CONTROLLI AUTOMATICI
Ingegneria Meccanica e Ingegneria del Veicolo

http://www.dii.unimore.it/~lbiagiotti/ControlliAutomatici.html

INTRODUZIONE A SIMULINK

Ing. Luigi Biagiotti

e-mail: luigi.biagiotti@unimore.it

http://www.dii.unimore.it/~lbiagiotti

Programma della lezione

- Cos’è Simulink e struttura del programma
- Librerie principali
- Costruire e lanciare una simulazione
- Inserire funzioni si trasferimento in Simulink
- Costruire modelli più complessi non basati su funzioni di trasferimento

Simulink

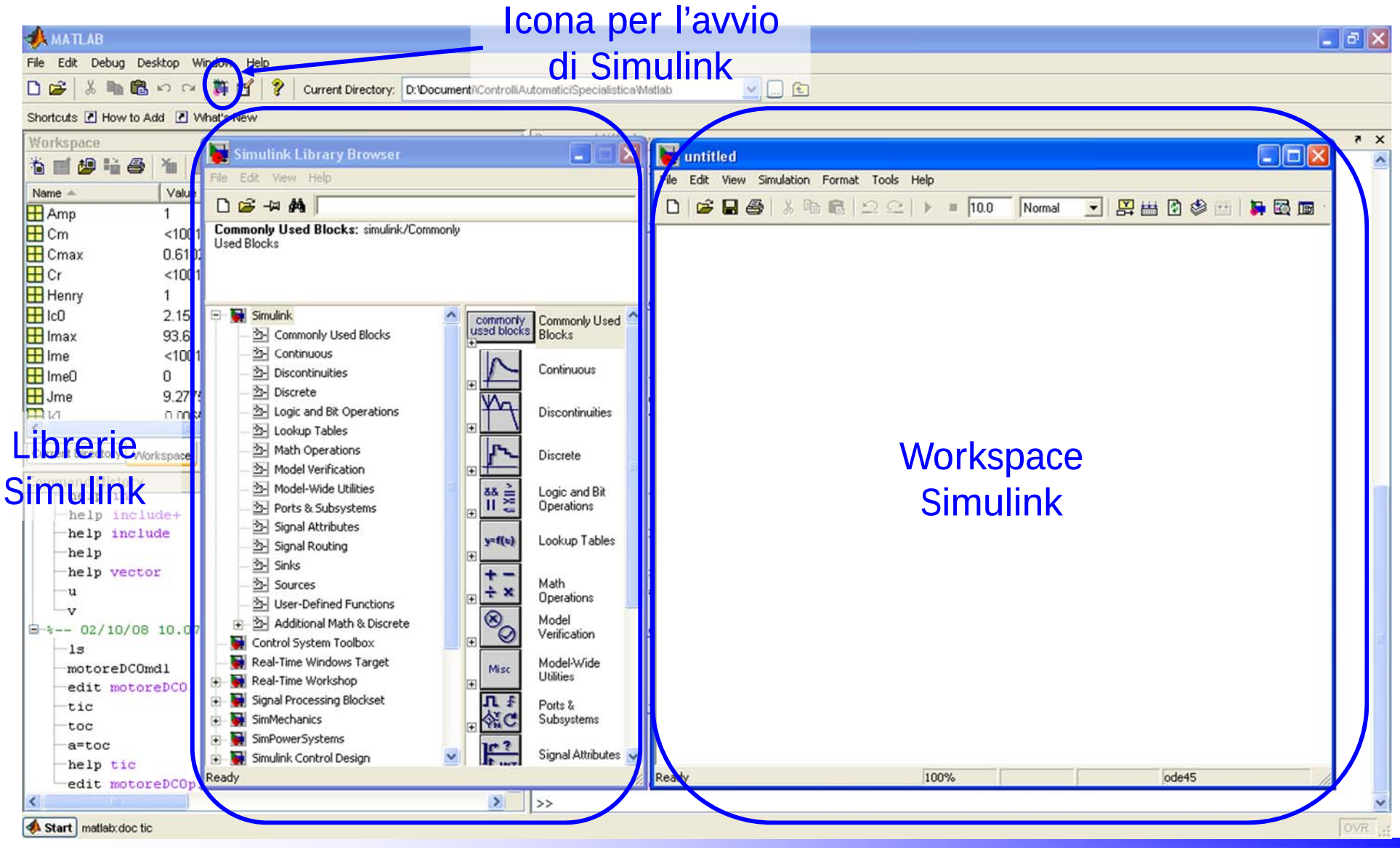
- Simulink è un programma costruito utilizzano i comandi di Matlab

- **Vantaggi:**
  - Interfaccia grafica
  - Blocchi predefiniti solamente da connettere
  - Elevata flessibilità nella variazione del progetto
  - Riduzione dei tempi di progetto
  - Riduzione dei costi rispetto a un test pratico
  - Condivisione con Matlab di tutte le variabili definite nel workspace

- **Per accedere a Simulink basta digitare *simulink* dal prompt di matlab**

Simulink

- All'avvio si possono distingure due parti: le **Librerie** e il **Workspace**.



Simulink

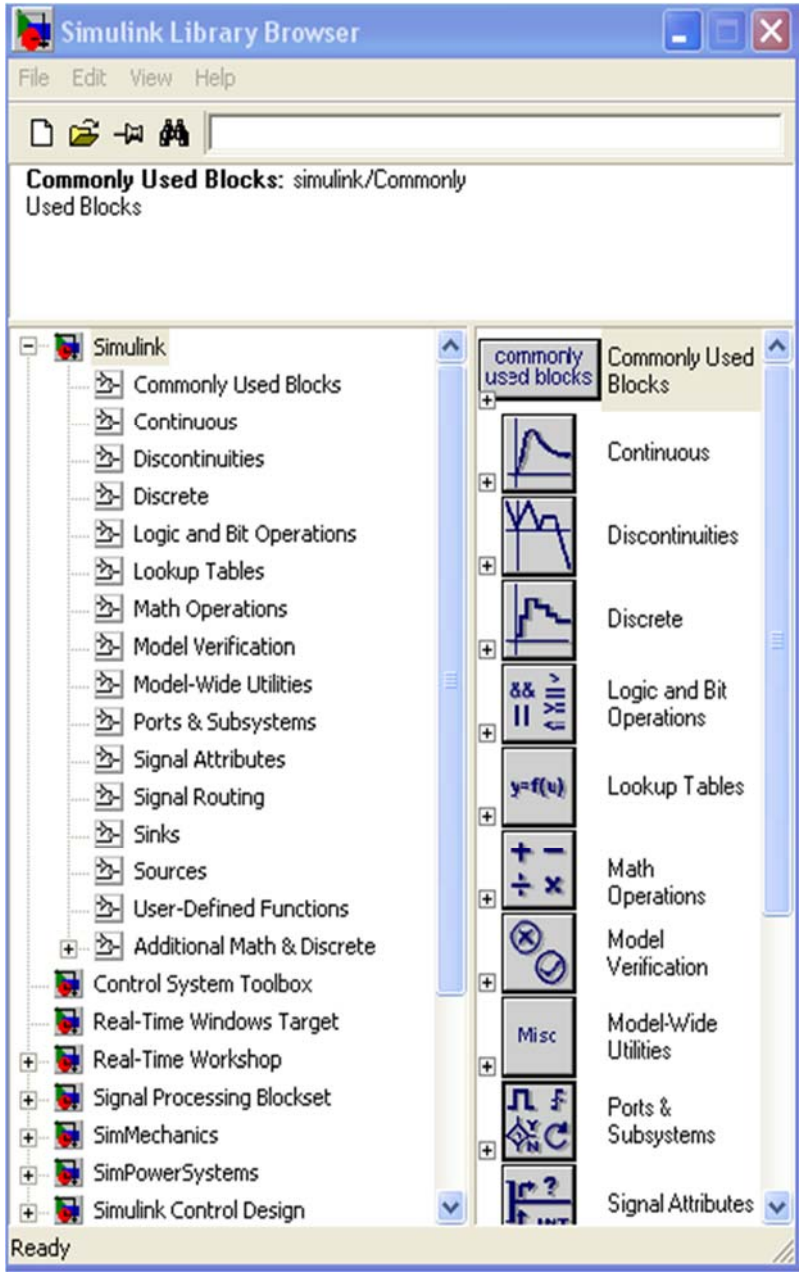
- Nelle *librerie* sono presenti i blocchi elementari che possiamo usare nel progetto.
- Nel *workspace* si costruisce il progetto interconnettendo i blocchi presi dalla librerie.

- I vari elementi si portano nel workspace semplicemente trascinandoveli dentro come se fossero icone.
- Le librerie sono **Read-only**. Per poter variare i parametri di un blocco occorre prima trascinarlo nel workspace.
- Facendo doppio click sull'icona trascinata nel workspace si apre una maschera che ci consente di impostare i parametri che caratterizzano il segnale

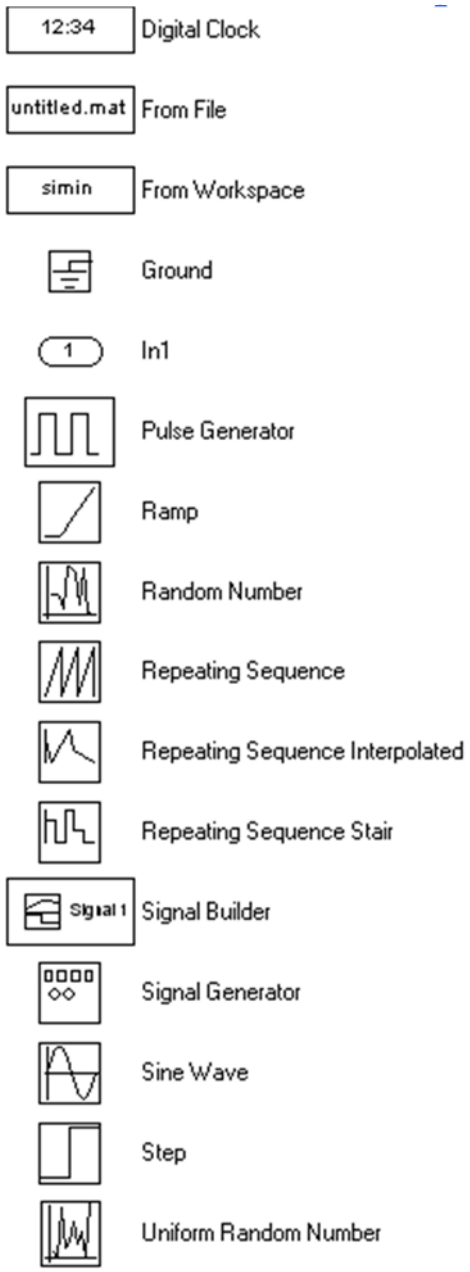
Simulink

Ci sono svariate librerie, noi useremo principalmente:

- *Sources:* Blocchi che generano segnali di vario genere
- *Sinks:* Blocchi per la visualizzazione grafica dei segnali
- *Math:* Blocchi per l'elaborazione matematica dei segnali
- *Continuous:* Blocchi per l'inserimento di funzioni di trasferimento

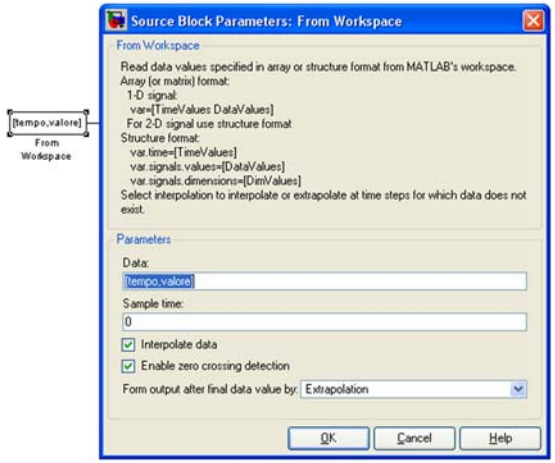


Simulink – Libreria Sources



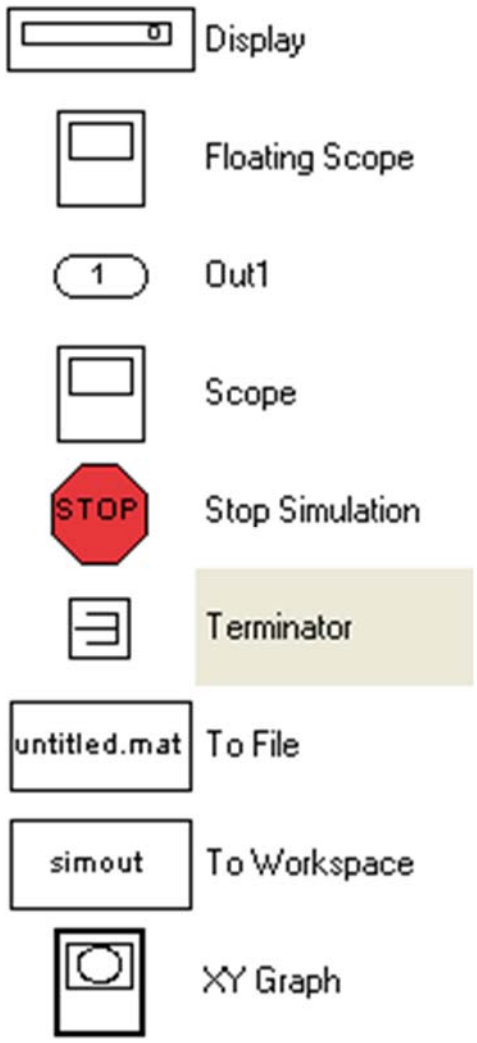
I blocchi più utilizzati sono:

- **Constant:** genera un valore costante.
- **Step:** genera un gradino.
- **Ramp:** genera una rampa.
- **Sine wave:** genera una sinusoide.
- **From workspace:** il riferimento può essere generato in precedenza nel workspace e passato come **`[tempo, valore],`** dove tempo e valore sono due vettori colonna di egual lunghezza



- **Repeating sequence**
- **Clock:** Scandisce gli istanti di tempo della simulazione

Simulink – Libreria Sinks



- Un insieme di strumenti che consente di visualizzare l'andamento di un segnale.
- I blocchi più importanti sono:
  - **Scope:** Visualizza il segnale di ingresso in funzione del tempo. (attenzione all'opzione **`limit data points to`**...)
  - **XYGraph:** Genera un grafico del segnale connesso all'ingresso y (il secondo) in funzione di quello connesso all'ingresso x (il primo).
  - **To Workspace:** Memorizza i valori del segnale connesso in una variabile matlab (nota bene: **`save format array`**).

Simulink – Libreria Sinks

- Si consiglia di utilizzare **To Workspace** in quanto dopo la simulazione si dispone non solo di un grafico ma di una variabile in cui sono contenuti tutti i valori assunti da un segnale. Si può elaborare poi tale variabile con gli strumenti messi a disposizione da Matlab.

- Per visualizzare l'andamento rispetto al tempo delle variabili, è necessario salvare in un'ulteriore variabile un vettore che scandisca gli istanti temporali della simulazione. Questo è possibile inserendo il blocco **clock** e collegandone l'uscita a un blocco **To Workspace** nello schema simulink.

Simulink – Funzioni di trasferimento

- Per inserire una funzione di trasferimento nello schema Simulink si utilizzano i blocchi presenti nella libreria **Continuous**:

  - **Transfer Fcn:** consente di editare una funzione di trasferimento immettendo il numeratore e il denominatore. Numeratore e denominatore sono rappresentati da due vettori che esprimono i coefficienti, secondo potenze discendenti di *s*, del polinomio corrispondente.

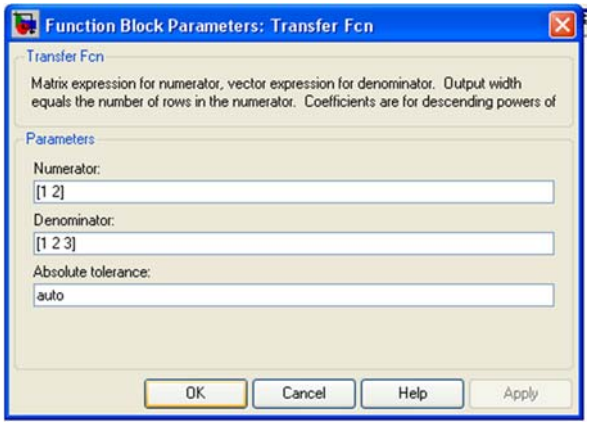
  - **Zero-Pole:** consente di editare una funzione di trasferimento specificando i suoi zeri e i suoi poli. Numeratore e denominatore sono rappresentati da due vettori i cui elementi rappresentano rispettivamente gli zeri e i poli della funzione di trasferimento.

- Se la funzione da inserire è un semplice integratore è già presente il blocco che lo implementa.

Esempio

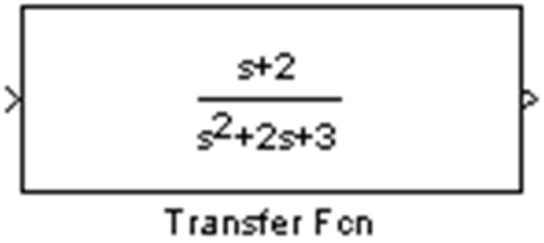
Se inseriamo nella maschera i vettori:

- *Numerator:* [1 2]
- *Denominator: [1 2 3]*

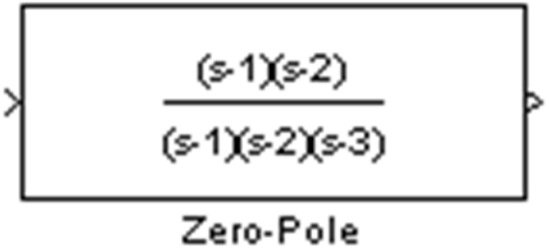


Otteniamo rispettivamente:

Transfer Fcn:



Zero-Pole:



Simulink – Lanciare una simulazione

- Una volta costruito il sistema da simulare occorre far partire la simulazione.

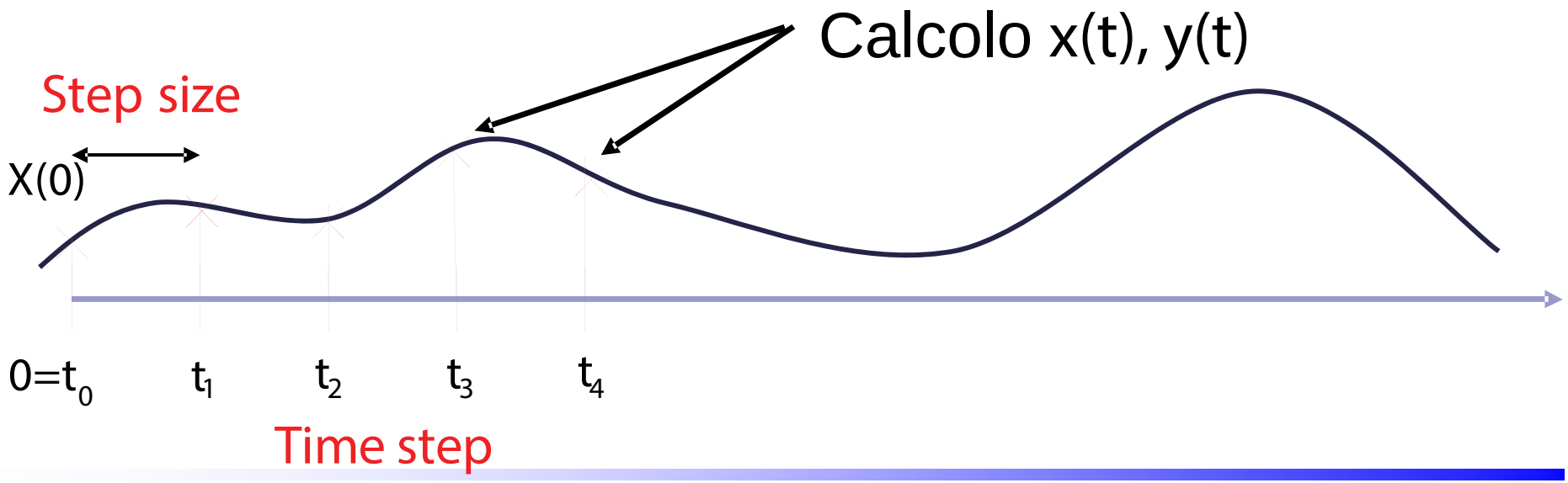
- Per lanciare la simulazione si può fare in uno dei seguenti modi:
  - Premere il tasto a forma di ***Play*** sulla toolbar
  - Selezionare ***Start*** dal menu ***Simulation***

- Per cambiare i parametri della simulazione (tra cui il tempo di simulazione e i metodi di integrazione numerica) selezionare ***Simulation parameters*** dal menu ***Simulation*.**

Simulink – Simulation parameters

- Simulink risolve equazioni differenziali ordinarie (definite per esempio utilizzando f.d.t.) mediante **tecniche di integrazione numerica**

- Simulazione di un sistema dinamico:
  - Calcolo dell'evoluzione dello stato e dell'uscita del sistema per una certa durata temporale
  - Lo stato e l'uscita vengono calcolati per certi istanti (detti time step) separati da intervalli di integrazione (detti step size)



Simulink – Metodi di integrazione numerica

Esistono diversi metodi di integrazione numerica (detti *solver):*

- Metodi di integrazione a passo di integrazione fisso (lo step size è costante)
  - Più è piccolo lo step size, più la simulazione è accurata, ma un tempo maggiore è richiesto per l'esecuzione della simulazione
- Metodi di integrazione a passo di integrazione variabile
  - il solver determina lo step size ottimale durante la simulazione (se la dinamica del sistema è veloce, lo step size si riduce)

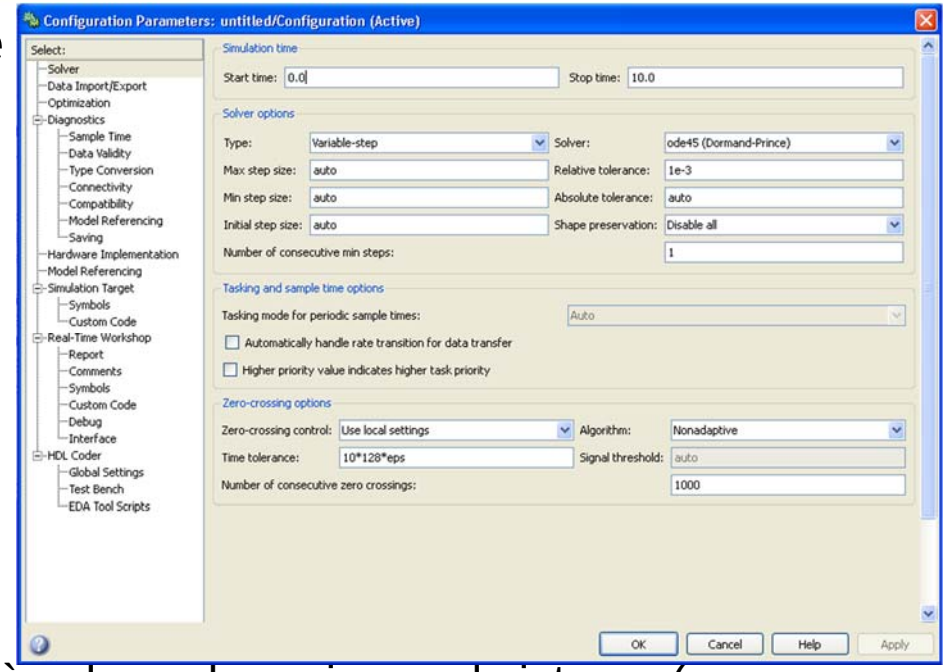
- **ODE45 è il risolutore "standard"**

Impostare I parametri della simulazione

- Istante iniziale e finale della simulazione
- Tipo della tecnica di integrazione numerica (es. variable step, ODE45)
- **Max step size** (massimo intervallo fra un istante di calcolo della soluzione dell'ODE e il successivo)

  DEVE essere scelto:

  - Minore della costante di tempo più veloce del sistema
  - Minore del periodo del segnale periodico più veloce che agisce sul sistema (es. 1/10 più piccolo)



- **Min step size** (minimo intervallo fra un istante di calcolo della soluzione dell'ODE e il successivo)

  Può essere utilizzato per ridurre il tempo richiesto per la simulazione (a discapito della precisione)

- **Relative/absolute tollerance** Accuratezza del risultato ottenuto con la simulazione (impostare "auto" o a valori sufficientemente piccoli, es. 1E-4)

Struttura dei modelli in Matlab-Simulink

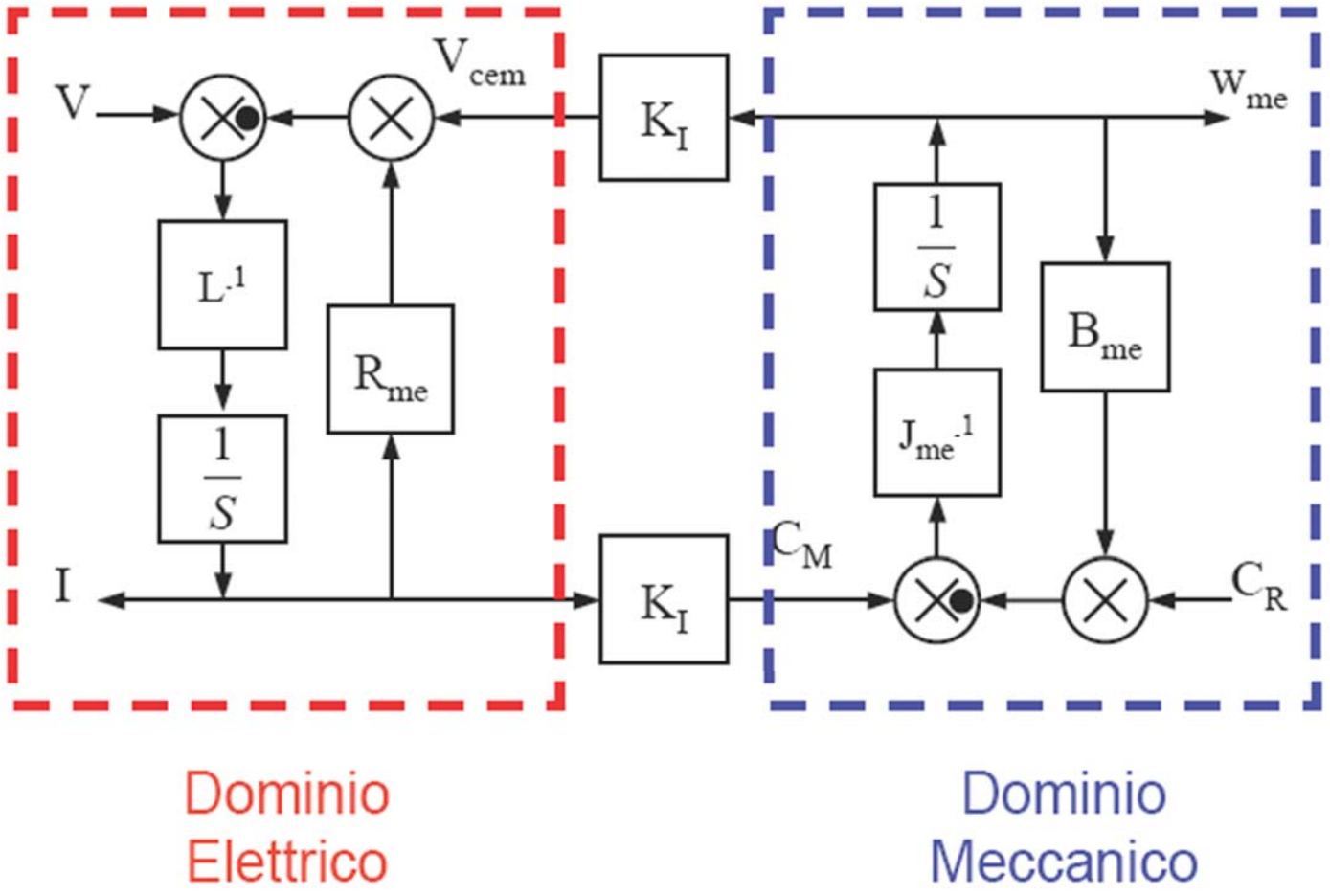
I modelli realizzati con Matlab-Simulink sono strutturati solitamente in due file principali:

- Modello Simulink (es. **`motoreDC0mdl.mdl`**)
  - Contiene il modello del sistema da simulare costruito con i blocchi di Simulink.
  - Memorizzai segnali della simulazione.

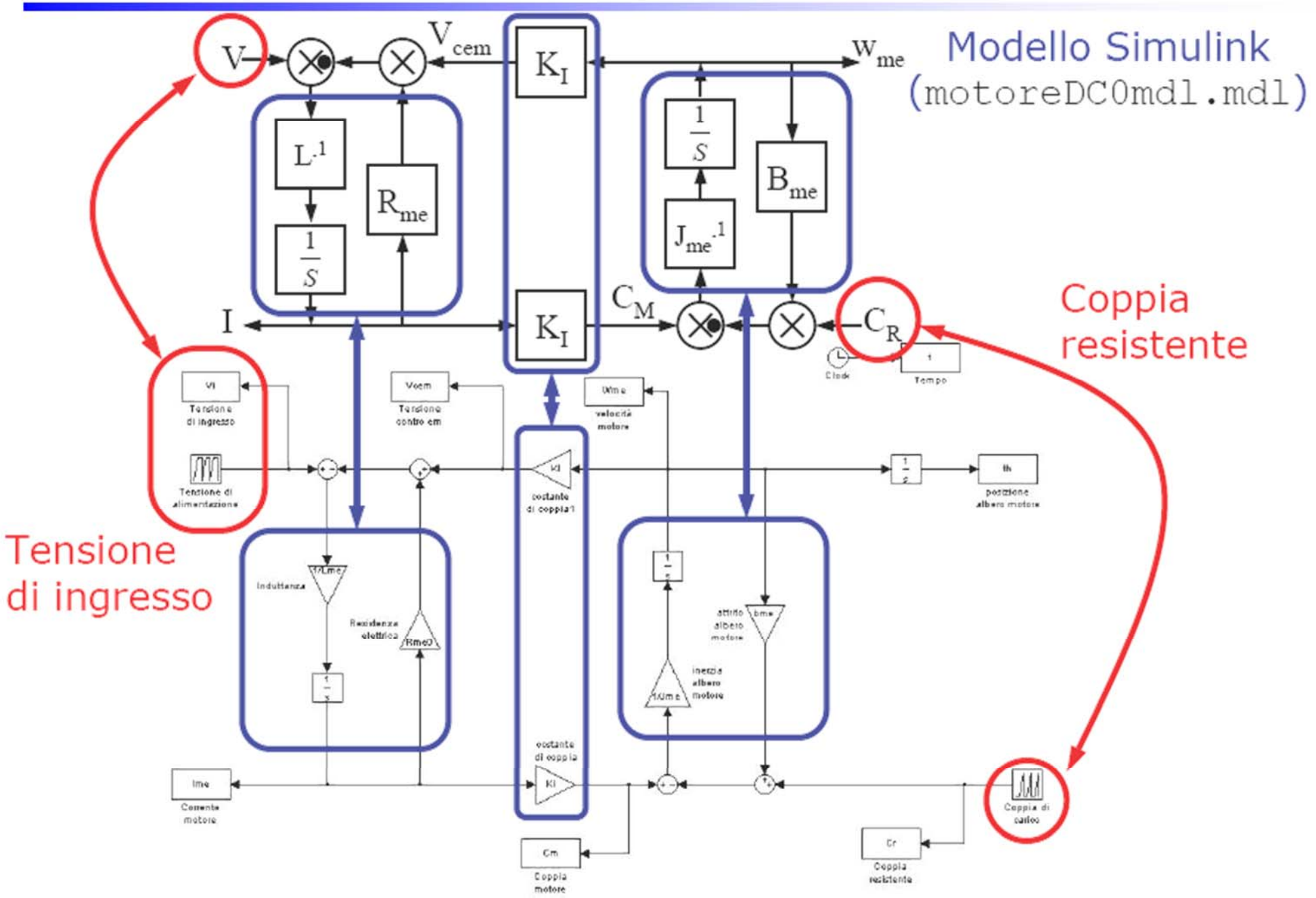
- File (di comandi) dei parametri del modello e della simulazione (es. **`motoreDC0.m`**).È il file principale per chiamare la simulazione. Contiene:
  - (Conversioni delle unità di misura)
  - Parametri del modello
  - Condizioni iniziali della simulazione
  - Segnali di ingresso del sistema
  - Graficazione dei risultati (che può essere svolta da una funzione aggiuntiva o da un file comandi aggiuntivo, es. **`motoreDC0plot.m`**)

NOTA BENE: per chiarezza e comodità, è meglio evitare di inserire valori numerici direttamente nello schema Simulink, sebbene sia possibile.

Modello dinamico (POG) del motore in corrente continua

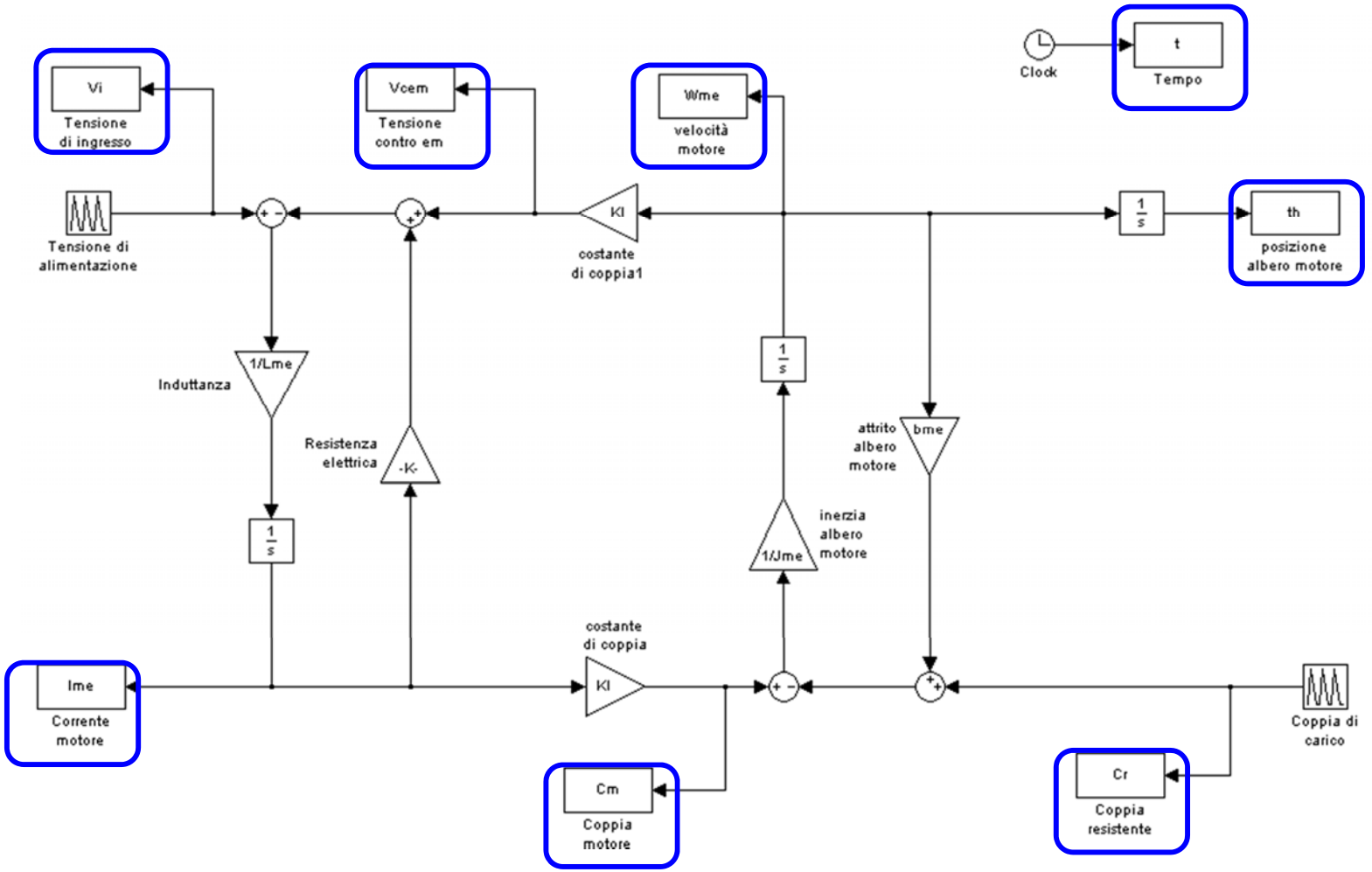


Dal modello dinamico allo schema simulink



Salvataggio dei segnali di una simulazione

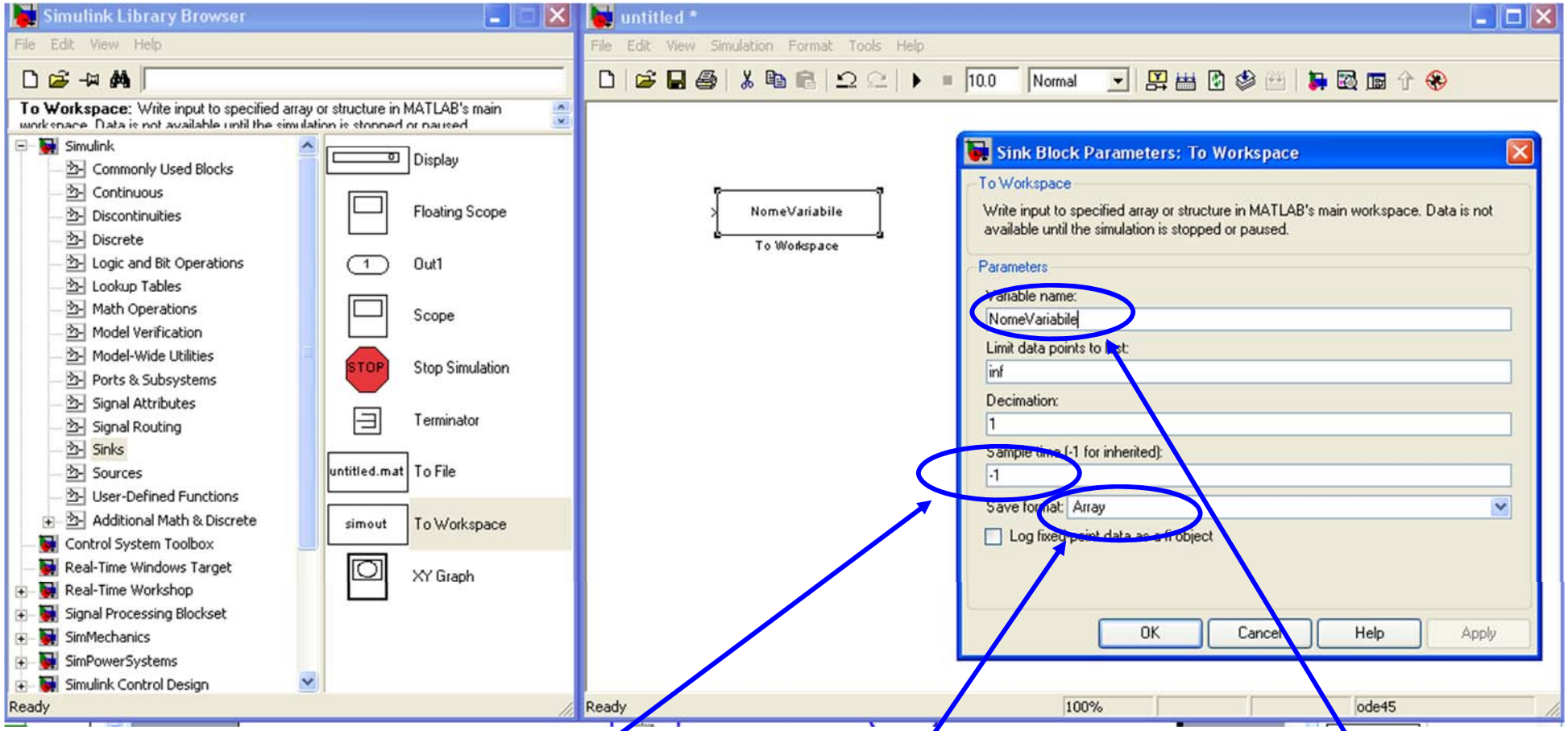
- I segnali simulati sono “misurati” e memorizzati in variabili di tipo array che saranno utilizzate per rappresentare i risultati.



- Si è utilizzato il blocco Sinks -> To Workspace che salva i segnali come variabili nel workspace

Il blocco To Workspace

- Il blocco To Workspace permette di definire:



Tempo di campionamento dei dati della simulazione:

-1 = definito da Simulink

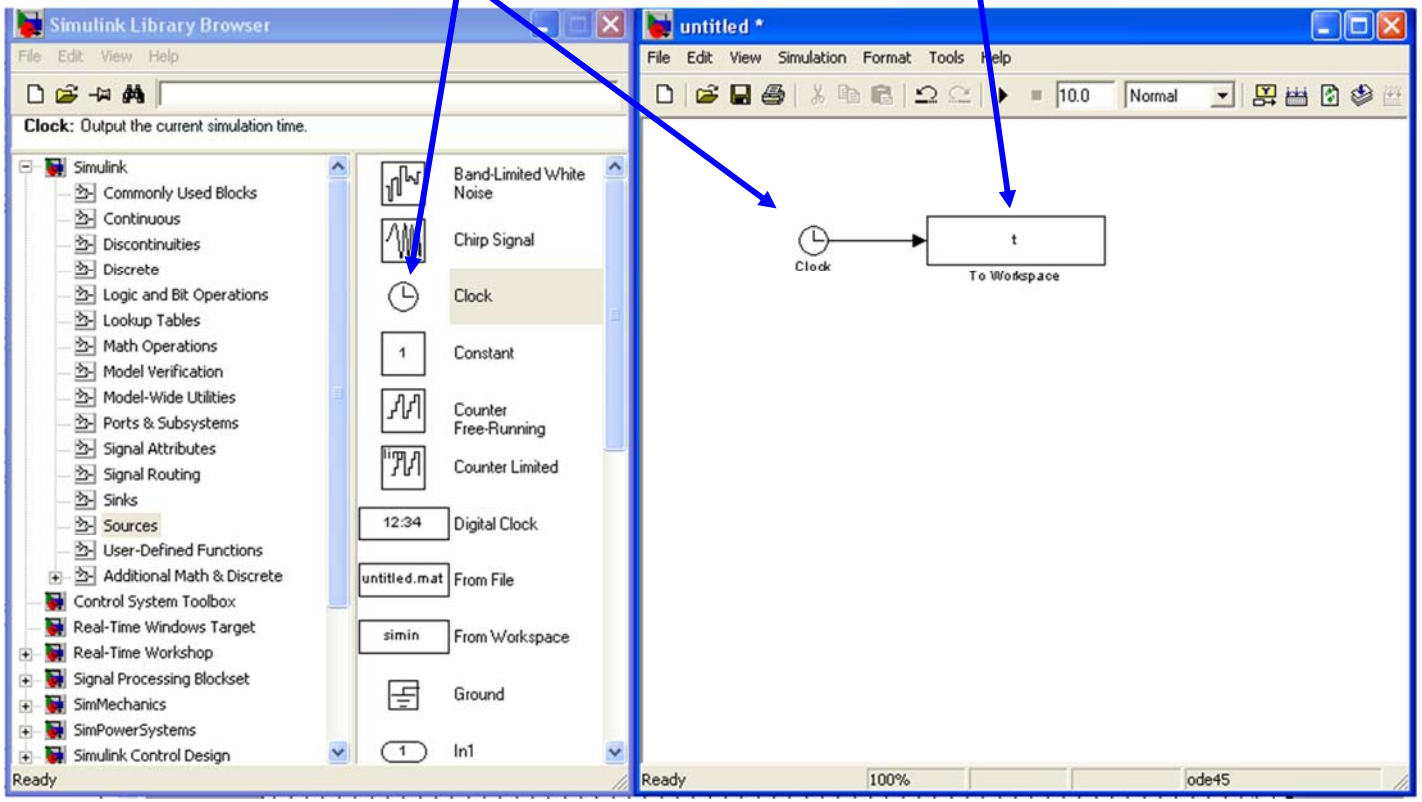
Ts>0 = definito dall'utente

Dati salvati come **vettori** o strutture

Nome della variabile

Memorizzazione degli istanti di tempo di simulazione

- Gli istanti di tempo che corrispondono ai dati memorizzati si ottengono usando la sorgente “Clock” e il blocco “ToWorkspace”



- La variabile **`t`** è un vettore colonna con *n* righe.
  - Tutti i dati memorizzati tramite i blocchi “ToWorkspace” hanno *n* righe e un numero di colonne pari alla dimensione della variabile.

File dei parametri

- La struttura consigliata per i file dei parametri è la seguente:
  - Definizione delle unità di misura e delle conversioni
  - Parametri del modello
  - Caricamento dei dati sperimentali (qualora ve ne fossero)
  - Condizioni iniziali della simulazione
  - Segnali di ingresso della simulazione
  - Parametri della Simulazione e Simulazione
  - Graficazione dei risultati della simulazione

- È tipicamente conveniente (specialmente quando le simulazioni richiedono molto tempo) gestire separatamente un file di comandi per la graficazione dei risultati delle simulazioni (**`motoreDC0plot.m`**)

File dei Parametri – Definizione delle unità di misura e conversioni

- Le unità di misura consigliate sono quelle del sistema internazionale. Per facilitare le conversioni nel sistema internazionale i fattori di conversione sono riportati all'inizio del file dei parametri **motoreDC0.m**

```
% Definizione delle unità di misura del SI
m=1; Km=1000*m; cm=0.01*m; mm=0.001*m;
s=1;msec=0.001*s; minuti=60*s; ora=60*minuti;
Kg=1;gr=0.001*Kg; mg=0.001*gr;
N=1;Nm=N*m;mNm=Nm/1000;rad=1;
gradi=pi*rad/180;
g = 9.81*m/s^2;% accelerazione di gravità
Kgf=g;%Kilogrammiforza
rpm=2*pi/60;
Amp=1;mAmp=0.001*Amp; V=1;
Ohm=V/Amp;
Henry=V*s/Amp;mHenry=0.001*Henry;
```

- Esempio: scrivendo

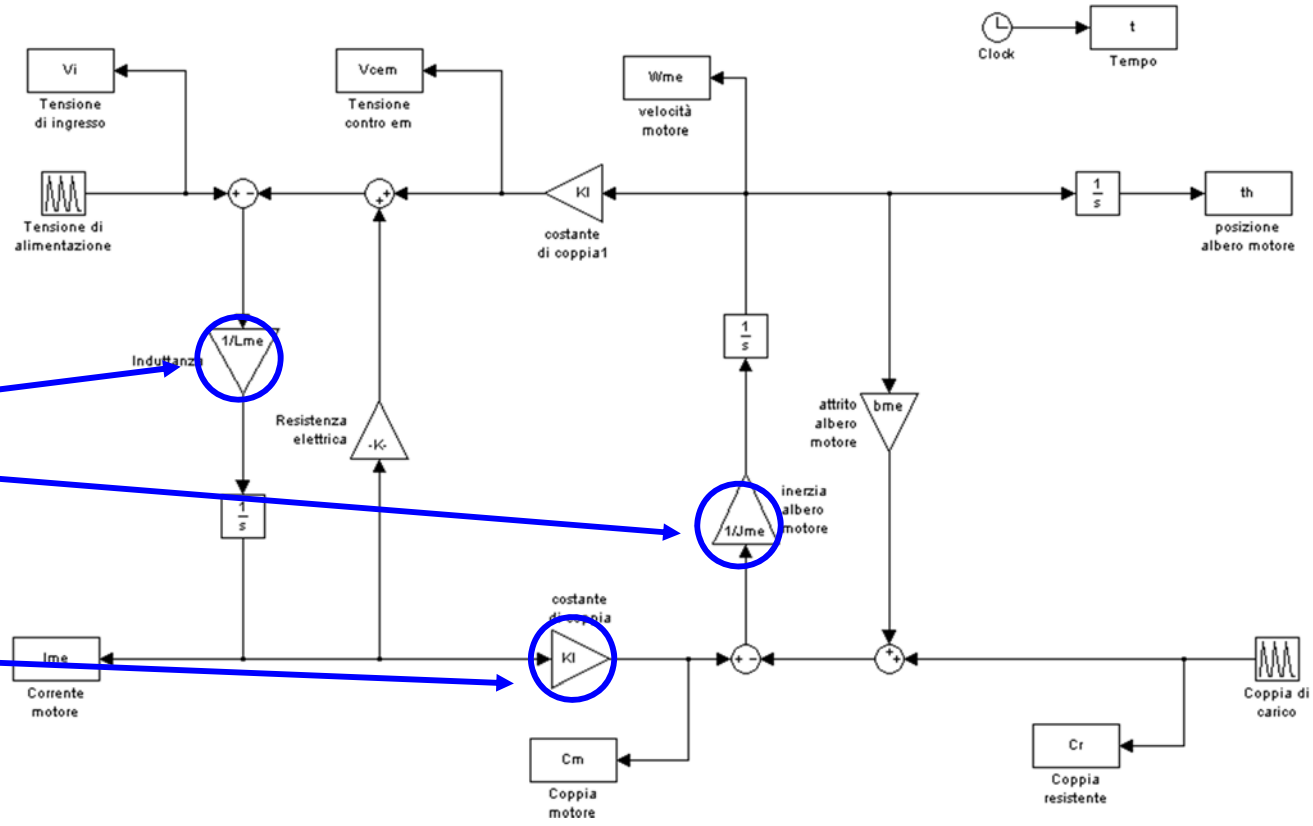
```
Vmax= 10000*rpm;
```

si ottiene vmax con valore in [rad/s] (unità SI) pur avendolo scritto in [rpm]

File dei parametri e parametri del modello

Tutti i parametri del modelloSimulinksono contenuti nel file dei parametri (**motoreDC0.m**)

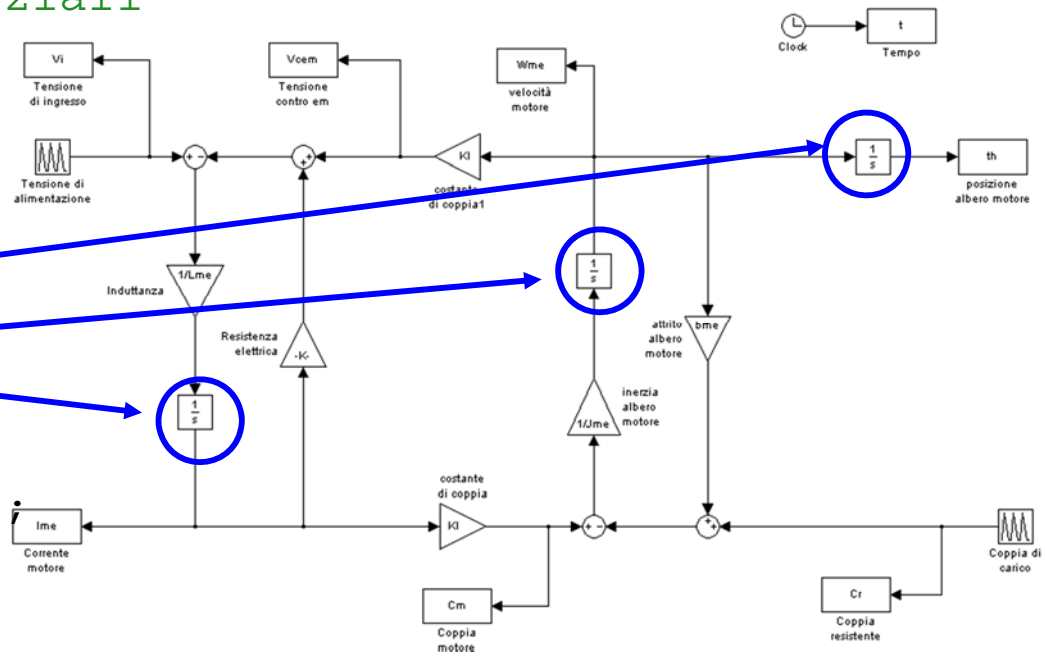
```
% Parametri del motore
tipo=1;
Switch tipo
Case 1
    Vn=12*V;
     Cmax=610.24*mNm;
     Imax=93.6*Amp;
     Wmax=16313*rpm;
     Ic0=2.15*Amp;
     Lme=60*10^-6;
     Jme=9.2775e-006;
case 2...end
% calcolo parametri
del modello
KI=Cmax/Imax;
Rme0=Vn/Imax;
bme=Ic0*KI/Wmax;
```



File dei parametri e condizioni iniziali

Ad ogni **integratore** del modello corrisponde una **condizione iniziale**
Tutte le condizioni iniziali sono contenute nel file **`motoreDC0.m`**

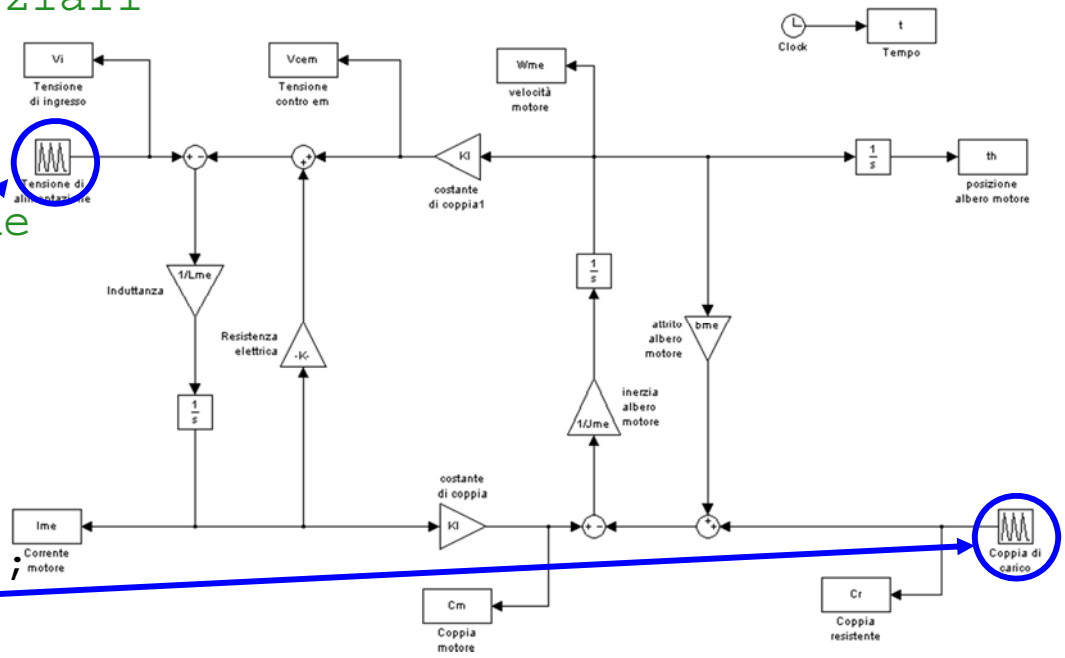
```
% dati di ingresso e condizioni iniziali
nprova=1;
switch nprova
case 1
% Rilievo caratteristica statiche
th0=0 % posizione iniziale albero
Wme0=0; % velocità iniziale motore
Ime0=Imax; % corrente iniziale
Tfin=60; % tempo della simulazione
TABTEMPI=[0 Tfin/2.2 Tfin/1.8 Tfin];
TABCR=[Cmax 0 0 Cmax];
TABVIN=[Vn Vn Vn Vn];
case 2
   ...
end
```



File dei parametri e condizioni iniziali

I segnali di ingresso sono contenuti nel file **motoreDC0.m**

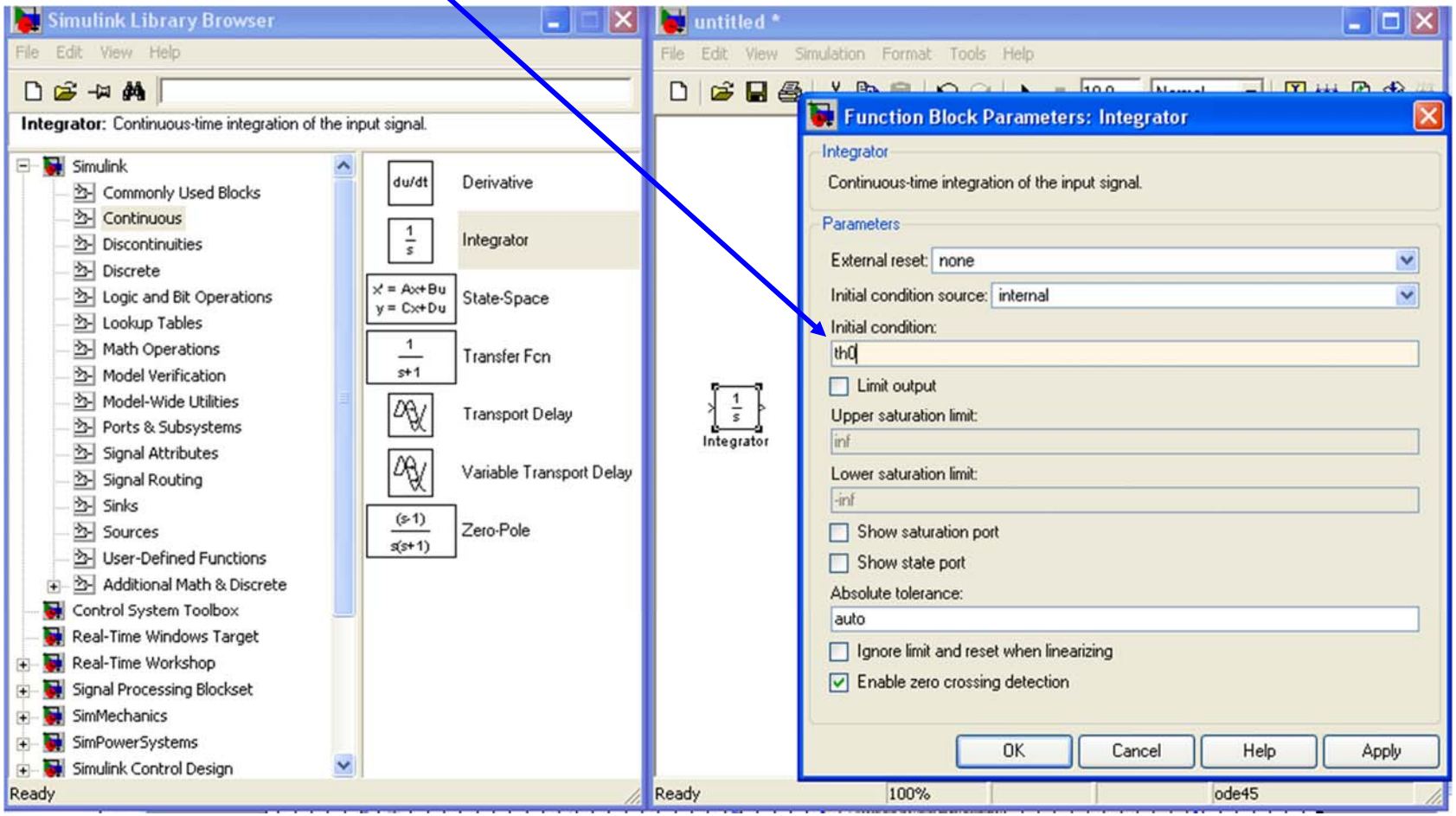
```
% dati di ingresso e condizioni iniziali
nprova=1;
switch nprova
case 1
   % Rilievo caratteristica statiche
th0=0 % posizione iniziale albero
Wme0=0; % velocità iniziale motore
Ime0=Imax; % corrente iniziale
Tfin=60; % tempo della simulazione

TABTEMPI=[0 Tfin/2.2 Tfin/1.8 Tfin];
TABCR=[Cmax 0 0 Cmax];
TABVIN=[Vn Vn Vn Vn];
case 2
   ...
end
```



File dei Parametri–Condizioni iniziali

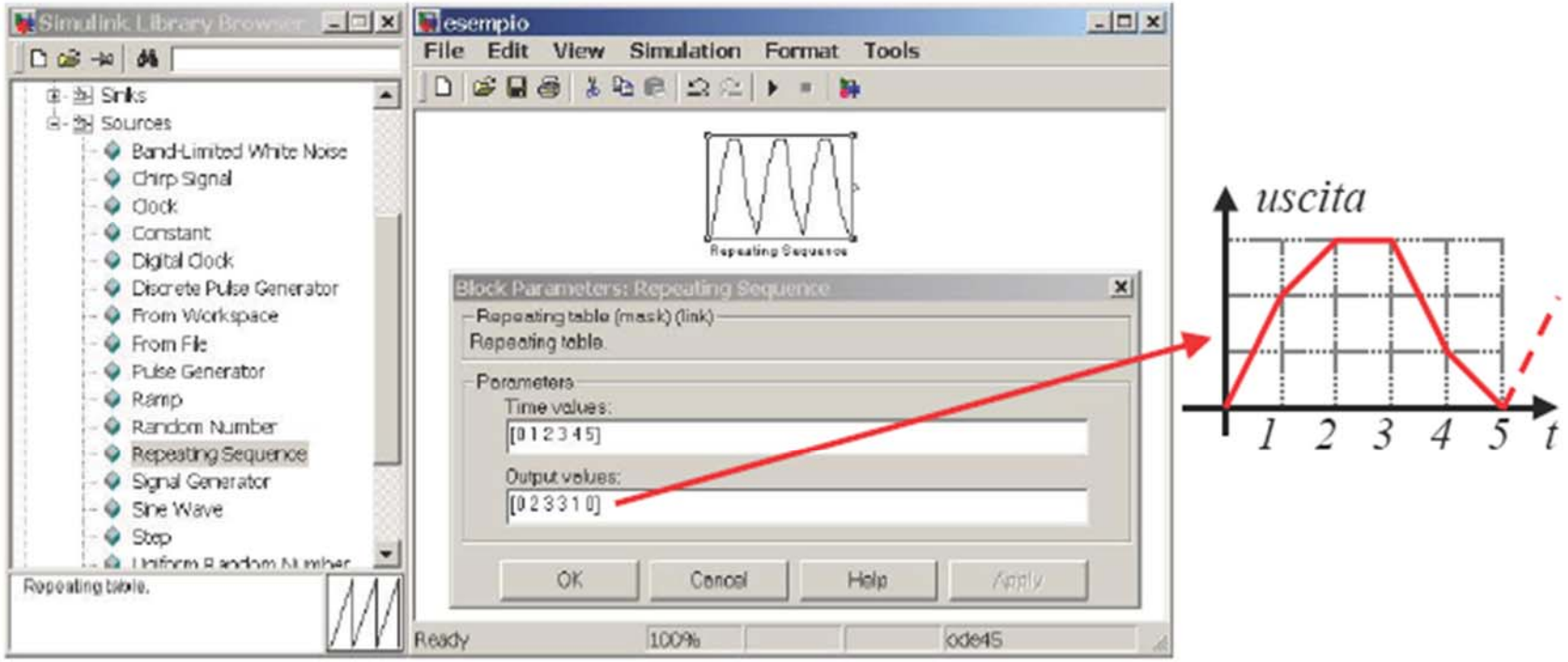
- **Ad ogni integratore del modello corrisponde una condizione iniziale**.
- Tutte le condizioni iniziali sono contenute nel file **motoreDC0.m**

```
th0=0 % posizione iniziale albero
```



Segnali di ingresso

- Il segnale di ingresso è definito sfruttando le “Repeating sequence” in cui si può dare una serie di valori della variabile di uscita in corrispondenza di determinati istanti. Le coppie istante-valore sono poi interpolate linearmente e ripetute nel tempo. È opportuno parametrizzare nel file dei parametri i vettori “Time values” e “Output values”.

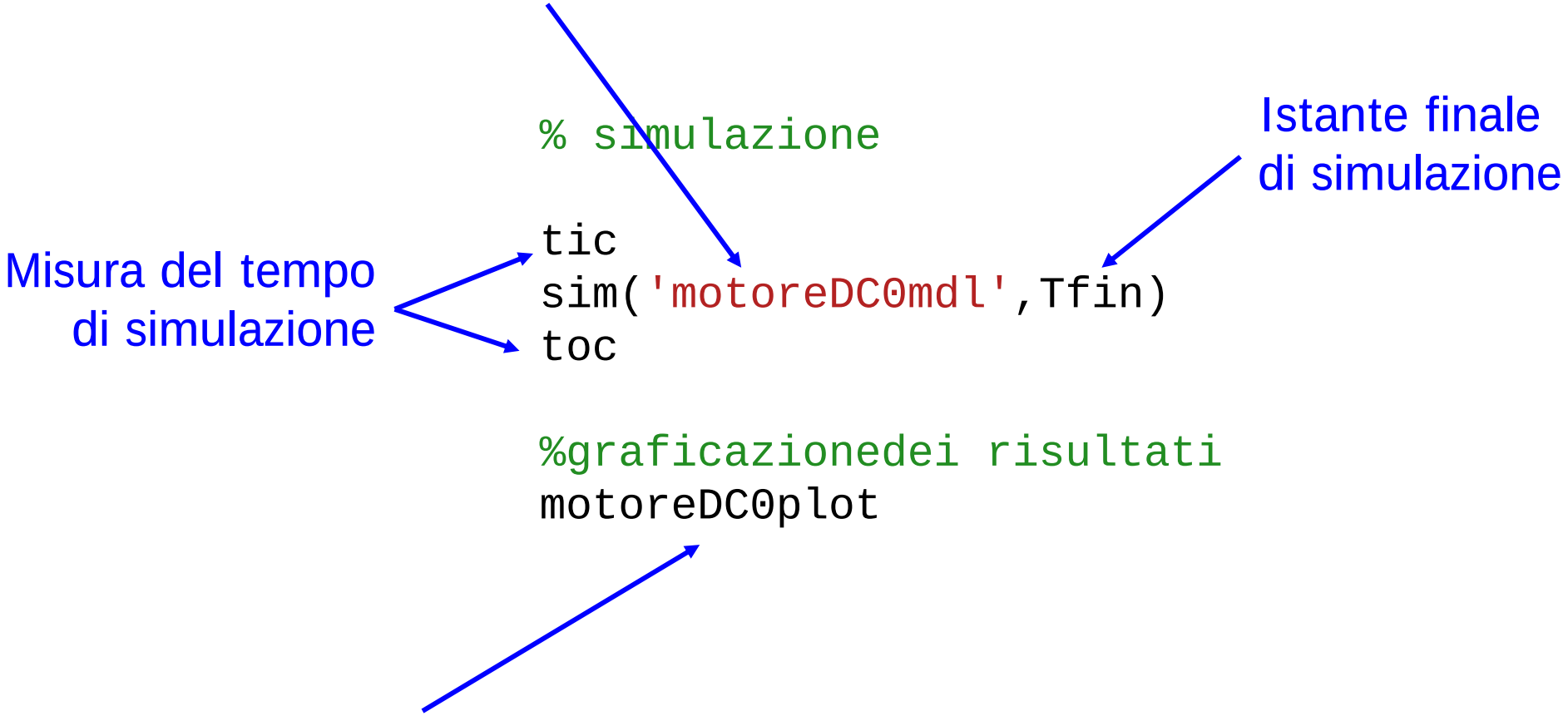


Esempio: coppia di carico al motore.

```
% dati di ingresso e condizioni iniziali
...
TABTEMPI=[0 Tfin/2.2 Tfin/1.8 Tfin];
TABCR=[Cmax 0 0 Cmax];
```

File dei Parametri–Simulazione

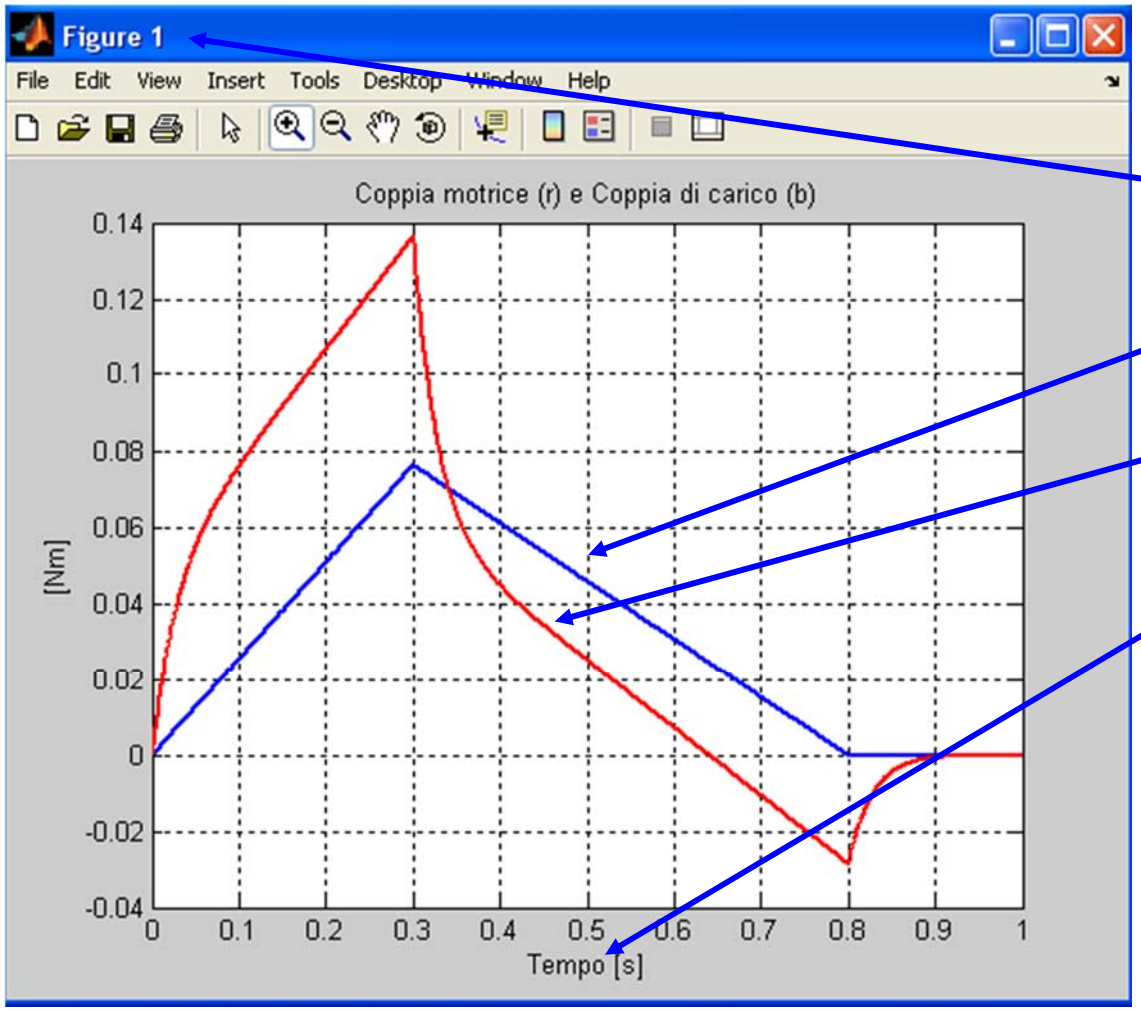
- Dopo il “caricamento” di tutti i dati necessari il file dei parametri “chiama” la simulazione del modello.



- Al termine della simulazione il file dei parametri “chiama” il file matlab **`motoreDC0plot.m`** per la graficazione dei risultati.

File di graficazione

- Tutti i segnali rilevati nella simulazione possono essere riportati in grafici. Il file **motoreDC0plot.m** contiene i comandi per la rappresentazione delle figure di interesse.



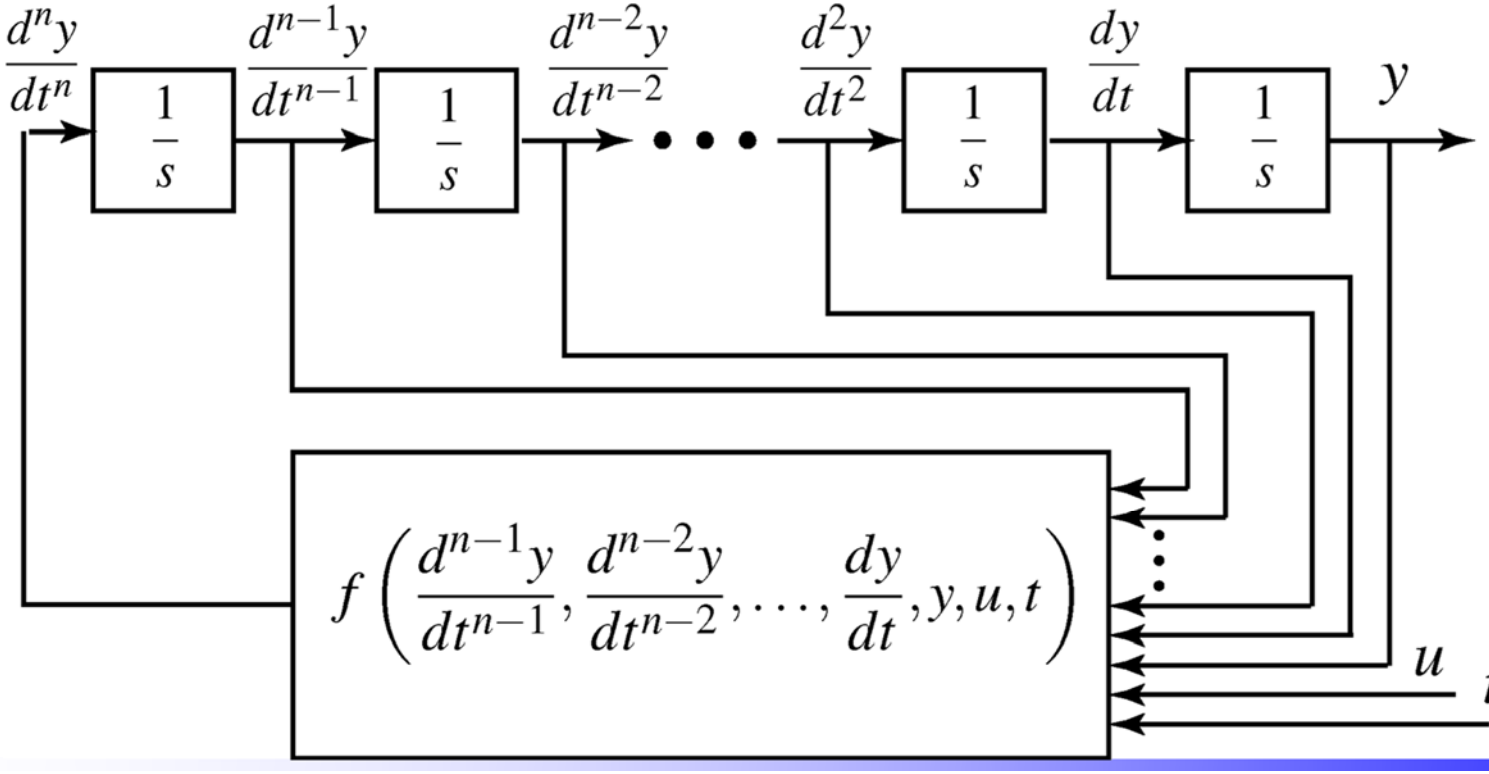

```
figure(1)
clf
plot (t,Cr,'b')
hold on
plot (t,Cm,'r')
title('Coppia motrice ...')
xlabel('Tempo [s]')
ylabel('[Nm]')
grid on
```

Simulazione di un'equazione differenziale

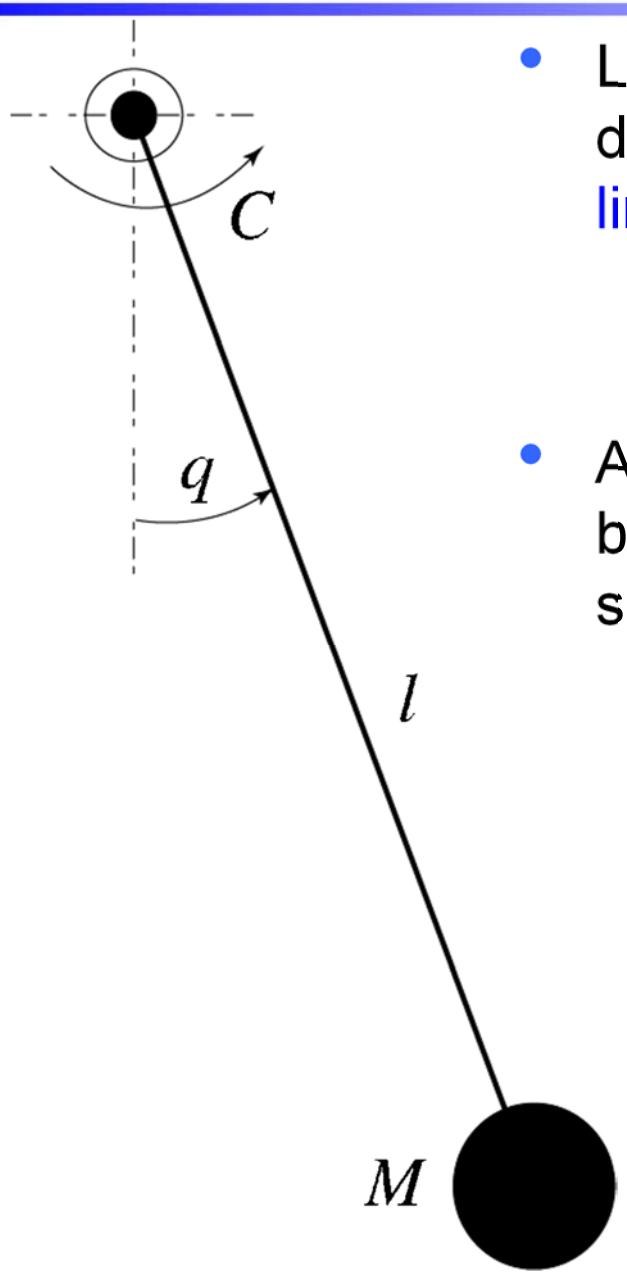
- È possibile simulare mediante uno schema simulink qualunque equazione differenziale, anche non lineare, anche con elementi tempo-varianti, espressa in forma esplicita

$$\frac{d^n y(t)}{dt^n} = f\left(\frac{d^{n-1}y}{dt^{n-1}}, \frac{d^{n-2}y}{dt^{n-2}}, \ldots, \frac{dy}{dt}, y, u, t\right)$$

- Il corrispondente schema a blocchi (simulink) risulta



Simulazione di un'equazione differenziale: il pendolo semplice



- La dinamica del sistema può essere descritta dalle seguente equazione differenziale non lineare

$$M l^2 \ddot{q} + M g l \sin(q) = C$$

- Al fine di rappresentare mediante schemi a blocchi l'equazione differenziale e poterne poi simulare l'uscita conviene riscriverla come

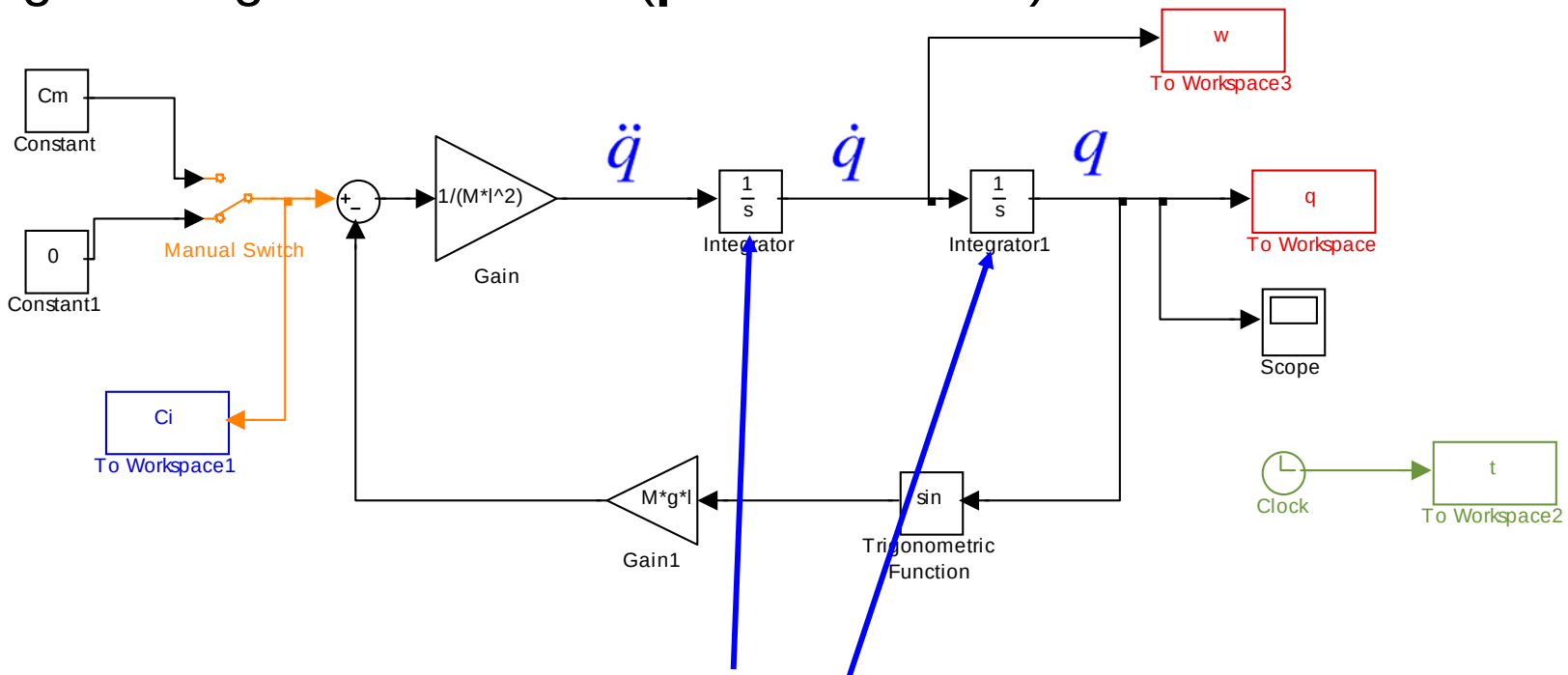
$$\ddot{q} = \frac{1}{Ml^2}\left(C - M g l \sin(q)\right)$$

Pendolo semplice – schema simulink

- Dall'implementazione (e integrazione) dell'equazione

$$\ddot{q} = \frac{1}{Ml^2}\left(C - Mgl\sin(q)\right)$$

segue il seguente schema (**pendolo.mdl**)



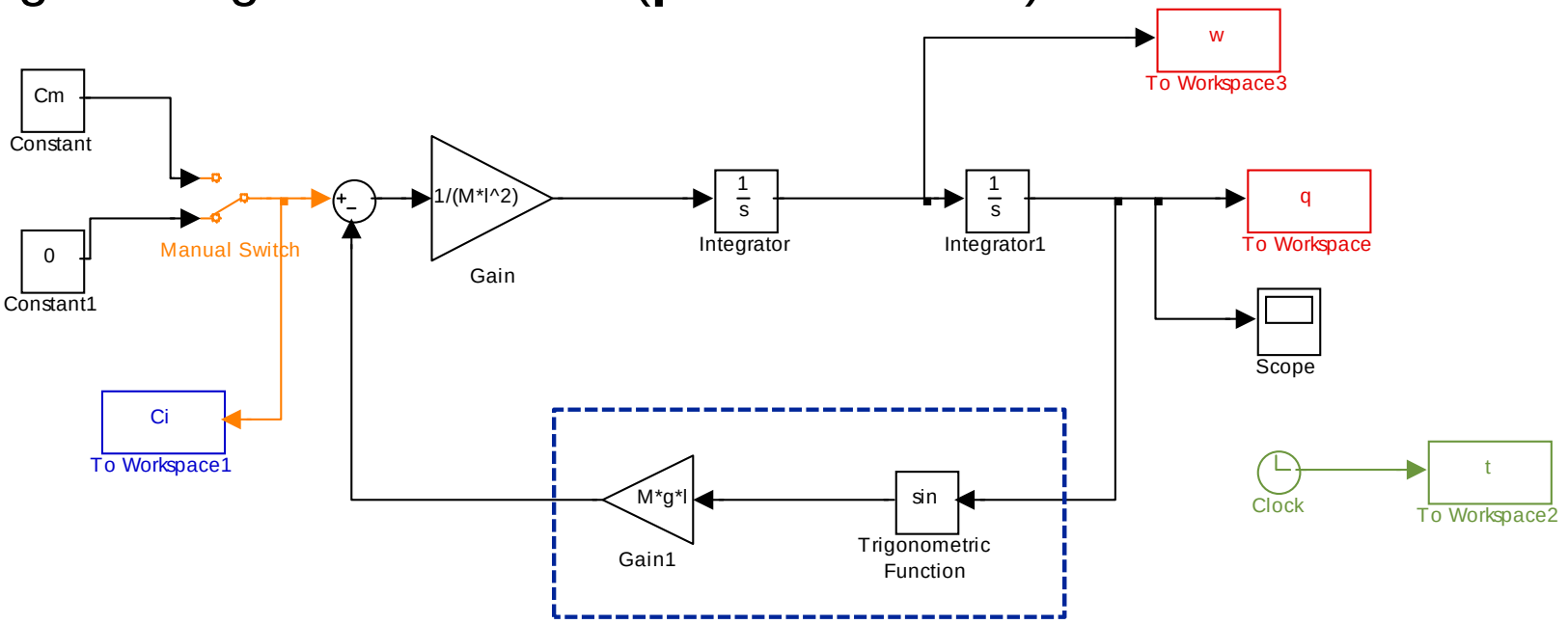
Parametri e condizioni iniziali su $\dot{q}$ e q definiti in **PendParams.m**

Pendolo semplice – schema simulink

- Dall'implementazione (e integrazione) dell'equazione

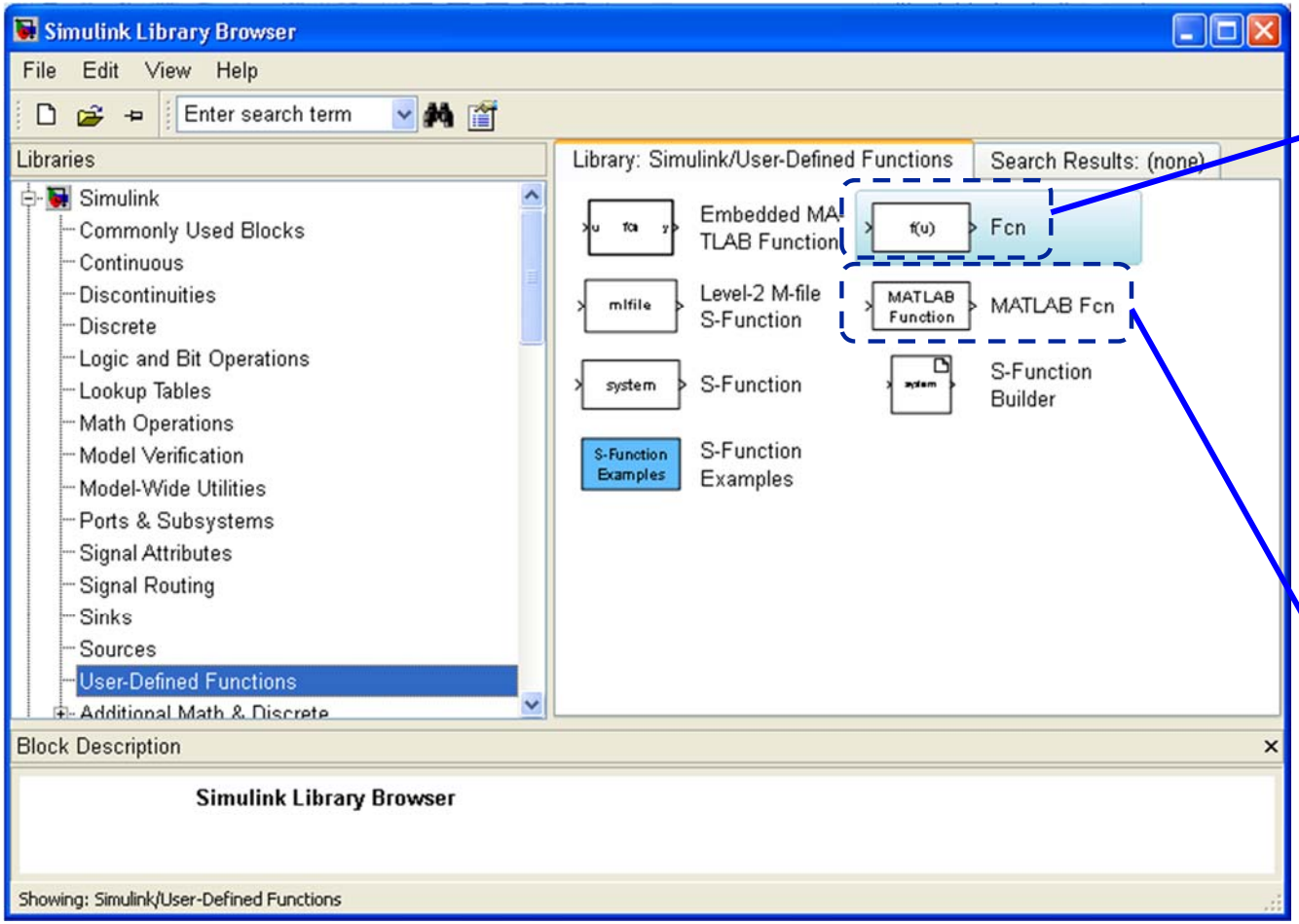
$$\ddot{q} = \frac{1}{Ml^2}\left(C - Mgl\sin(q)\right)$$

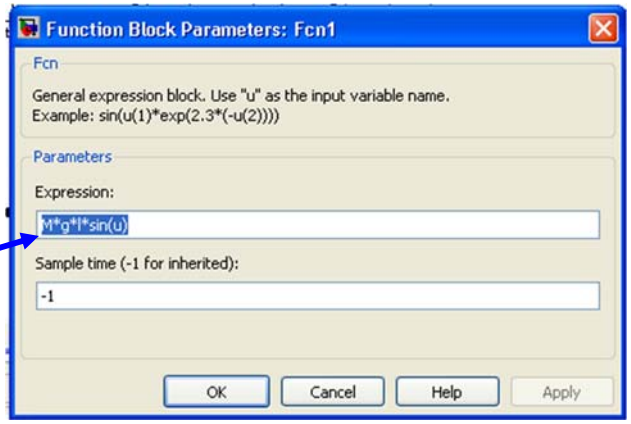
segue il seguente schema (**pendolo.mdl**)



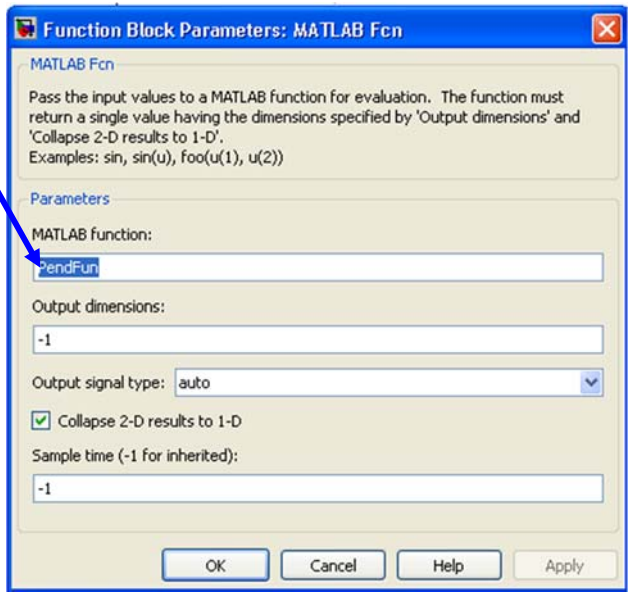
Per definire la funzione $Mgl\sin(q)$ è possibile utilizzare altri blocchi simulink, in particolare quelli che consentono di inserire funzioni definite dall'utente

Blocchi per l'inserimento di funzioni definite dall'utente





Funzione definita direttamente nel blocco



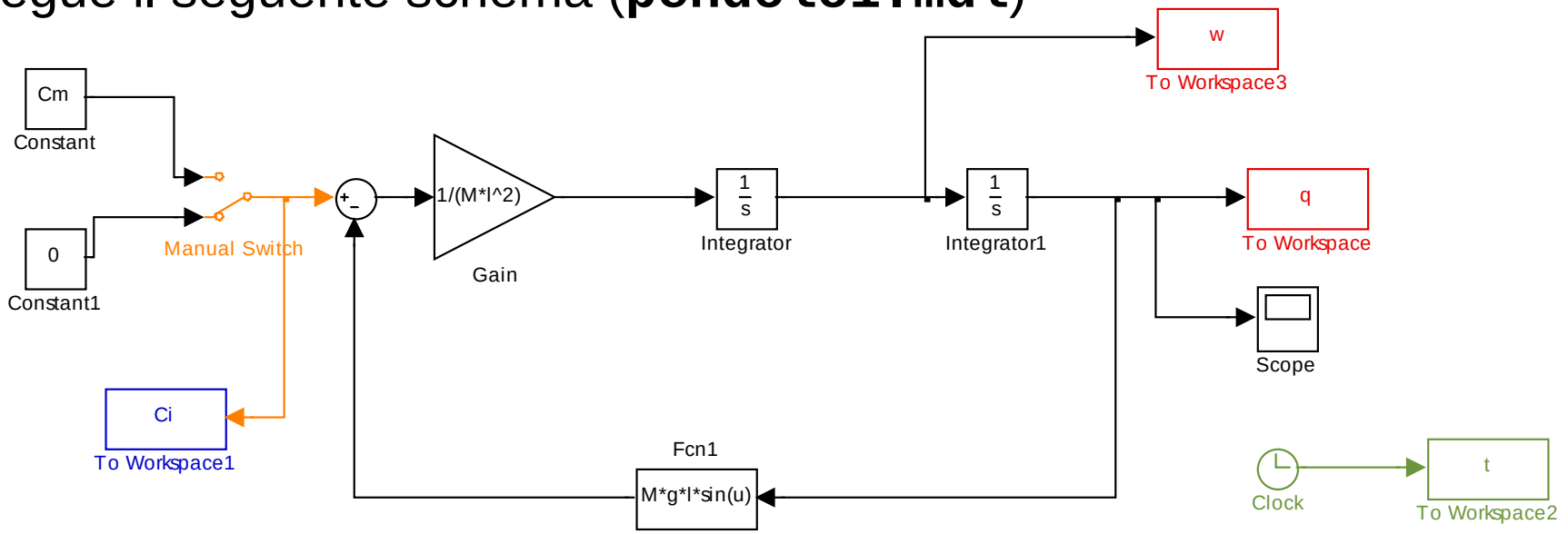
Funzione definita tramite m-function

Pendolo semplice – schema simulink

- Dall'implementazione (e integrazione) dell'equazione

$$\ddot{q} = \frac{1}{Ml^2}\left(C - Mgl\sin(q)\right)$$

segue il seguente schema (**pendolo1.mdl**)

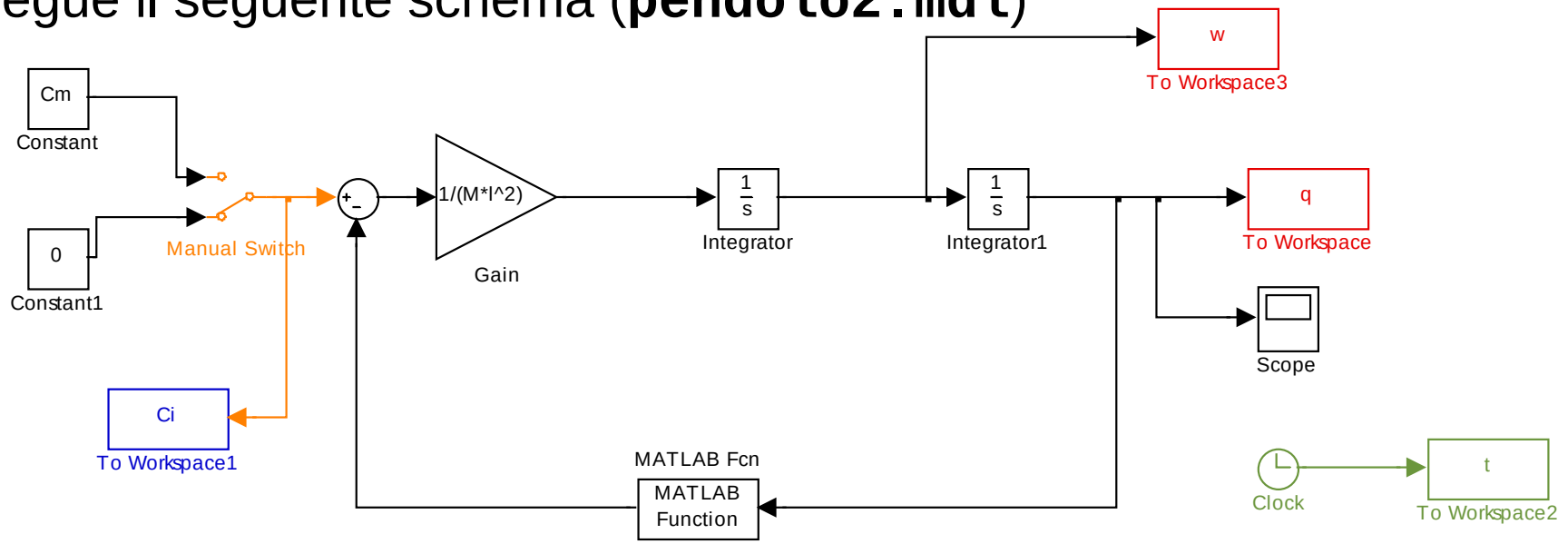


Pendolo semplice – schema simulink

- Dall'implementazione (e integrazione) dell'equazione

$$\ddot{q} = \frac{1}{Ml^2}\left(C - Mgl\sin(q)\right)$$

segue il seguente schema (**pendolo2.mdl**)



Listato della m-function

```
function [Out] = PendFun(q)
      global M g l;
    Out = M*g*l*sin(q);
```

Parametri dichiarati global per averne visibilità all'interno (lo stesso è fatto nell'm-file in cui sono definiti: PendParams.m)

Simulazione di un'equazione differenziale: esempio

- Data l'equazione differenziale

$$3\dddot{y}(t) + 2\ddot{y}(t) + 4\dot{y}(t) + y(t) = 7u(t)$$

simulare mediante uno schema simulink:

1. la risposta forzata con un ingresso a gradino
2. la risposta libera con condizioni iniziali

$$\ddot{y}(t) = 0, \quad \dot{y}(t) = 3, \quad y(t) = -2$$

3. la risposta completa del sistema

CONTROLLI AUTOMATICI
Ingegneria Meccanica e Ingegneria del Veicolo

http://www.dii.unimore.it/~lbiagiotti/ControlliAutomatici.html

INTRODUZIONE A SIMULINK

Ing. Luigi Biagiotti
e-mail: luigi.biagiotti@unimore.it
http://www.dii.unimore.it/~lbiagiotti