
- Il comando **`lsim()`** già visto (vedere FdA-M.1-IntroMatlab) permette di calcolare la risposta di una FdT rispetto a qualunque segnale
- **NOTA:** se usato su un oggetto di tipo **`tf`**, il comando **`lsim()`** considera implicitamente stato iniziale x0=0, indipendentemente dal valore fornito come parametro:

(con **`vals`** come da slide 20-22)

```
>> timevals=0:0.1:10
>> lsim(G,vals,timevals)
```

Matlab: risposta della FdT con Control Systems Tlbx

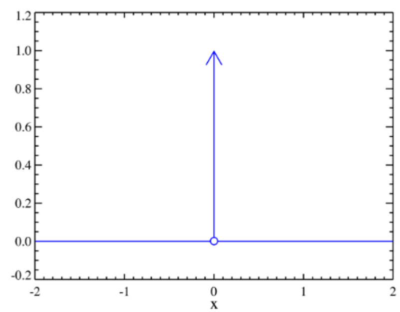
- Grafico ottenuto con **`lsim()`**:

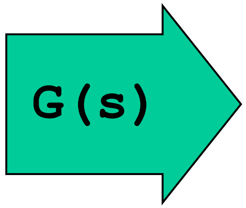


Matlab: risposta della FdT con Control Systems Tlbx

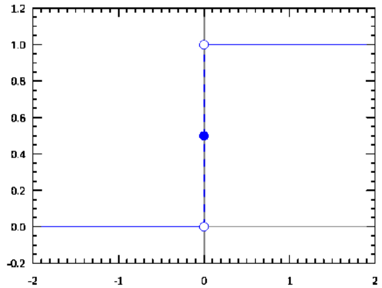
- La risposta a segnali tipici come l'impulso di Dirac o il gradino (i.e. **`dirac(t)`** e **`heaviside(t)`** nel **Symbolic Toolbox**) è direttamente ottenuta da funzioni specifiche del Control Systems Toolbox:

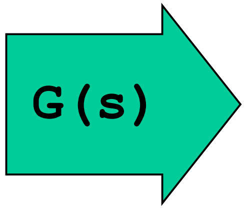
```
>> impulse(G)
```






```
>> step(G)
```





Matlab: risposta della FdT con Control Systems Tlbx

- Symbolic Toolbox:
  - Impulso:

```
>> W=C*eA*B
>> fplot(W,[0 2])
```

  - Gradino:

```
>> Gs=laplace(W)
>> Us=1/s
>> Ys=G*Us
>> yt=ilaplace(Ys)
>> fplot(yt,[0 2])
```

- Control Systems Tlbx:
  - Impulso:

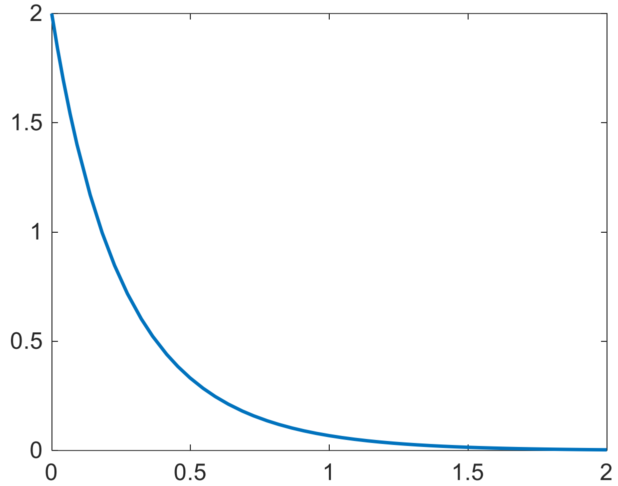
```
>> impulse(G)
```

  - Gradino:

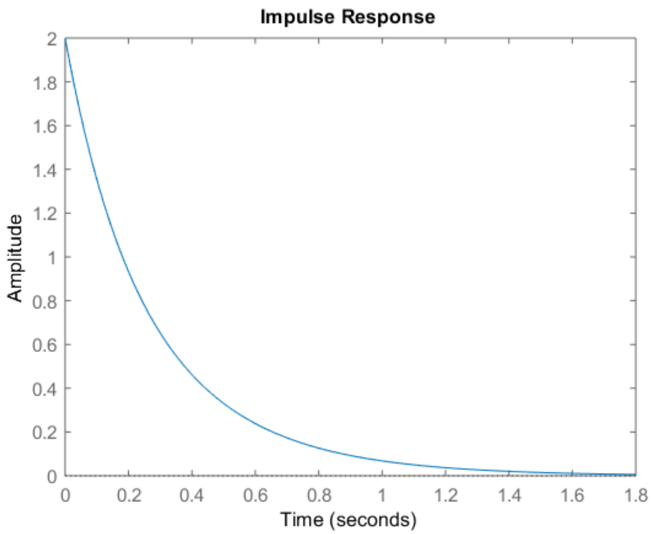
```
>> step(G)
```

Matlab: risposta della FdT con Control Systems Tlbx

- Symbolic Toolbox:
  - Impulso:

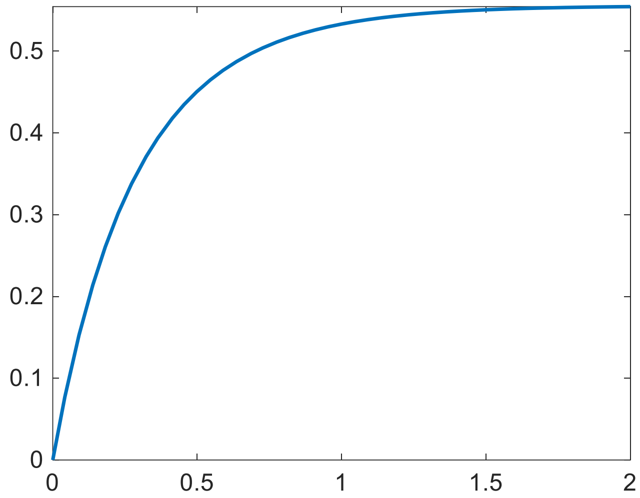


- Control Systems Tlbx:
  - Impulso:

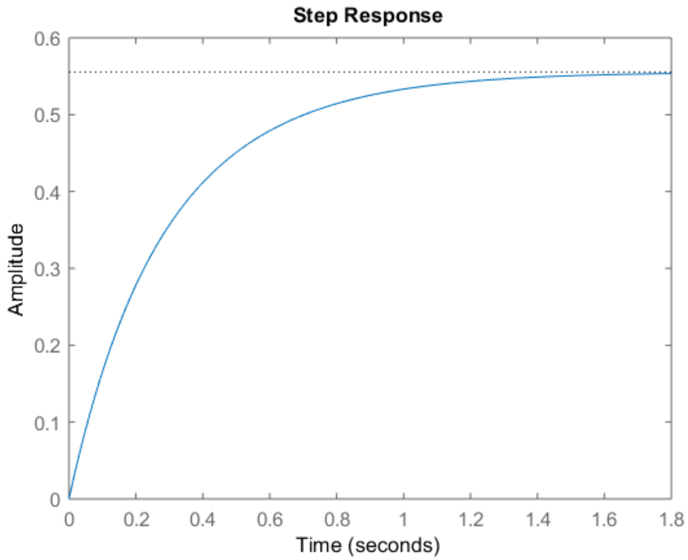


Matlab: risposta della FdT con Control Systems Tlbx

- Symbolic Toolbox:
  - Gradino:

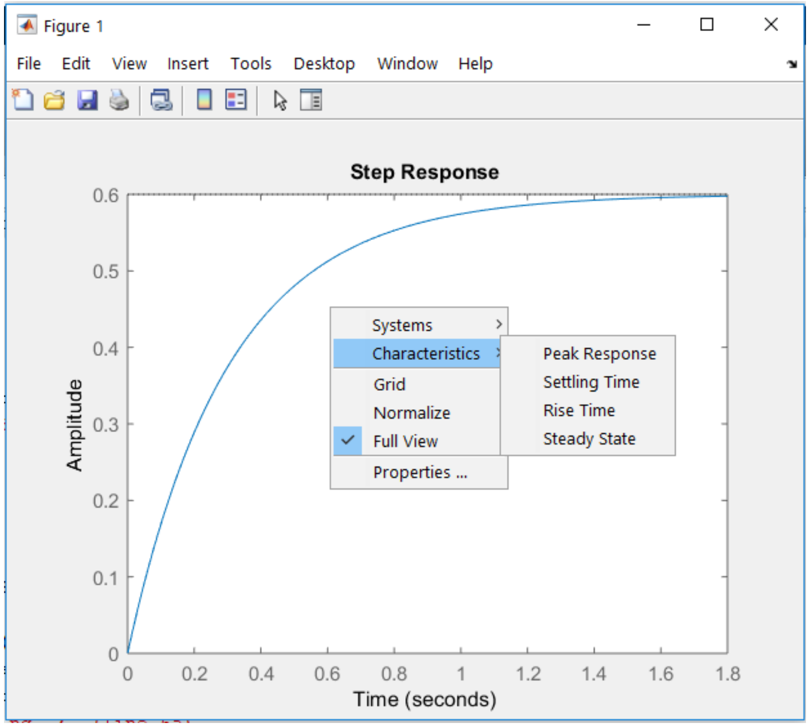


- Control Systems Tlbx:
  - Gradino:



Matlab: risposta della FdT con Control Systems Tlbx

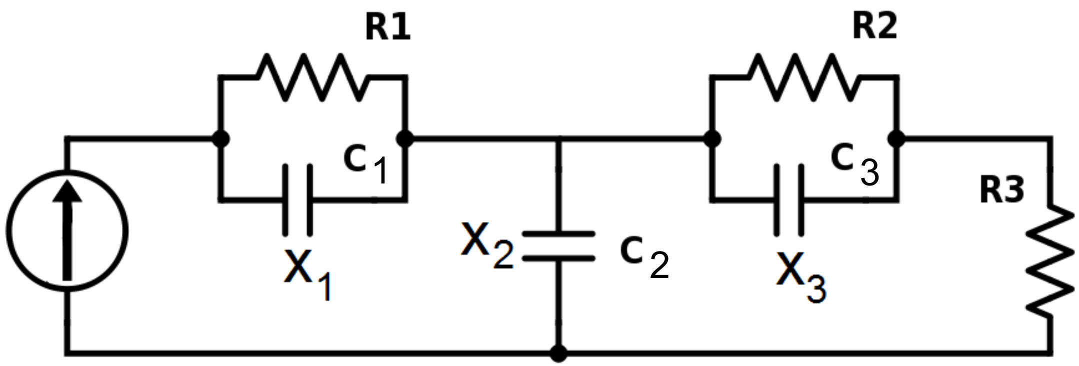
- **NOTA:** il grafico ottenuto con il Control Systems Toolbox è interattivo e molto più ricco di funzionalità rispetto al grafico *standard* del Symbolic Math Toolbox
- Si verifichino le funzionalità supportate tramite il click con tasto destro del mouse…



ESERCIZIO RIEPILOGATIVO:
da equazioni differenziali a State-Space e a Transfer Function

Esempio riepilogativo: circuito RC multimaglia

- Si consideri il seguente circuito (con $y = x_3$):



$$C_1\dot{x}_1 + \frac{x_1}{R_1} = u$$

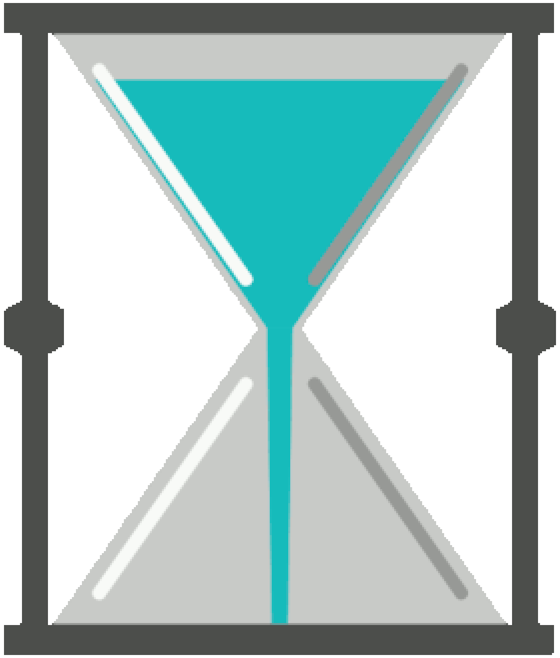
$$C_2\dot{x}_2 + \frac{x_2 - x_3}{R_3} = u$$

$$C_3\dot{x}_3 - \frac{x_2 - x_3}{R_3} + \frac{x_3}{R_2} = 0$$

1. Trovare le matrici A,B,C,D del modello State-Space
2. Con: $C_1 = C_2 = C_3 = 0{,}1$;
   $R_1 = R_2 = 10$; $R_3 = 5$;
   costruire il modello **ss()**
3. Costruire la FdT del sistema con `tf()`

Esercizio riepilogativo: circuito RC multimaglia

Let's do it!!



Passaggio alla funzione di trasferimento

- Si noti il grado dei polinomi nella FdT e lo si confronti con l'ordine di partenza del sistema
- NOTA: **con A 3x3 si dovrebbe ottenere una FdT con denominatore di grado 3…**

Passaggio alla funzione di trasferimento

- Il sistema considerato **NON** è **completamente raggiungibile-controllabile** e **completamente osservabile-ricostruibile!!**
- Per tale motivo, i poli della FdT (i.e. radici del denominatore) sono un sottoinsieme degli autovalori di A:

```
>> eig(A)                  >> pole(G)
ans =                      ans =
   -4.5616                    -4.5616
   -1.0000                    -0.4384
   -0.4384
```

Matlab: test di controllabilità / osservabilità

- Nel caso considerato, risulta:

```
>> P=ctrb(A,B)
P = 10    -10     10
    10    -20     80
     0     20   -100
>> rank(P)
ans = 3
>> Q=obsv(A,C)
Q =  0      0      1
     0      2     -3
     0    -10     13
>> rank(Q)
ans = 2
```

NOTA: sistema completamente controllabile, MA NON completamente osservabile…

TRASFORMATE DI LAPLACE IN MATLAB

FINE