

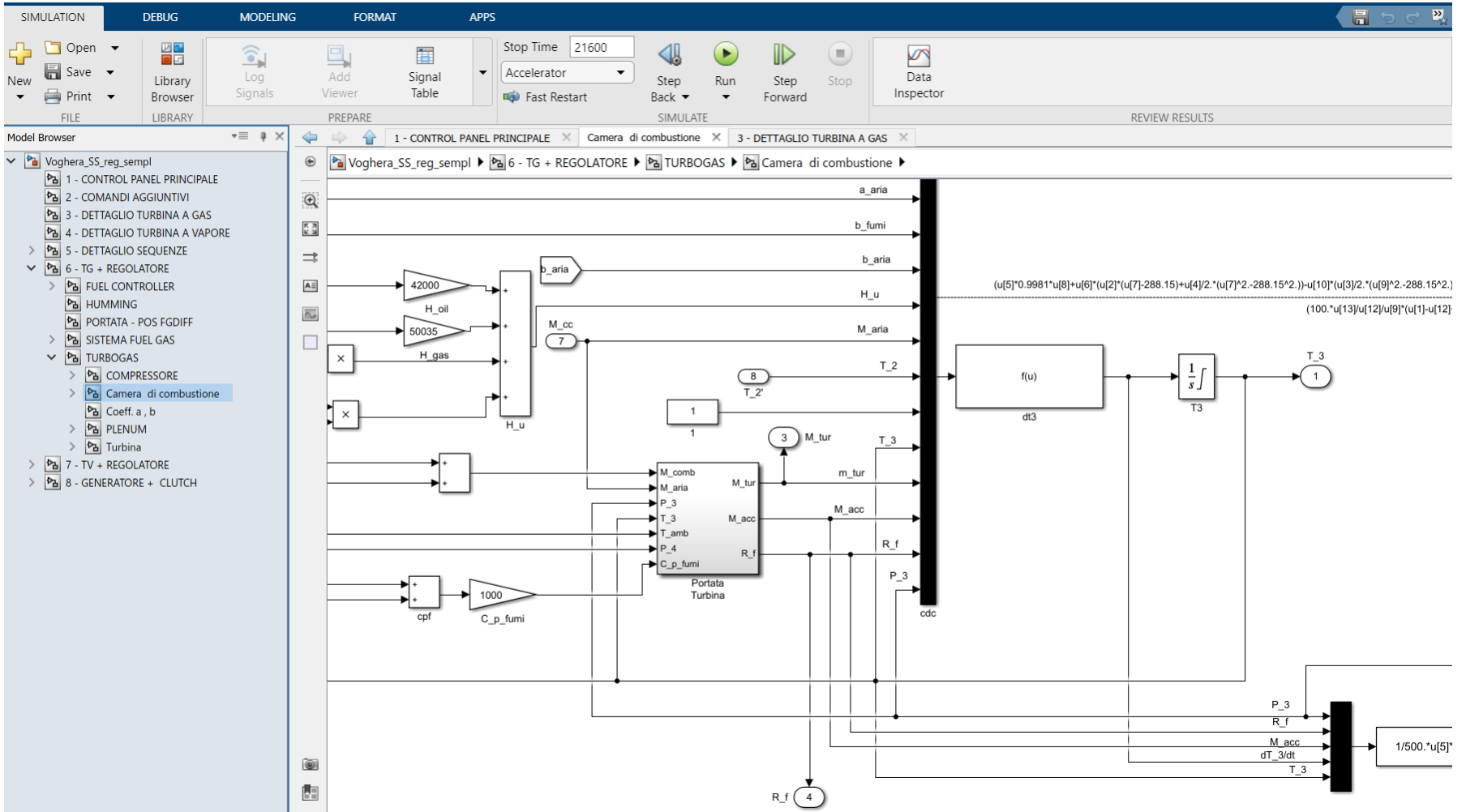
# **Simulazione dei Sistemi dinamici con Matlab-Simulink**

## **Modellazione Simulink – parte prima**

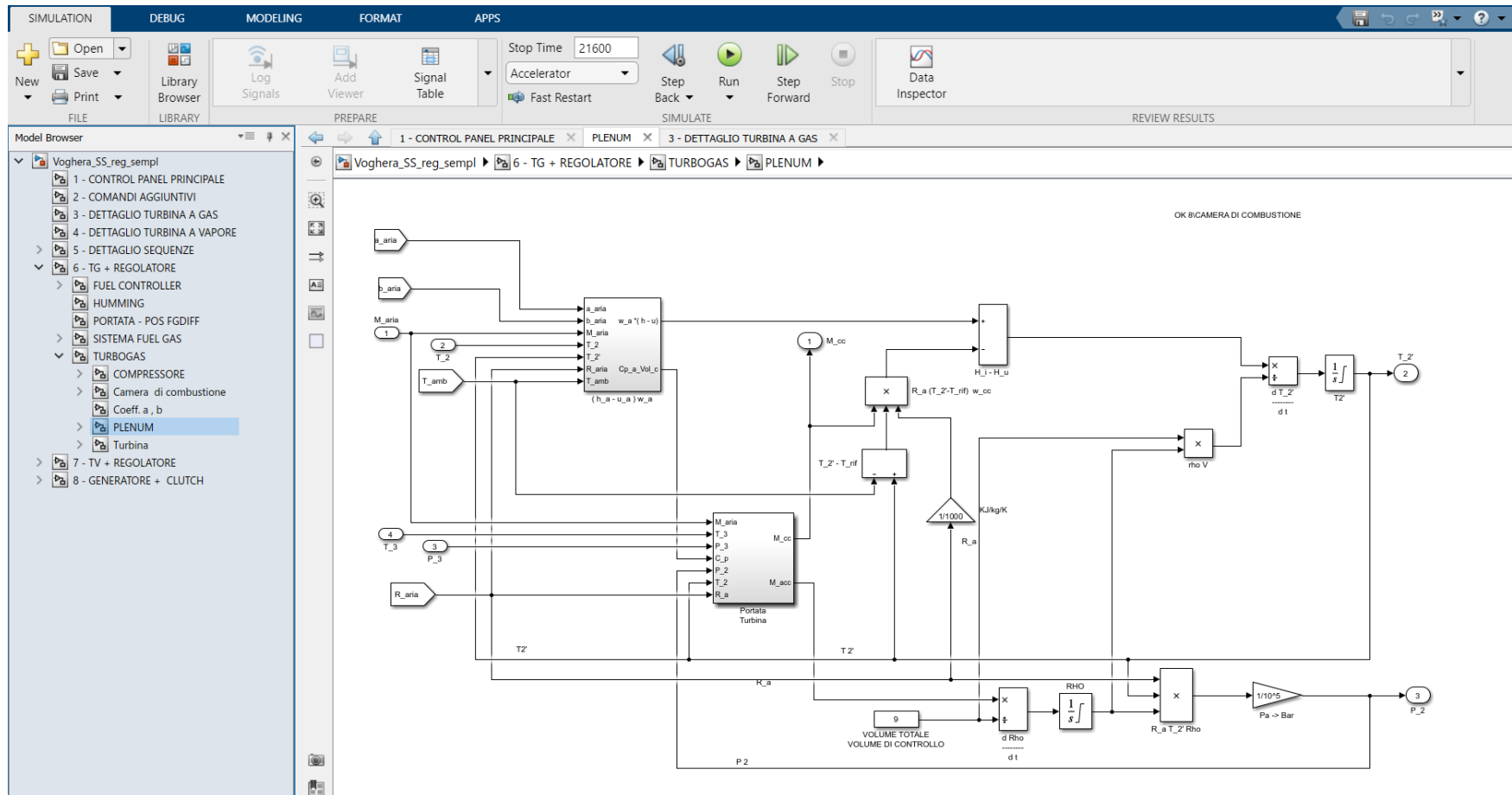
**Ing. Alessandro Pisano**  
**apisano@unica.it**

# Simulink consente la realizzazione di modelli per la simulazione dinamica per via **grafica**

Voghera\_SS\_reg\_sempl/.../TURBOGAS/Camera di combustione - Simulink academic use



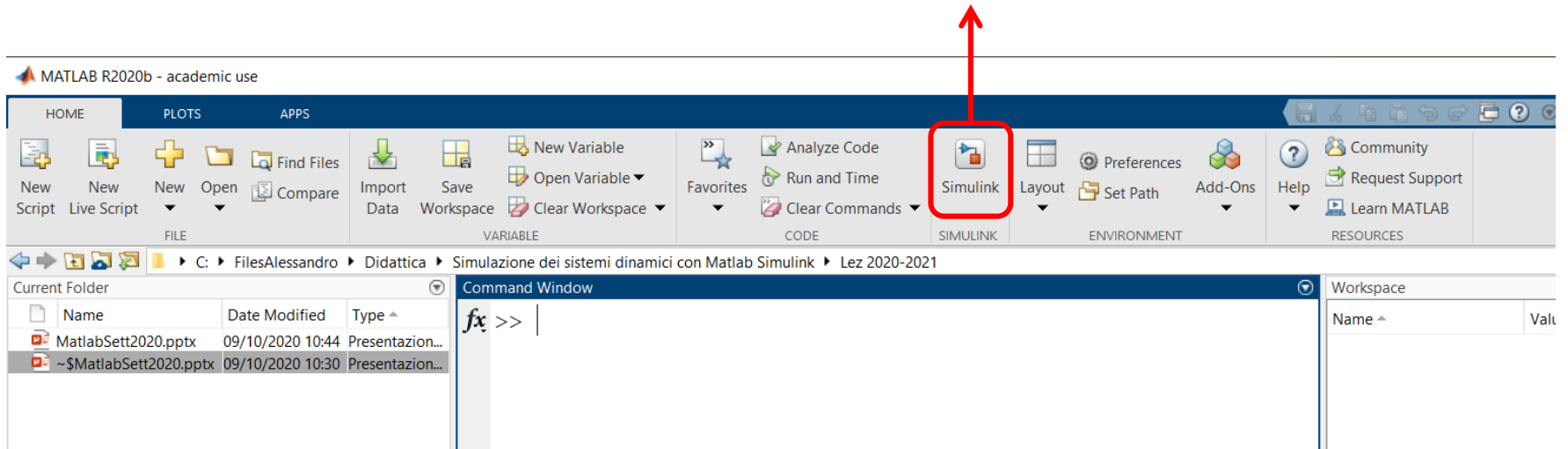
# Simulink consente la realizzazione di modelli per la simulazione dinamica per via **grafica**



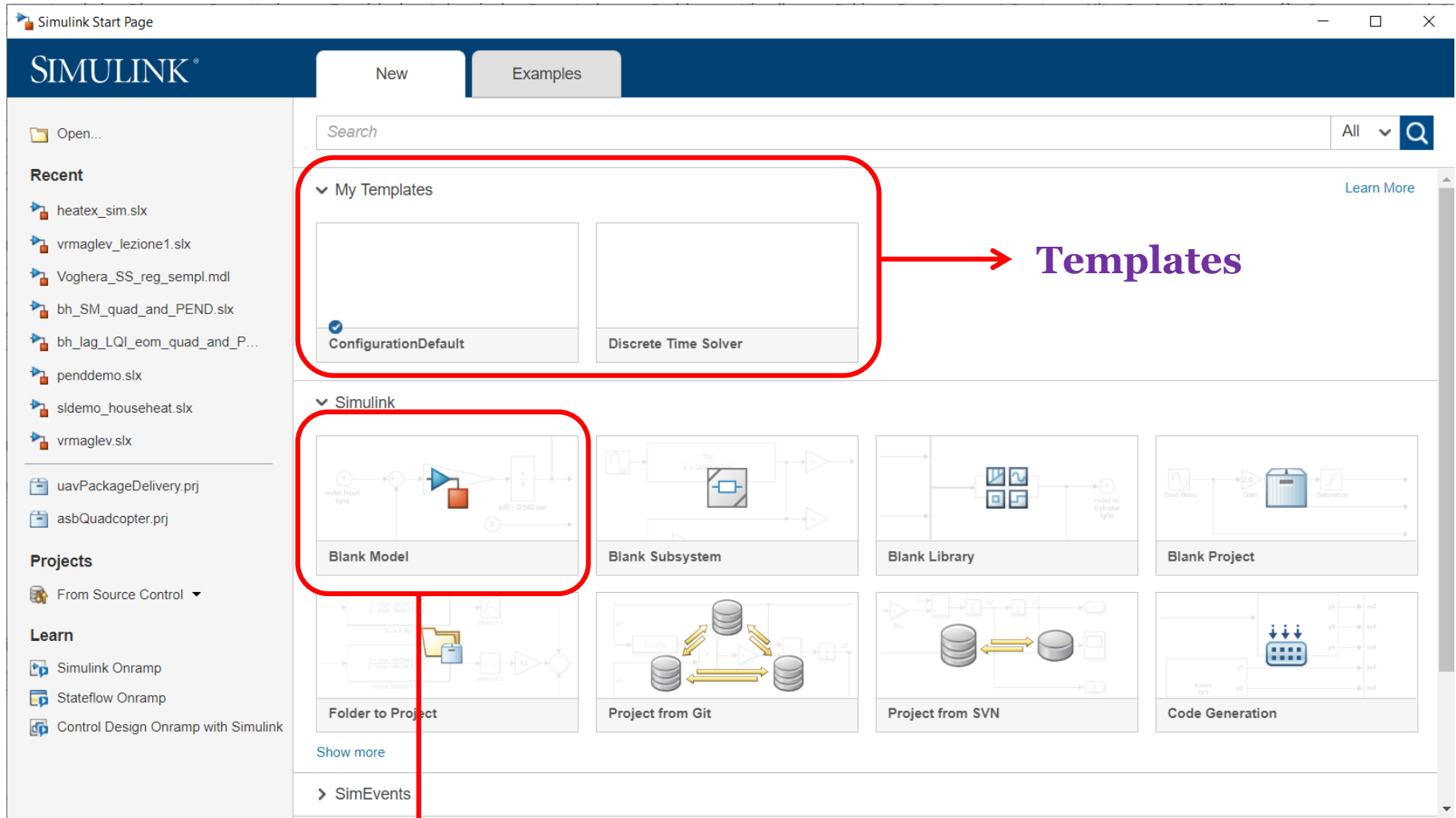
Ciascuna linea di connessione corrisponde ad un **segnale**. I vari blocchi applicano ai segnali in ingresso determinate operazioni matematiche, e producono in uscita segnali risultanti da tali elaborazioni.

# Finestra di avvio di Matlab (R2020b)

## Avvio SIMULINK



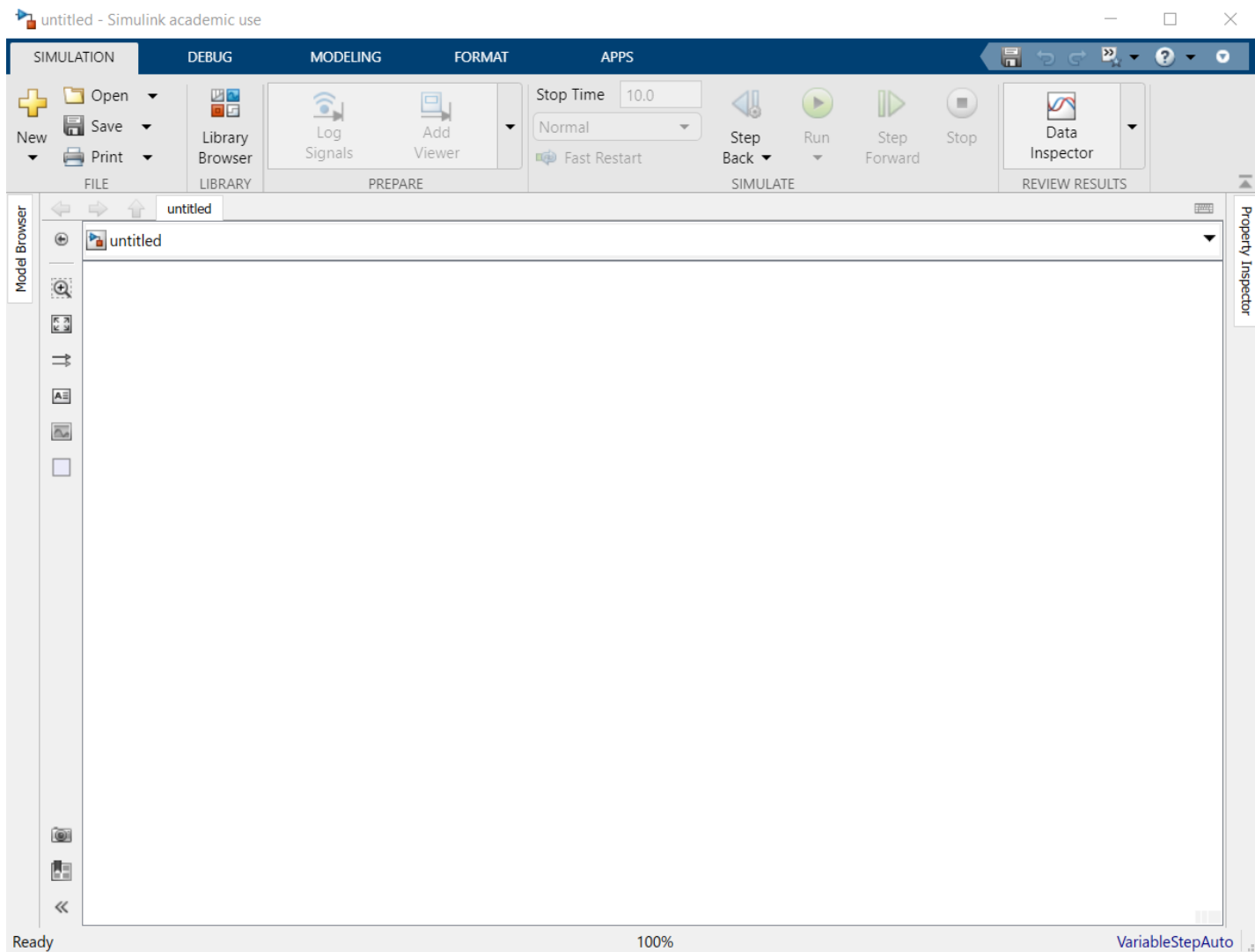
# Finestra di avvio SIMULINK



**Templates**

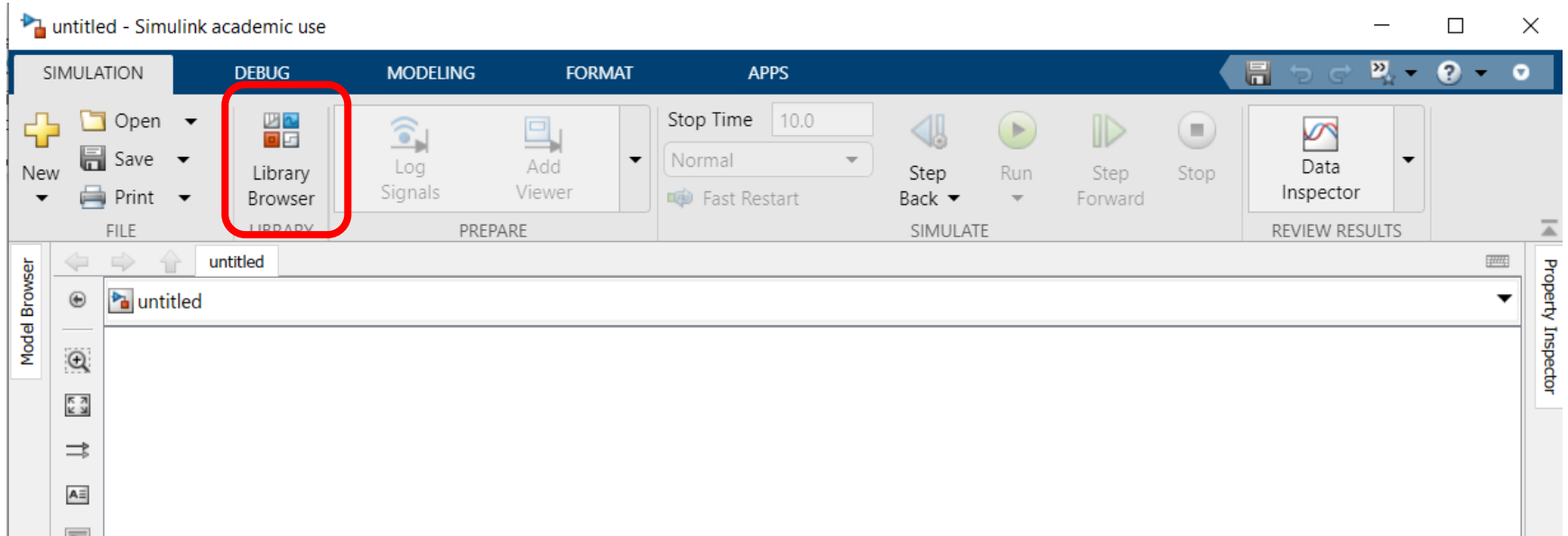
**Aprire un modello Simulink vuoto  
avente le proprietà di default**

## Modello Simulink in bianco

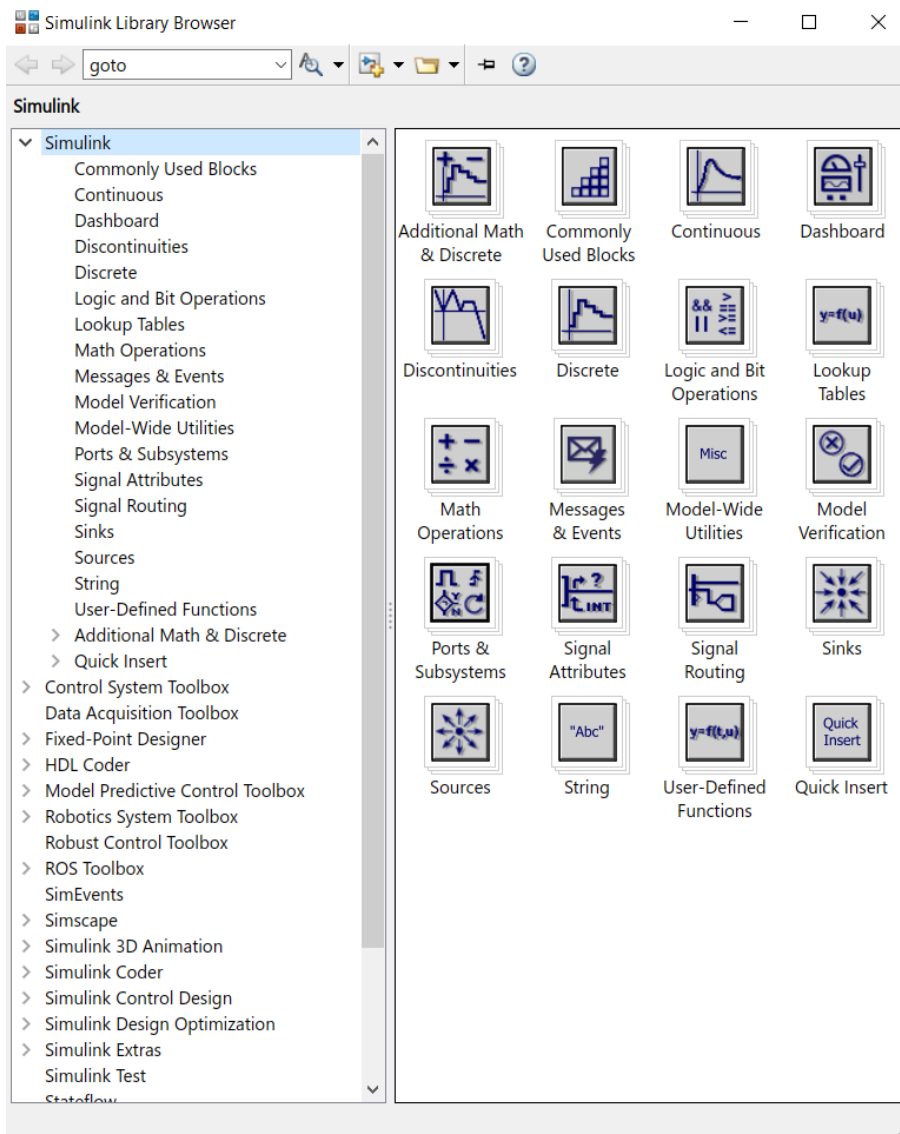


La realizzazione di modelli di simulazione dinamica avviene per via grafica **assemblando fra loro all'interno della pagina di lavoro un certo numero di blocchi Simulink**, in modo da implementare le funzionalità desiderate.

I blocchi Simulink sono allocati all'interno di librerie. E' possibile accedere al «Library browser» cliccando il relativo pulsante nella finestra che ospita il modello in bianco (Menu: «SIMULATION»)



# Simulink Library Browser



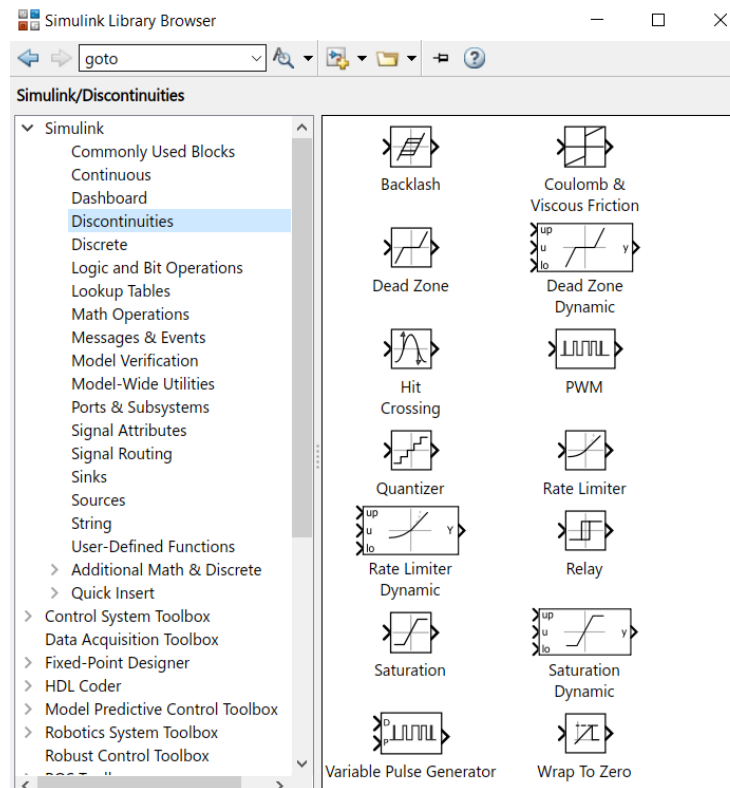
**Contiene le librerie di blocchi elementari SIMULINK, da quelli di base fino a quelli più sofisticati orientati a particolari applicazioni**



Indaghiamo rapidamente il contenuto di alcune librerie di base che contengono i blocchi Simulink che useremo in questo corso.

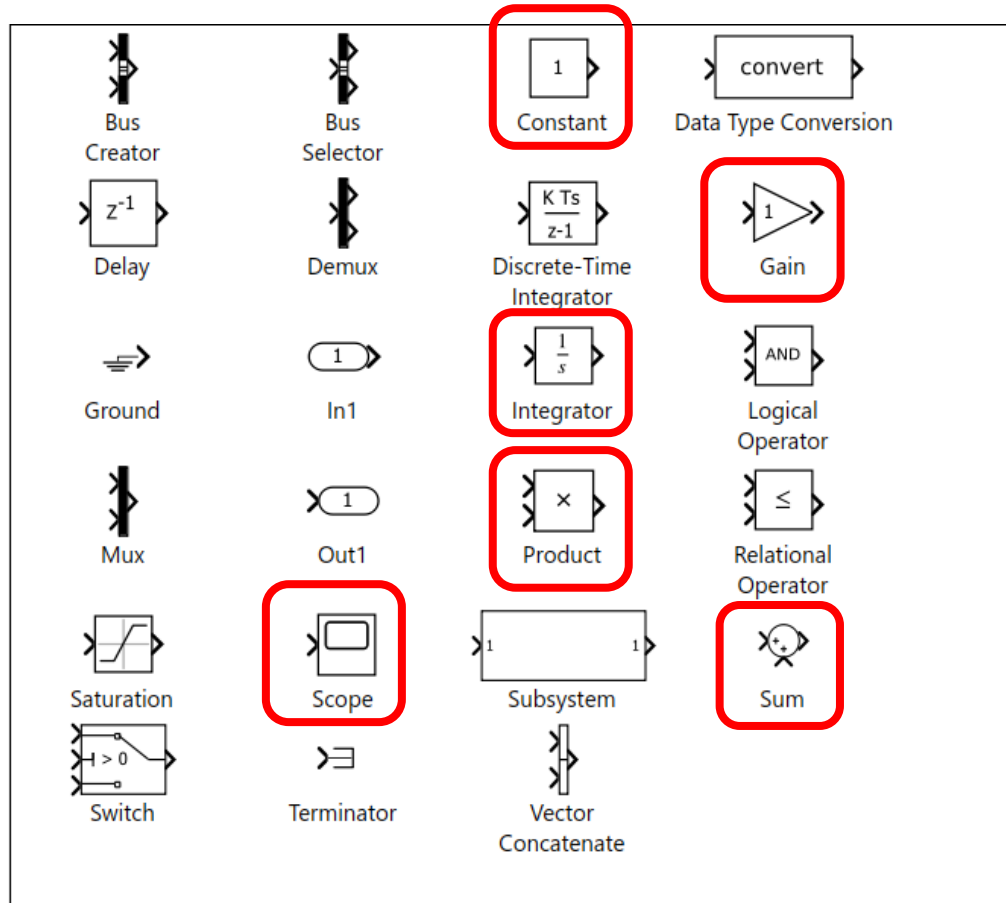
Per accedere al contenuto di una libreria si faccia click sul nome della stessa nella parte sinistra della finestra **Simulink Library Browser**

Contestualmente sarà visualizzato il contenuto della libreria nella parte destra della medesima finestra.



La prima libreria della lista è la Libreria “**Commonly used blocks**”

## COMMONLY USED BLOCKS



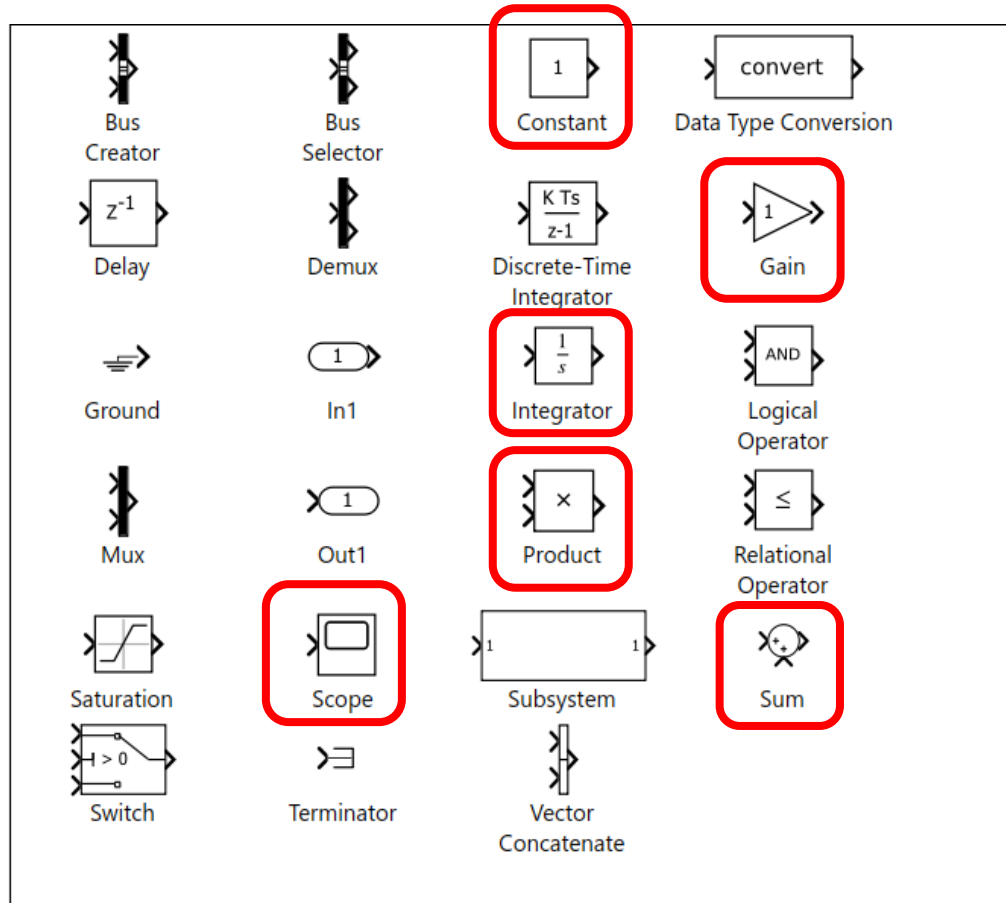
E' una libreria che contiene un insieme di blocchi che implementano funzionalità variegate e che vengono impiegati particolarmente di frequente nei modelli di simulazione.

Nello screenshot a sinistra, che ne mostra il contenuto, sono evidenziati alcuni blocchi di particolare importanza.

Il blocco **Constant** genera al suo terminale di uscita un segnale costante nel tempo.

Il blocco **Gain** moltiplica il segnale connesso al terminale di ingresso per un guadagno costante, e restituisce il risultato nel terminale di uscita.

## COMMONLY USED BLOCKS



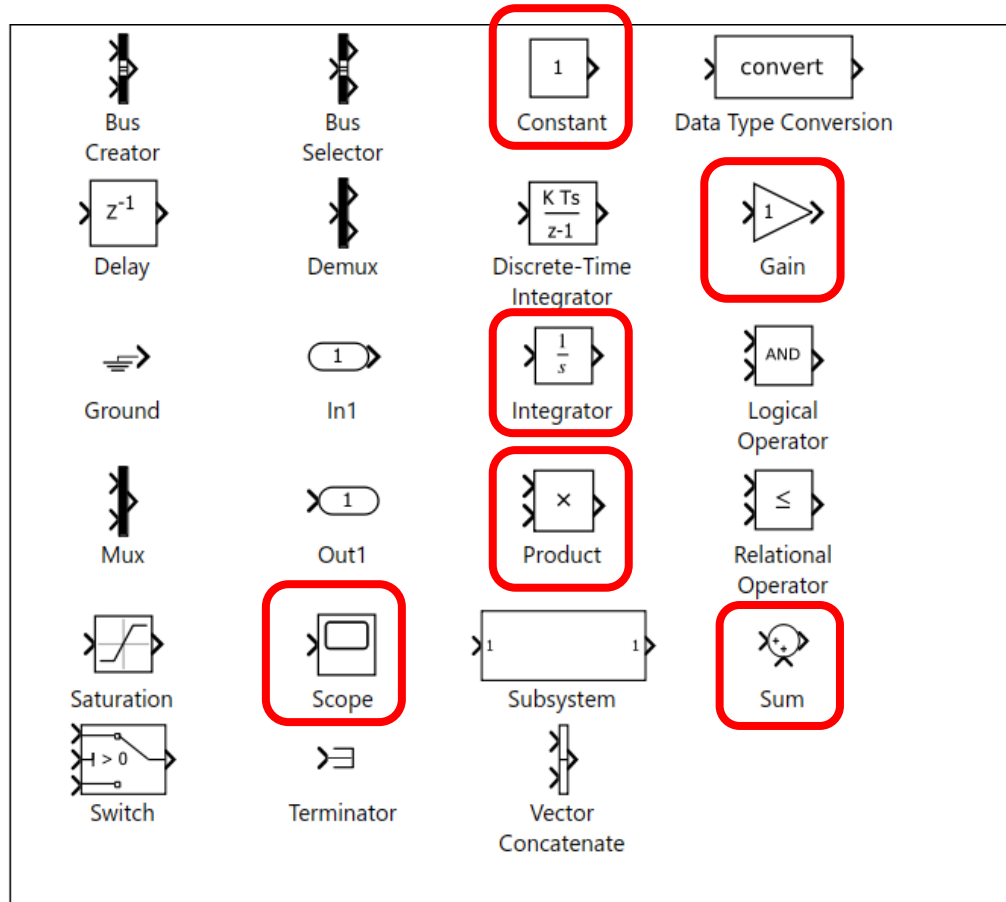
I terminali di ingresso e di uscita compaiono tipicamente sul lato sinistro e sul lato destro del blocco. In dipendenza dalle funzioni, alcuni blocchi avranno solo un terminale di uscita, altri solo un terminale di ingresso, altri ancora avranno sia terminali di ingresso che terminali di uscita.

Il blocco **Integrator** integra il segnale connesso al terminale di ingresso e restituisce il risultato nel terminale di uscita.

Il blocco **Product** moltiplica fra loro i segnali connessi ai due terminali di ingresso, restituisce il risultato nel terminale di uscita.

La prima libreria della lista è la Libreria “**Commonly used blocks**”

## COMMONLY USED BLOCKS



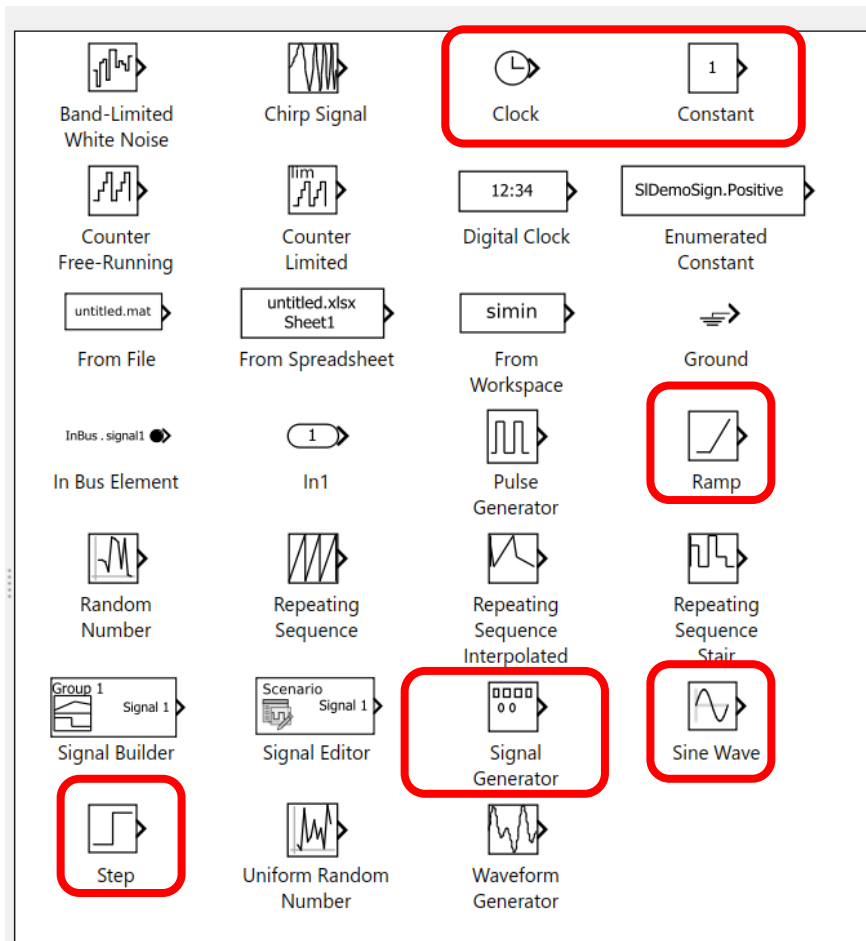
Il blocco **Scope** visualizza in una finestra grafica l'evoluzione temporale del segnale collegato al terminale di ingresso. Come tale, il blocco Scope ha unicamente un terminale di ingresso.

Il blocco **Sum**, infine, **somma** fra loro i segnali connessi ai due terminali di ingresso, restituisce il risultato nel terminale di uscita.

Tutti i blocchi presenti nella libreria “**Commonly used blocks**” sono anche inseriti all'interno di altre librerie contenenti blocchi dalle funzionalità in qualche modo «simili».

I blocchi Simulink si importano nella pagina di lavoro direttamente dalla libreria che li ospita mediante **drag-and-drop**.

## SOURCES



Realizziamo alcune simulazioni preliminari in cui si generano e visualizzano alcuni segnali

I blocchi per la generazione di segnali si trovano nella Libreria «*Sources*»

Sono evidenziati il blocco **Clock** che rende accessibile la variabile temporale, il blocco **Constant** già descritto, i blocchi **Ramp**, **Sine Wave** e **Step**, che generano le tre corrispondenti forme d'onda a rampa, sinusoidale, o a gradino, ed il blocco **Signal Generator** che consente di scegliere fra 4 diverse tipologie mediante un menu presente nella **finestra di configurazione del blocco**.

Ogni blocco prevede un set di uno o più parametri di configurazione (ad esempio, per il blocco «Gain» il valore del guadagno). I parametri di configurazione di un blocco sono settati all'interno di una **finestra di configurazione** alla quale si accede facendo **doppio click sul blocco**.

Come test preliminare, generiamo e visualizziamo un segnale costante.

## **Realizzazione di un modello Simulink**

### **3 fasi**

1. Importare nella pagina di lavoro i blocchi elementari Simulink necessari trascinandoli con il mouse dalla rispettiva libreria (drag-and-drop)
2. Parametrizzare i blocchi Simulink nelle rispettive finestre di parametrizzazione, alle quali si accede facendo doppio click con il mouse sopra il blocco stesso.
3. Collegare tra loro i blocchi Simulink tracciando le opportune linee di interconnessione in modo da realizzare le funzionalità desiderate

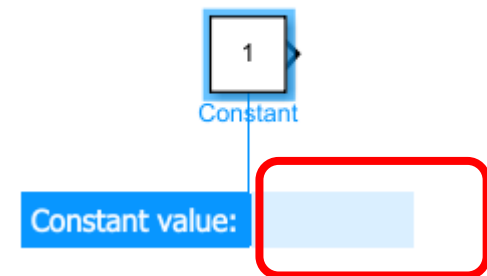
## **Esempio introduttivo:** costruzione e visualizzazione di un segnale costante

Sono sufficienti due blocchi elementari: un blocco che generi il segnale desiderato, ed un blocco che ne permetta la visualizzazione.

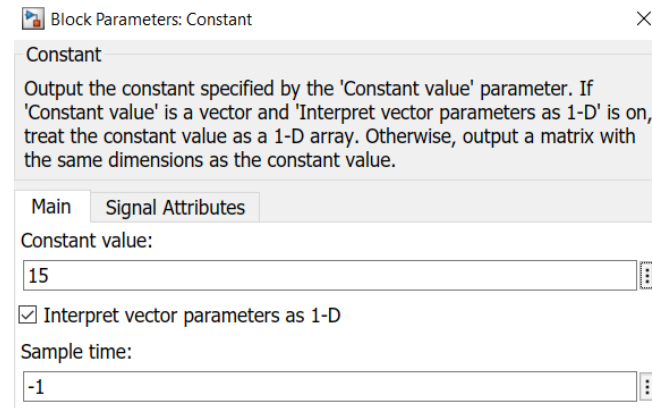
Entrambi i blocchi necessari sono reperibili nella libreria dei *Commonly Used Blocks*. Il primo blocco lo troveremo anche nella libreria “*Sources*” (blocco *Constant*), Il secondo blocco (blocco *Scope*), si trova anche nella libreria “*Sinks*”.

I blocchi necessari vanno importati nella pagina di lavoro *Untitled* trascinando con il mouse (*drag-and-drop*) l'icona del blocco all'interno della pagina di lavoro. Importiamo il blocco constant.

Quando un blocco viene importato nella finestra di lavoro si apre automaticamente una finestra di dialogo che consente di impostare i principali parametri di funzionamento. Si inserisca nella casella il valore desiderato per l'ampiezza del segnale costante (ad es. 15) e si preme invio. Il blocco cambia aspetto, e riporta al suo interno il valore del guadagno.



Fare doppio click sul blocco Constant, e nella finestra di configurazione dei parametri cambiare il valore del Sample time da `inf` a `-1`.

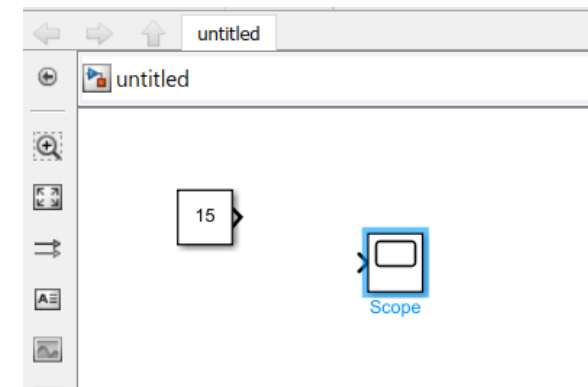


Il blocco Scope ha come parametro di configurazione il numero di porte di ingresso. Il valore di default è 1, che va bene per il test quindi la casella può essere ignorata.



Number of input ports:

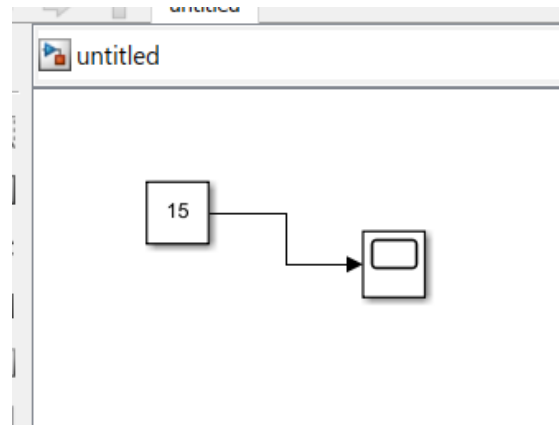
Si deve ora collegare l'uscita del generatore di funzione "Constant" con l'ingresso del blocco di visualizzazione "Scope".





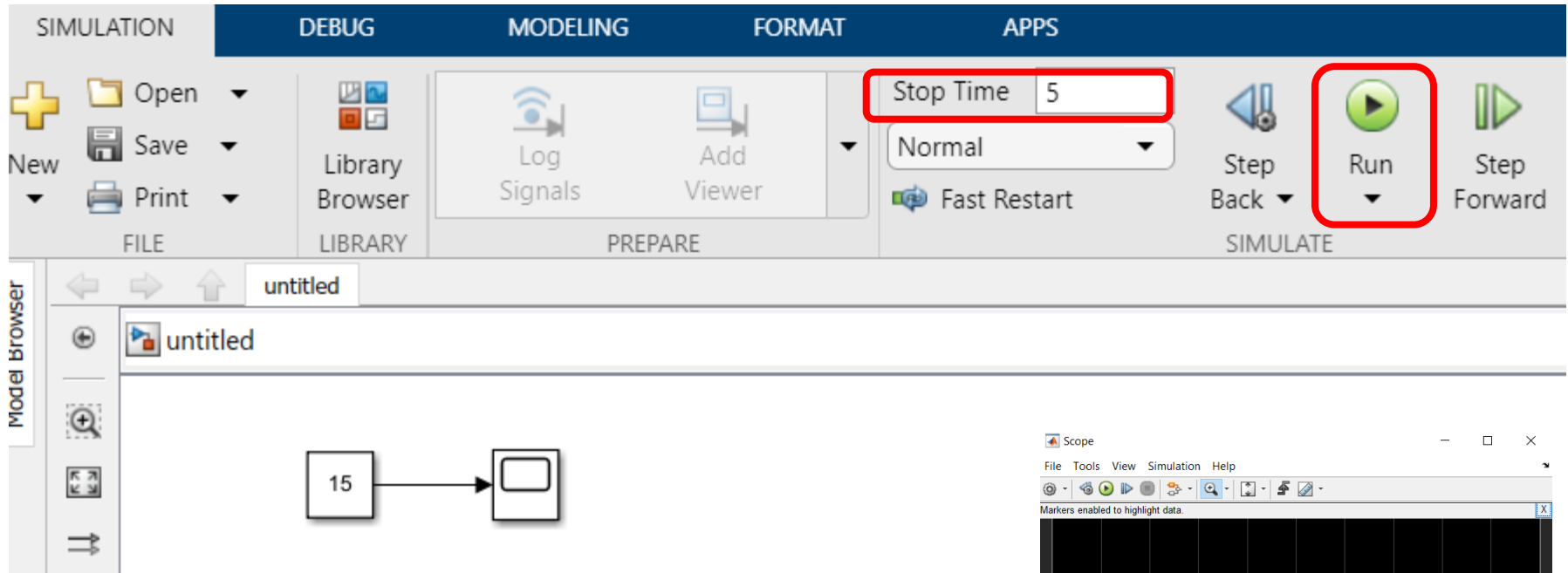
Per effettuare un collegamento tra due blocchi vi è una procedura rapida. Si deve selezionare il blocco di origine (cliccandovi sopra), e si deve successivamente selezionare il blocco di destinazione con il tasto **ctrl** premuto.

Un collegamento correttamente eseguito viene indicato come in Figura

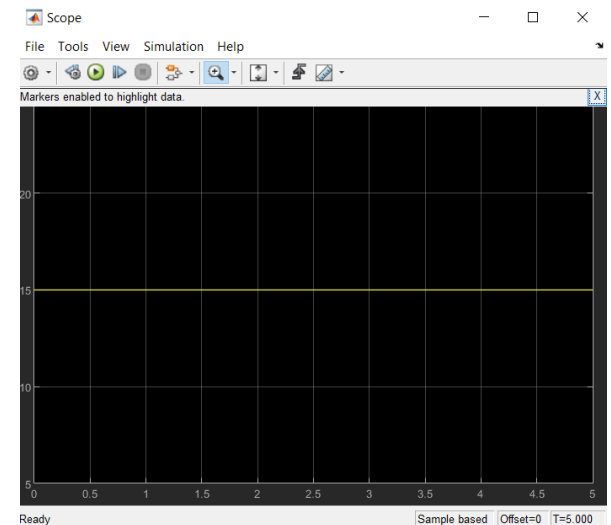


In alternativa, si può portare la freccia del mouse nel punto di inizio della linea di collegamento e quindi “tracciarla” tenendo premuto il tasto sinistro del mouse, fino al punto di destinazione. Le linee di collegamento, così come i blocchi, possono essere rimossi dallo schema con il tasto `cancel`.

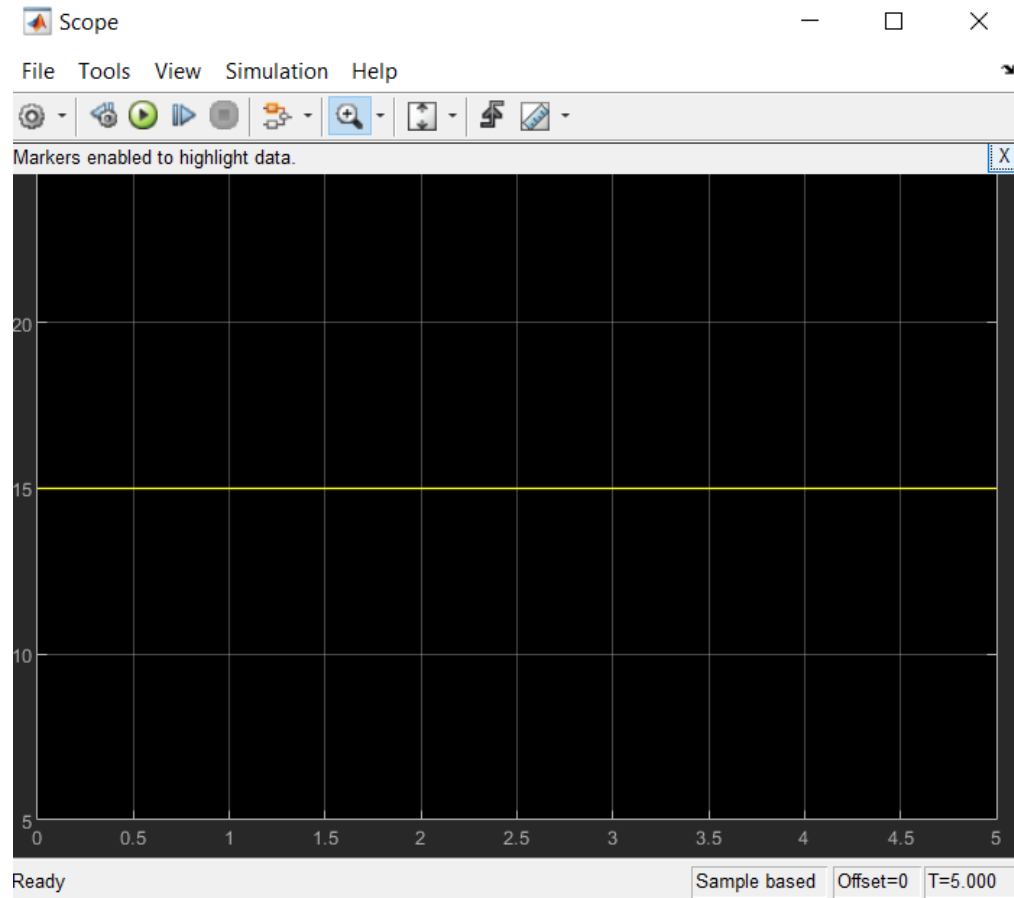
Impostare nella casella **Stop Time** la durata della simulazione (il valore di default è 10 secondi), e cliccare sul **pulsante Run** per avviare la simulazione.



**Attenzione:** il blocco Constant ha di default un «Sample Time» infinito. Tale valore va modificato in «-1», che fa sì che l'uscita del blocco venga riaggiornata in funzione del passo di avanzamento temporale del modello

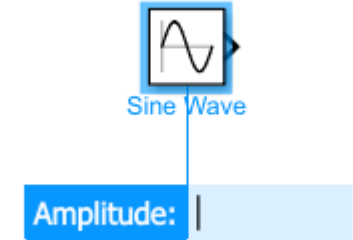


Al termine della simulazione fare doppio click sul blocco Scope per visualizzare il segnale in una finestra grafica:

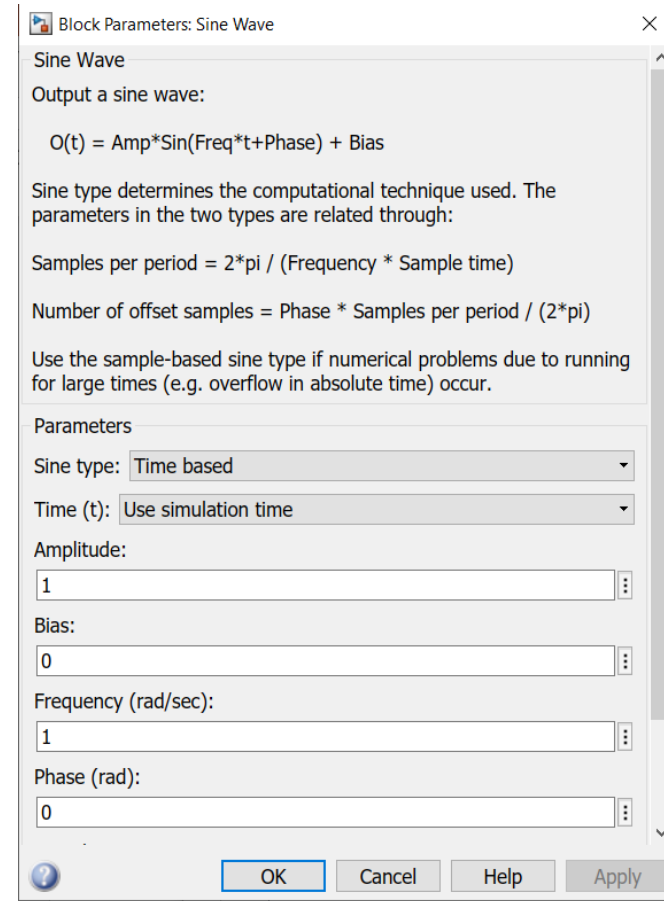


Generiamo un **segnale sinusoidale**.

Importiamo il blocco Sine Wave dalla libreria Sources e impostiamo il valore dell'ampiezza (il valore di default è 1)



Per impostare la frequenza e l'eventuale sfasamento fare doppio click sul blocco. Si apre una finestra di dialogo all'interno della quale vanno impostati i parametri di funzionamento.



## Sine Wave

Output a sine wave:

$$O(t) = \text{Amp} * \sin(\text{Freq} * t + \text{Phase}) + \text{Bias}$$

Puo essere impostato anche un valore costante di **Bias** (v. formula in alto).  
Al termine della configurazione premere il tasto OK.

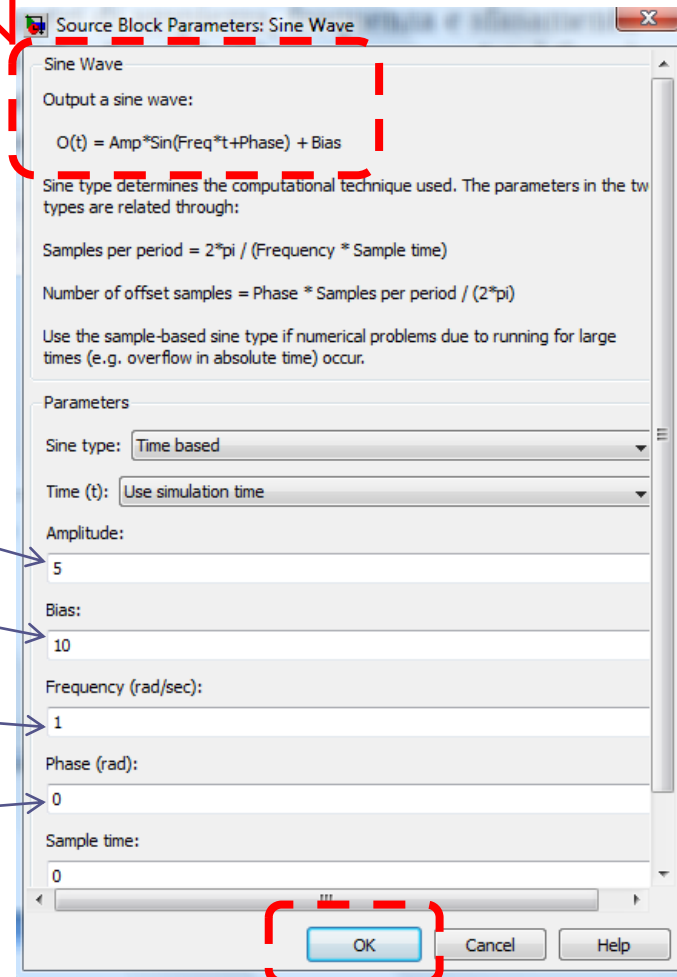
**Ampiezza**

**Bias**

**Frequenza**

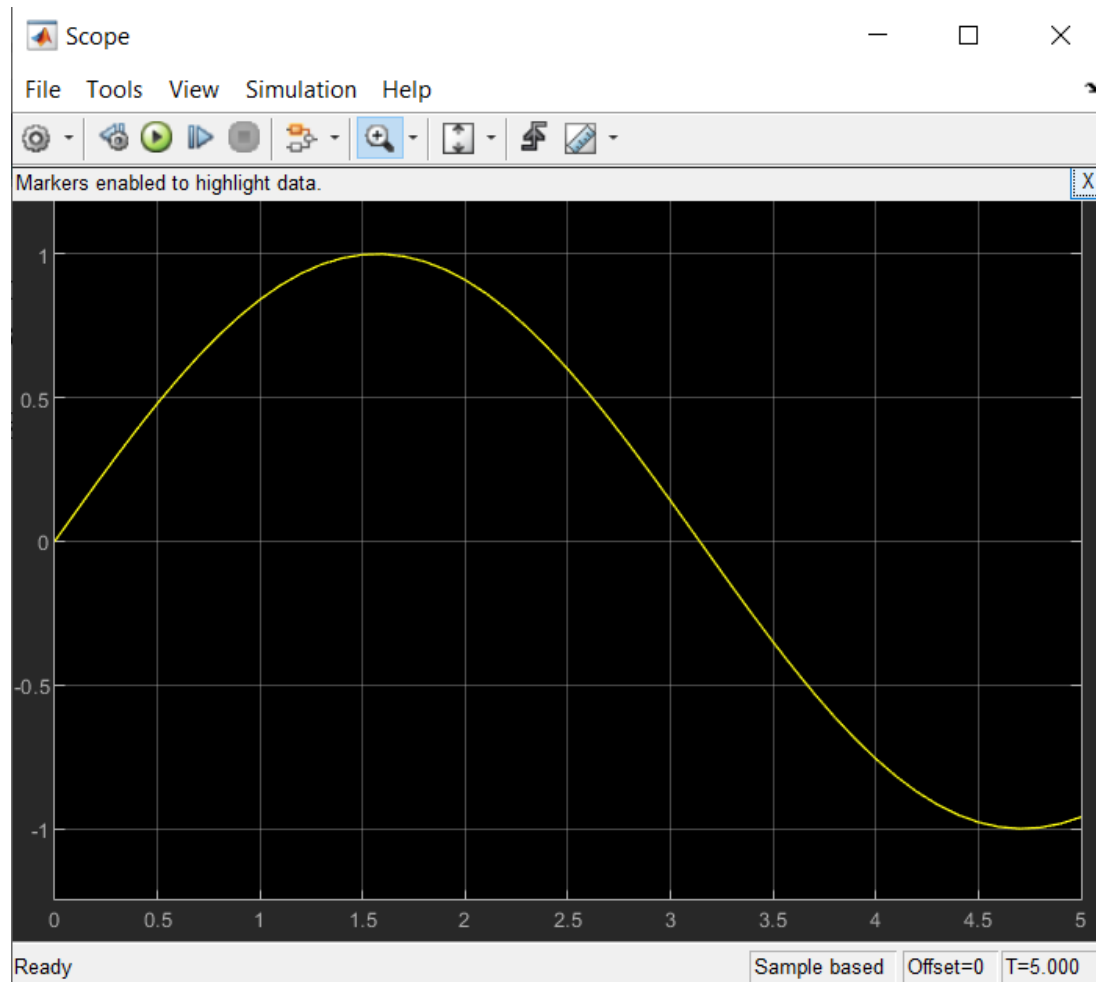
**Sfasamento**

**Tasto OK**



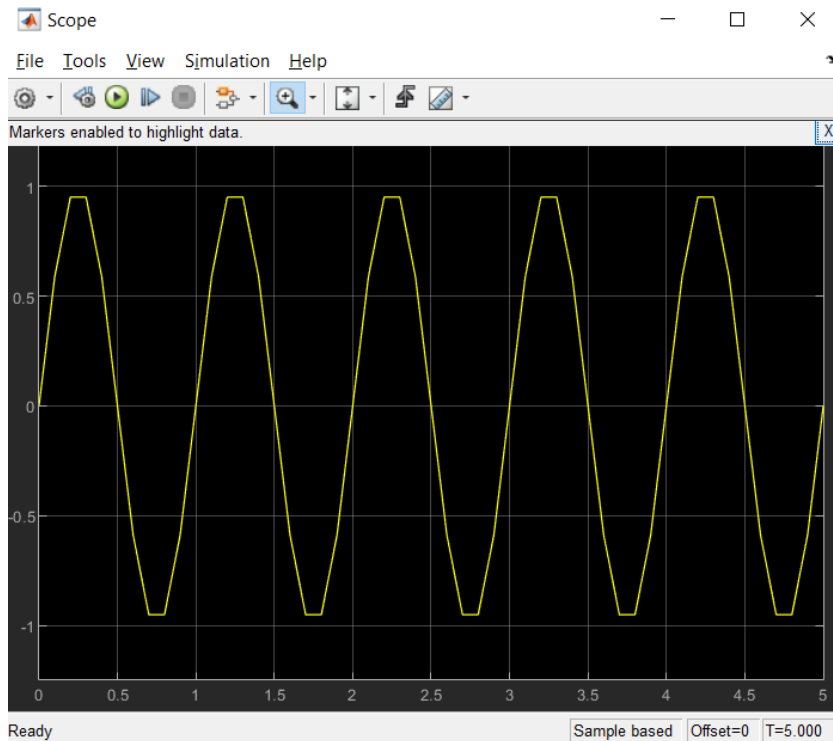
Si vuole generare il segnale  $10 + 5 \sin(t)$ . I parametri sono da settare come mostrato nella slide precedente

Scollegare il blocco constant dallo Scope, e collegare il blocco Sine Wave. Eseguire nuovamente la simulazione (pulsante RUN)



Ora si aumenti la frequenza della sinusoide da 1 rad/s a  $2\pi$  rad/s (1 Hz).

Si ripeta la simulazione.

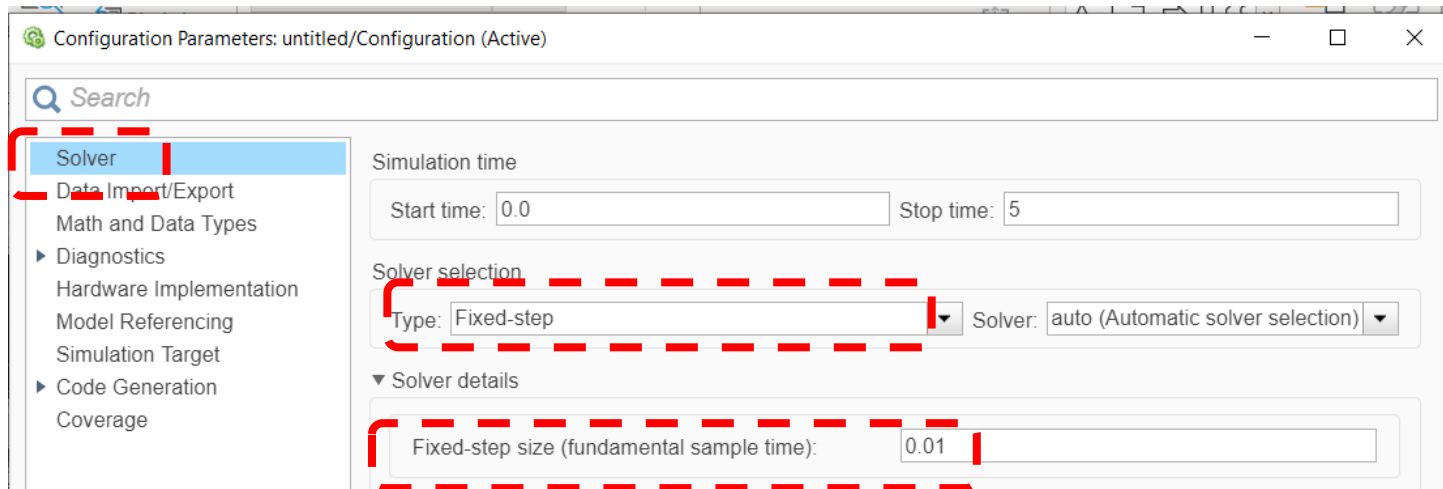


### Grafico “spigoloso”

Il grafico è stato realizzato interpolando un numero di punti insufficiente.

Si deve andare a modificare il “solver”, che definisce (fra le altre cose) il passo di discretizzazione temporale che viene impiegato nella esecuzione del modello.

Fare click con il tasto destro in qualunque punto dello schema e scegliere dal menu «Model Configuration Parameters» (Ctrl+E).



Nel menu Solver impostare il Solver Type ed il Fixed Step Size come in figura. In questo modo i segnali della simulazione saranno aggiornati (ricalcolati) ogni 0.01 secondi.

Queste impostazioni vanno eseguite ogni volta che si apre un nuovo modello vuoto. Impareremo successivamente come creare un template di configurazione che consenta di creare direttamente un nuovo modello con le impostazioni desiderate.

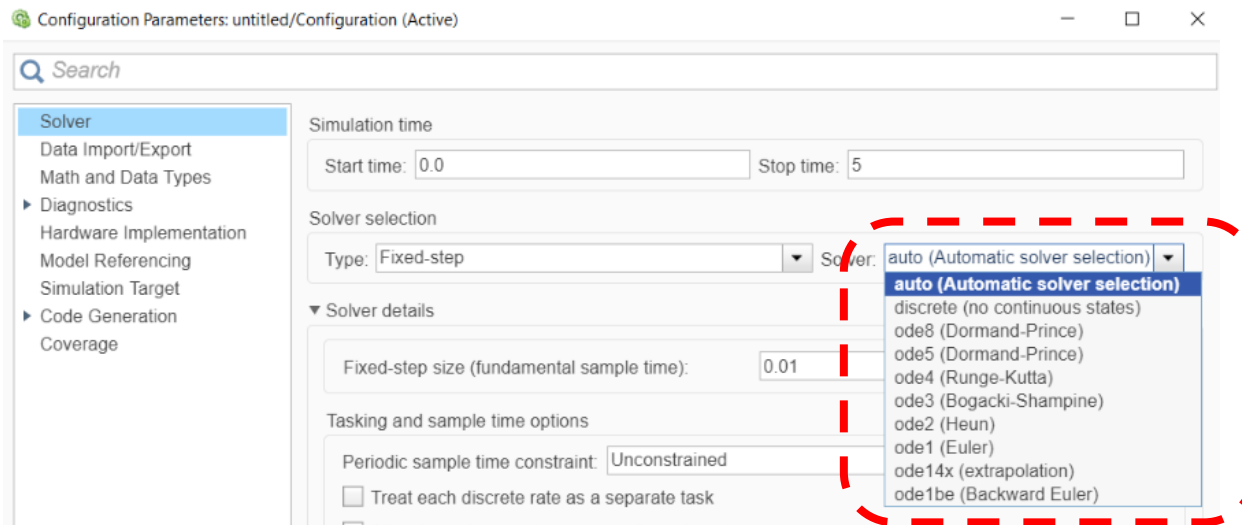
Naturalmente se si avesse necessita di generare una sinusoide a frequenza molto più elevata la scelta di 0.01 secondi per il fixed-step size potrebbe diventare non più appropriata, ed il valore dovrebbe essere ulteriormente ridotto.



E' disponibile una ampia varietà di solutori numerici. Nello screenshot sottostante sono riportate le varie opzioni di scelta per i solutori a passo fisso. La scelta del solutore con il quale di risolvono numericamente le equazioni differenziali del modello è ovviamente irrilevante per il semplice esempio in esame che non coinvolge alcun legame differenziale

Una **scelta ottimale per il solutore** bilancia, per il problema in esame, la precisione della soluzione e la mole di calcoli richiesta, che influenza il tempo di simulazione.

L'impostazione di scelta automatica del solver, che è la scelta di default, può dar luogo a soluzioni inaccurate in taluni casi «critici» come ad esempio modelli differenziali in cui intervengono segnali con scale di variazione temporale molto diverse fra loro (problemi «stiff») o modelli con elementi discontinui (non-smooth dynamics) e sono necessari solutori specifici.



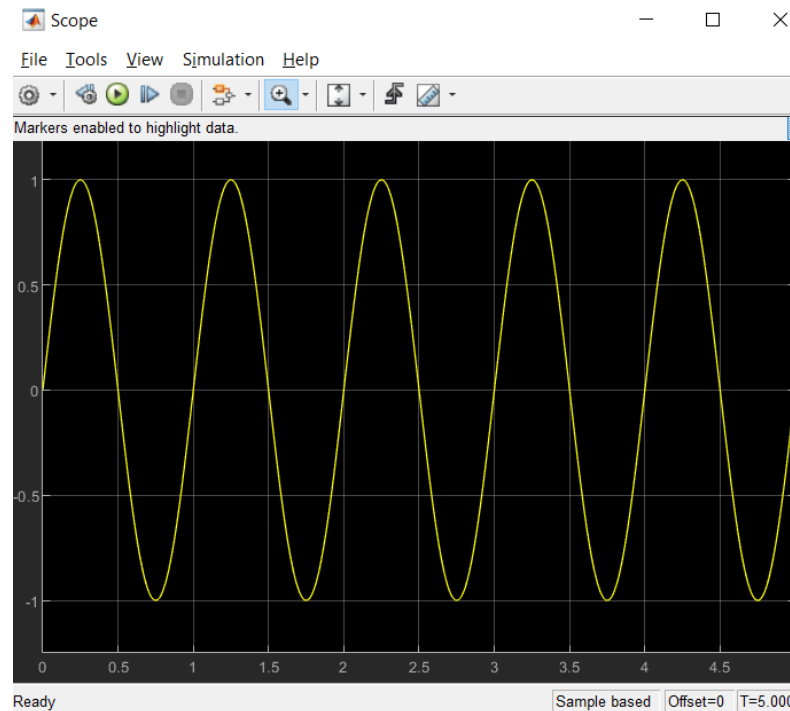
Per una discussione più approfondita in merito alla scelta del solutore:

<https://www.mathworks.com/help/simulink/ug/choose-a-solver.html>

## Si ripeta la simulazione e si riaggiorni il grafico

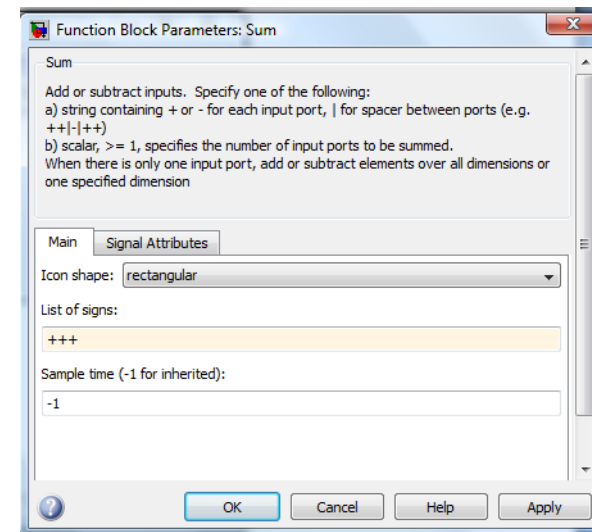
Il grafico della sinusoide è ora correttamente rappresentato.

In base alla scelta fatta per il passo fisso del solutore, vengono ora generati, e interpolati dal grafico, 100 campioni per ogni secondo di simulazione



Per visualizzare un segnale costituito dalla **somma di tre sinusoidi** importiamo nella pagina di lavoro due nuove istanze del blocco elementare Sine Wave, ed importiamo anche un blocco che rappresenti un nodo sommatore (blocco Sum dalla libreria dei Commonly Used Blocks).

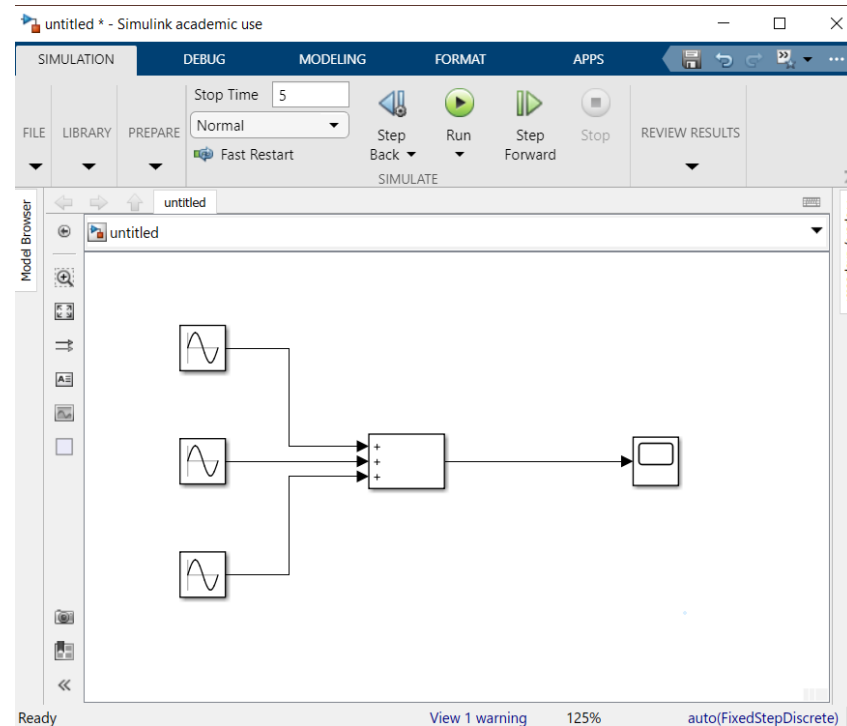
Il blocco Sum è parametrizzato per mezzo di una stringa (es. `++-+ -+`) la cui lunghezza corrisponde al numero di segnali in ingresso al blocco mentre il segno `+` o `-` definisce se il corrispondente ingresso sia da sommare agli altri termini o da sottrarre.



Scegliamo `+++` e impostiamo la Icon shape in rectangular.  
L'aspetto del blocco diventa



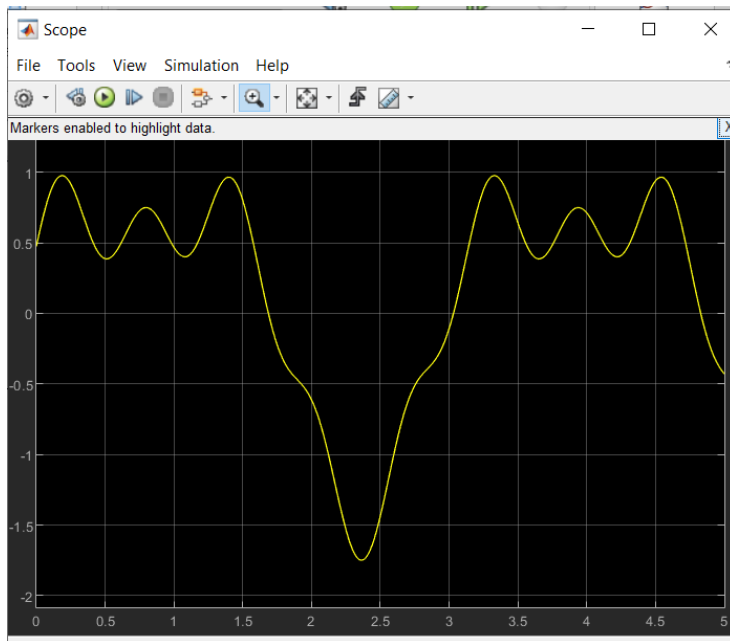
Si realizzi l'interconnessione riportata in Figura.



Si vuole generare il segnale  $\sin(2t) + 0.5 \cos(4t) + 0.25 \sin(10t - 0.1)$ .

Salvare il file (formato .slx)

	$\sin(2t)$	$0.5 \cos(4t)$	$0.25 \sin(10t - 0.1)$
<b>Amplitude</b>	1	0.5	0.25
<b>Bias</b>	0	0	0
<b>Frequency</b>	2	4	10
<b>Phase</b>	0	$\pi / 2$	$-0.1$



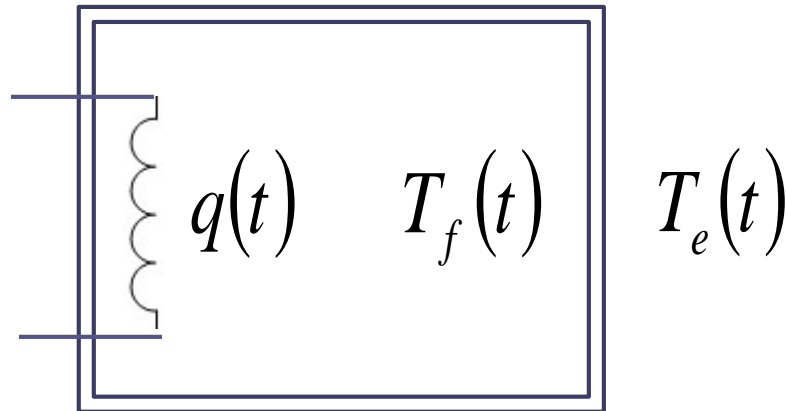
**N.B.**      $\cos(at) = \sin(at + \pi/2)$

**File:**     Crea3Sinusoidi.slx

Realizziamo ora un modello di simulazione dinamica vero e proprio.

## Simuliamo un sistema termico

Consideriamo il modello matematico di uno scambiatore di calore (ad es. un **forno**): un volume chiuso circondato da una parete e contenente un fluido all'interno del quale è presente una sorgente di calore  $q(t)$



Sia  $T_e(t)$  [°C] la temperatura esterna al volume,  $T_f(t)$  [°C] la temperatura del fluido interno al volume, e  $q(t)$  [J/s] una sorgente di calore interna al volume.

Sia  $C_f$  [J/K] la capacità termica (calore specifico) del fluido, e sia  $K_{ie}$  (J/K s) il coefficiente di scambio termico tra interno ed esterno.

Modello matematico:

$$C_f \dot{T}_f(t) = K_{ie}(T_e(t) - T_f(t)) + q(t)$$

## Modello matematico del fenomeno che si desidera simulare:

$$C_f \dot{T}_f(t) = K_{ie}(T_e(t) - T_f(t)) + q(t)$$

Equazione differenziale lineare del primo ordine

## Quantità note:

Parametri fisici del sistema	$C_f, K_{ie}$
Ingressi esterni	$q(t), T_e(t)$
Condizione iniziale	$T_f(0) = T_{f0}$

## Quantità da determinare:

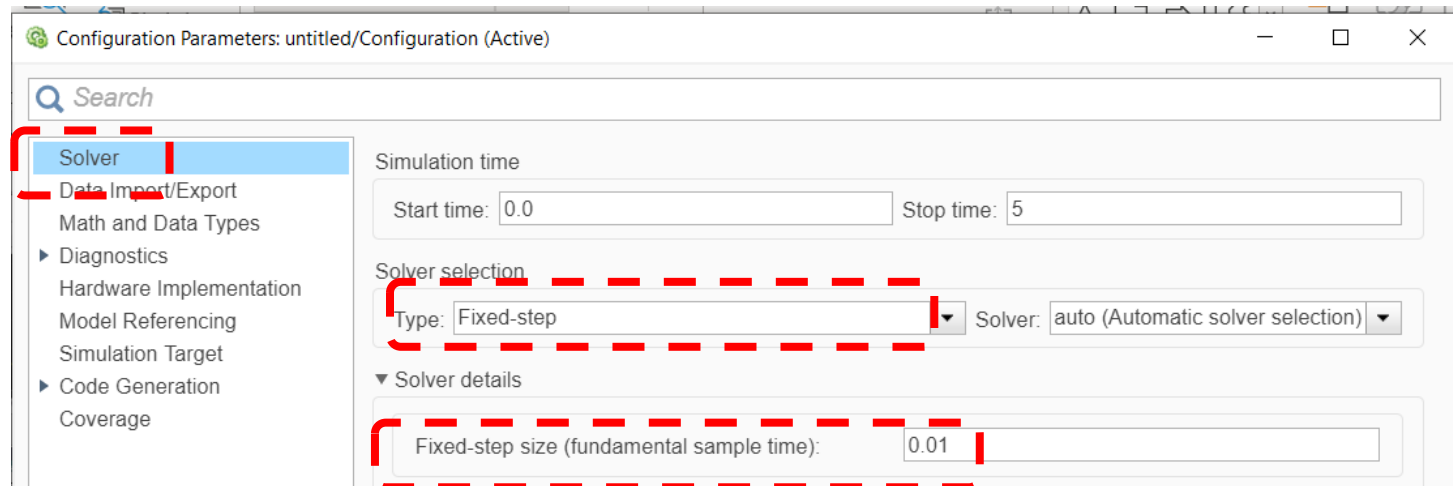
Funzione incognita	$T_f(t)$
--------------------	----------

Simulink risolve **numericamente** l'equazione differenziale, e rende accessibili i valori della funzione incognita in un certo insieme di istanti discreti  $T_f(t_1), T_f(t_2), \dots$

## Creazione di un template

Creiamo un modello template che sia parametrizzato mediante un solutore a passo fisso, con scelta automatica del particolare solutore, e valore del Fixed Step Size pari a 0.01 secondi.

Apriamo un Blank Model dalla Simulink Start Page e configuriamo le impostazioni desiderate dal Menu «Model Configuration Parameters».





Fatto cio, dopo avere premuto i pulsanti «Apply» ed «Ok», salvare il modello (pulsante Save) selezionando «Template»

Export untitled to Model Template

Create a template from a model to reuse or share the settings and contents of the model without copying the model each time.

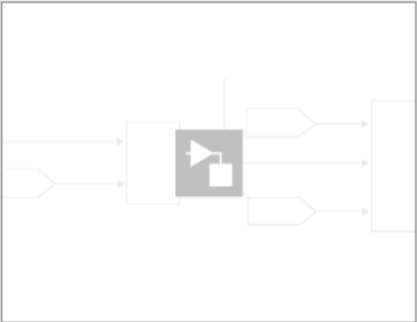
Title: Fixed Step Automatic Solver. Step=0.01.

Author: Alessandro Pisano

Group: My Templates

Description:

Modello configurato con un solutoe a passo fisso, scelta automatica del particolare solutore, e fixed step size di 0.01.

Image: 

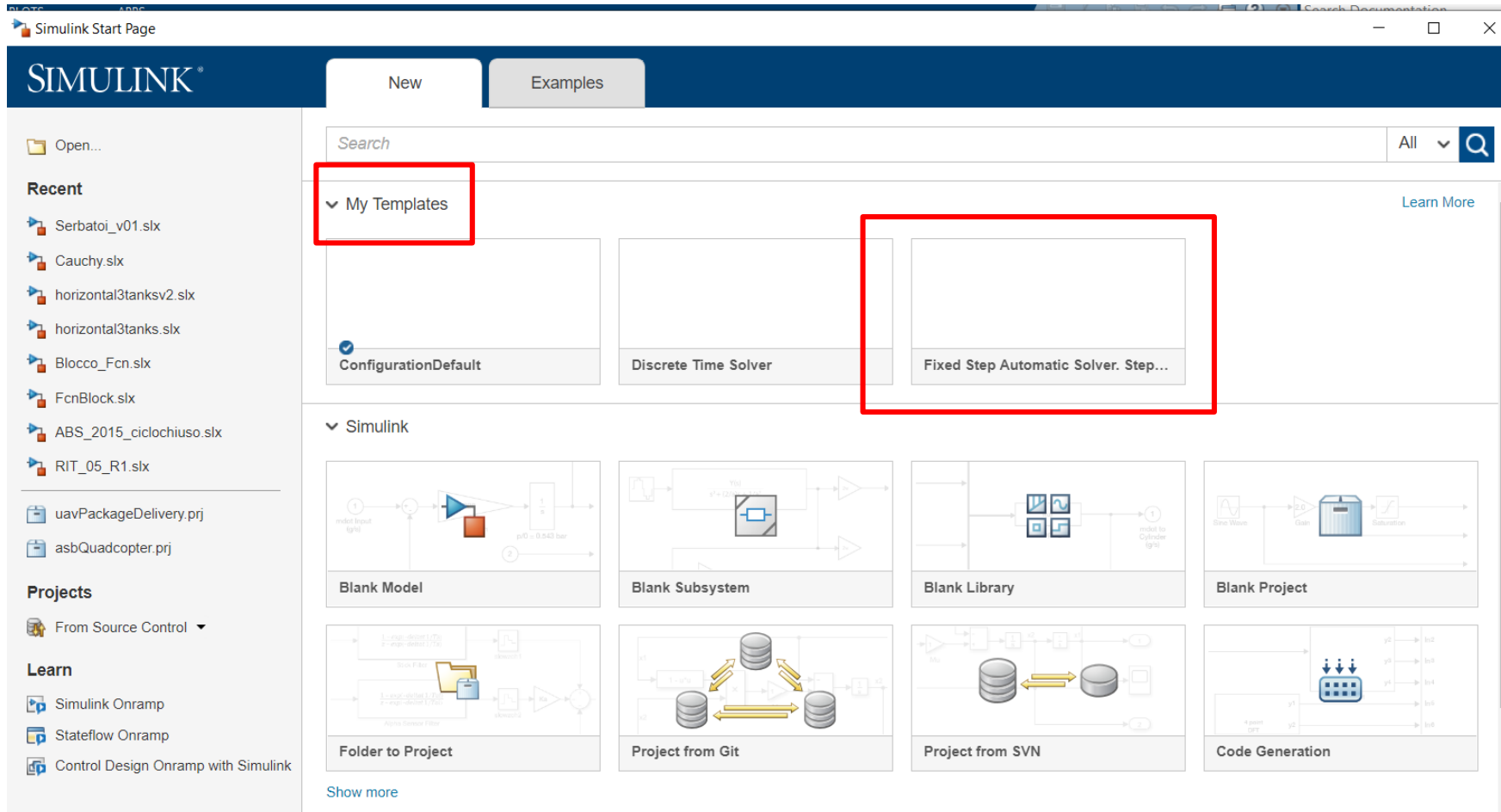
Change...

File location: C:\Users\pisan\Documents\MATLAB\FixedStep\_do101.sltx

Browse...

Help Export Cancel

Il Template è ora disponibile nella sezione My Templates della Simulink Start Page.



Per tradurre una equazione differenziale in uno «schema a blocchi» che possa essere implementato in Simulink per modellarla bisogna seguire una procedura sistematica

**Passo 1:** Riscrivere l'equazione differenziale in **forma esplicita** (cioè isolando alla sinistra dell'uguale la derivata di ordine più elevato della funzione incognita)

$$C_f \dot{T}_f(t) = K_{ie}(T_e(t) - T_f(t)) + q(t)$$

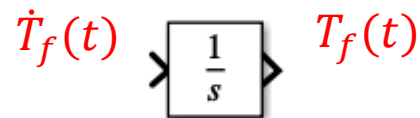


$$\dot{T}_f(t) = \frac{K_{ie}}{C_f}(T_e(t) - T_f(t)) + \frac{1}{C_f}q(t)$$

**Passo 2:** Importare nella pagina di lavoro (aprirne una ex-novo a partire dal template creato poco prima) un numero di blocchi «Integrator» (libreria Commonly Used Blocks) pari all'ordine della equazione differenziale (1 in questo caso).

Ora completeremo lo schema in modo tale che la temperatura  $T_f(t)$  del fluido sia generata al terminale di uscita del blocco integratore

Se all'uscita del blocco integratore è presente la temperatura  $T_f(t)$  del fluido , necessariamente all'ingresso del blocco dovrà essere presente il segnale rappresentativo della derivata  $\dot{T}_f(t)$  (v. figura)



L'equazione differenziale riscritta in forma esplicita fornisce l'espressione di  $\dot{T}_f(t)$  in funzione delle altre costanti e variabili presenti nel modello. Il segnale da applicare in ingresso all'integratore è costruito implementando tale espressione mediante gli opportuni blocchi Simulink

$$\dot{T}_f(t) = \frac{K_{ie}}{C_f} (T_e(t) - T_f(t)) + \frac{1}{C_f} q(t)$$

Si inserisca a monte dell'integratore un sommatore con due terminali positivi di ingresso. Costruiamo separatamente i due contributi  $\frac{K_{ie}}{C_f} (T_e(t) - T_f(t))$  e  $\frac{1}{C_f} q(t)$  da sommare fra loro.

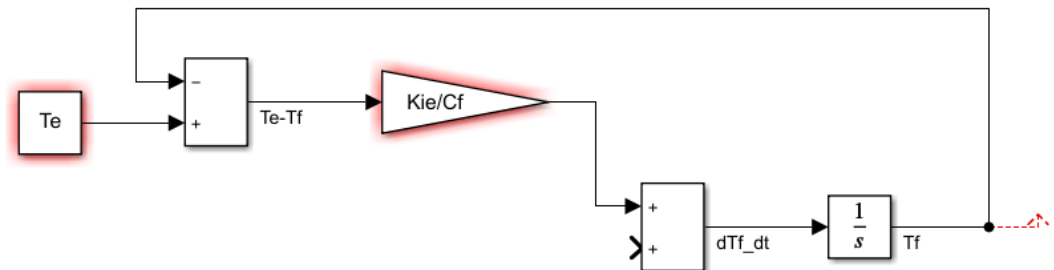
Consideriamo un valore costante  $T_e$  per la temperatura esterna  $T_e(t)$ .

Realizziamo il modello in forma parametrica utilizzando le variabili  $C_f$ ,  $K_{ie}$ ,  $T_e$  alle quali si attribuirà un valore mediante un file script.

**Primo contributo:**  $\frac{K_{ie}}{C_f} (T_e(t) - T_f(t))$

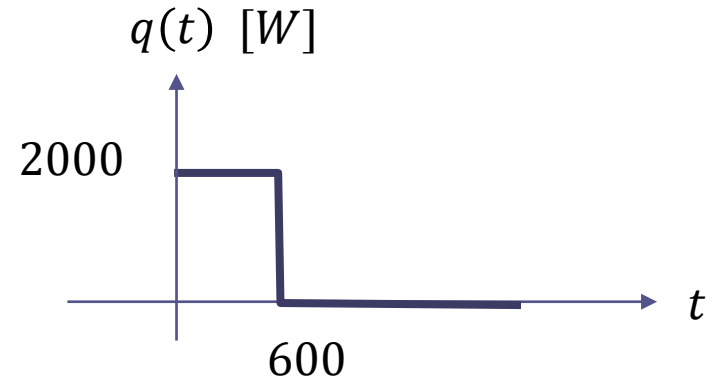
Serve un blocco gain per rappresentare il guadagno  $\frac{K_{ie}}{C_f}$ , un secondo blocco sommatore per calcolare la differenza  $T_e(t) - T_f(t)$  ed un blocco constant per generare la temperatura esterna  $T_e$

La temperatura  $T_f(t)$  del fluido è accessibile al terminale di uscita del blocco integratore.

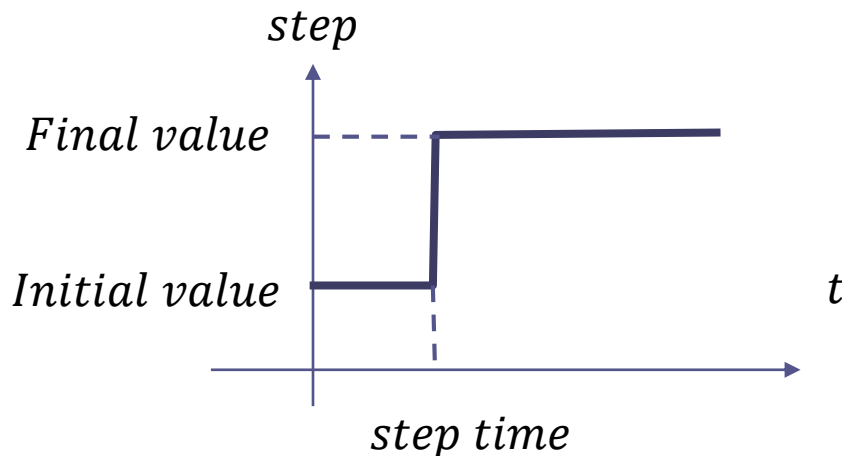


**Secondo contributo:**  $\frac{1}{c_f} q(t)$

Consideriamo il profilo a lato per la sorgente di calore  $q(t)$



Realizziamo il segnale  $q(t)$  mediante il blocco Step (Libreria Sources). Il blocco step genera un segnale avente la forma riportata nella figura sottostante. I parametri sono i valori iniziale e finale, e l'istante della transizione (step time).



*Parametri di configurazione  
del blocco Step*

*Initial value = 2000*

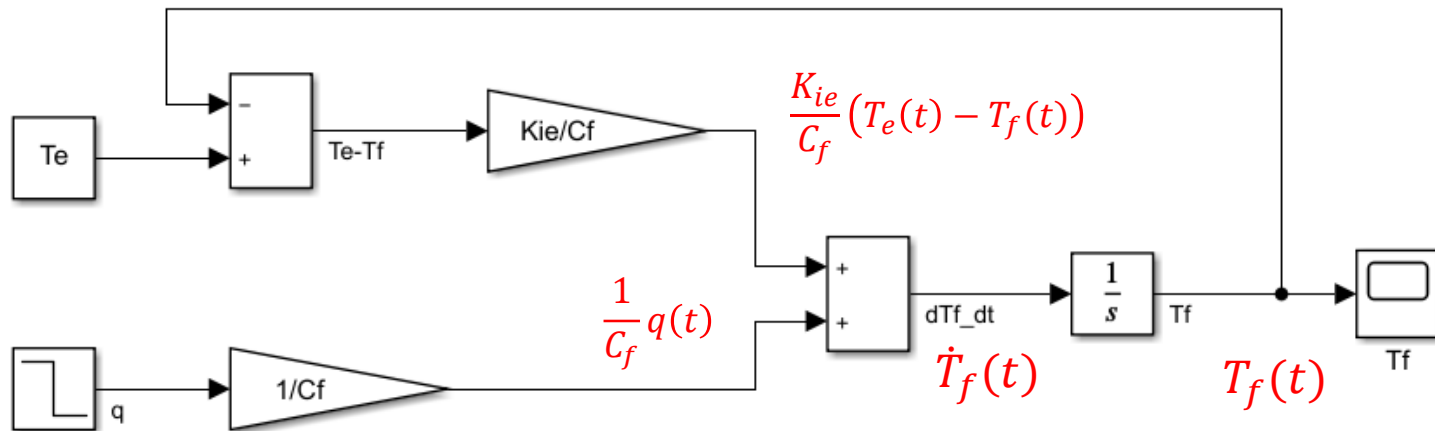
*Final value = 0*

*Step time = 600*

Salviamo nel workspace i parametri definiti nel seguente Script

**File: Forno\_dati.m**

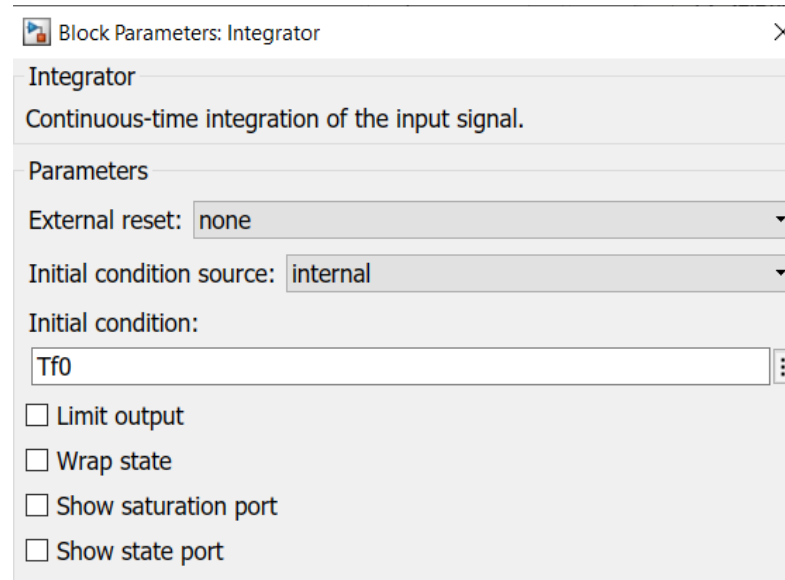
```
Tf0=15;      %Temp. iniziale del fluido [°C]
Te=20;       %Temp. esterna [°C]
Cf=1e4;      % J/K per 1 kg di sostanza
Kie=100;     % J / K s , per 1 m^2 di superficie di scambio
q=2000;      % J/s q=2kW
```



Si noti come all'ingresso dell'integratore, punto cui corrisponde il segnale  $dT_f/dt$ , venga

“costruito” elemento per elemento il segnale  $\dot{T}_f(t) = \frac{K_{ie}}{C_f} (T_e(t) - T_f(t)) + \frac{1}{C_f} q(t)$

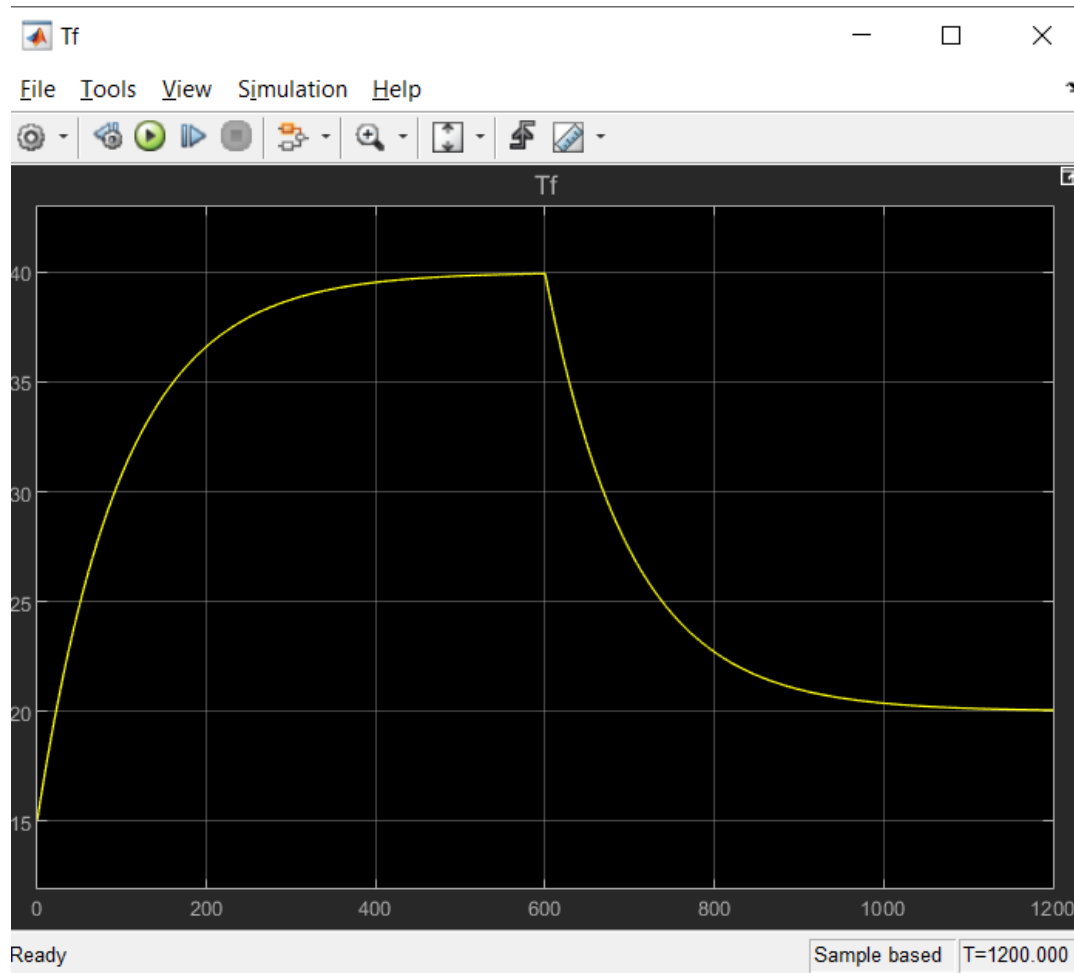
Il parametro  $T_{f0}$ , il valore della temperatura all'istante iniziale, si imposta come parametro di configurazione del blocco Integrator (Initial Condition).



La durata della simulazione si imposta nella casella **Stop Time**. Impostiamo come durata 1200 secondi.

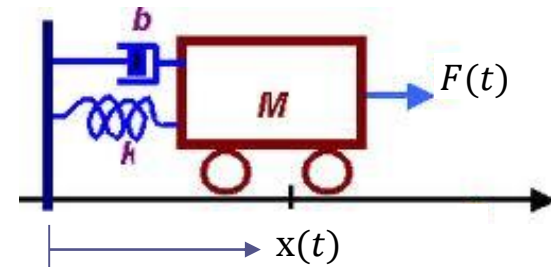


Si avvii la simulazione. Facendo doppio click sul blocco Scope si visualizza il grafico temporale della temperatura del fluido interno al volume.



Ora modelliamo un sistema dinamico più complicato.

## Sistema massa-molla-smorzatore



Un semplice sistema dinamico di derivazione meccanica il cui comportamento è descritto da una **equazione differenziale lineare del secondo ordine**:

$$M \ddot{x}(t) + b \dot{x}(t) + kx(t) = F(t)$$

$$M = 10 \text{ kg}$$

$$b = 5 \text{ N s / m}$$

$$k = 20 \text{ N / m}$$

Applichiamo la medesima procedura impiegata nell'esempio precedente

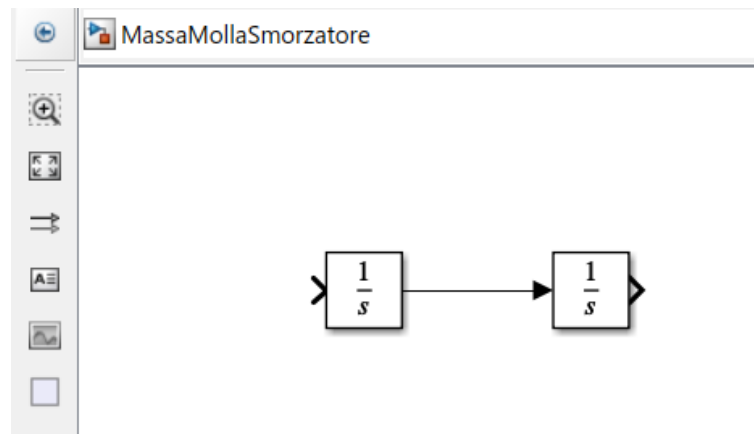
Passo 1: Riscrivere l'equazione differenziale in **forma esplicita** (cioè isolando alla sinistra dell'uguale la derivata di ordine più elevato della funzione incognita)

$$M \ddot{x}(t) + b \dot{x}(t) + kx(t) = F(t)$$



$$\ddot{x}(t) = -\frac{b}{M} \dot{x}(t) - \frac{k}{M} x(t) + \frac{1}{M} F(t)$$

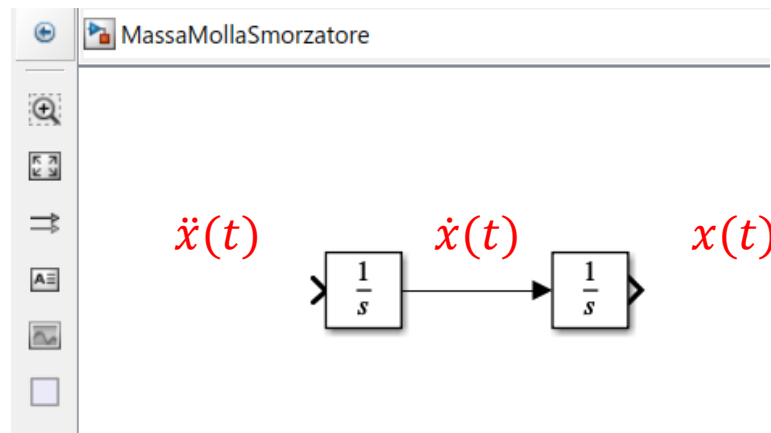
Passo 2: Importare nella pagina di lavoro (crearne una ex-novo) un numero di blocchi «Integrator» (libreria Commonly Used Blocks) pari all'ordine della equazione differenziale (2 in questo caso) e disporli in serie fra loro collegandoli come in figura



Salvare il file come `massamollasmorzatore.xls`

Ora completeremo lo schema in modo tale che la posizione del carrello sia generata al terminale di uscita del secondo integratore sulla destra

Se all'uscita dell'ultimo blocco integratore è presente la posizione  $x(t)$  del carrello, all'ingresso di tale integratore sarà presente il segnale rappresentativo della velocità  $\dot{x}(t)$ , e procedendo ulteriormente a ritroso all'ingresso del primo integratore, quello posizionato più a sinistra, dovrà essere presente l'accelerazione  $\ddot{x}(t)$  (v. figura)



Inseriamo a monte del primo integratore un blocco sommatore con tre terminali di ingresso, i primi due aventi segno negativo (stringa: " - - + ") per poter combinare fra loro i tre contributi che sommati fra loro generano l'accelerazione:

$$\ddot{x}(t) = -\frac{b}{M} \dot{x}(t) - \frac{k}{M} x(t) + \frac{1}{M} F(t)$$

Salviamo nel workspace i parametri definiti nel seguente Script

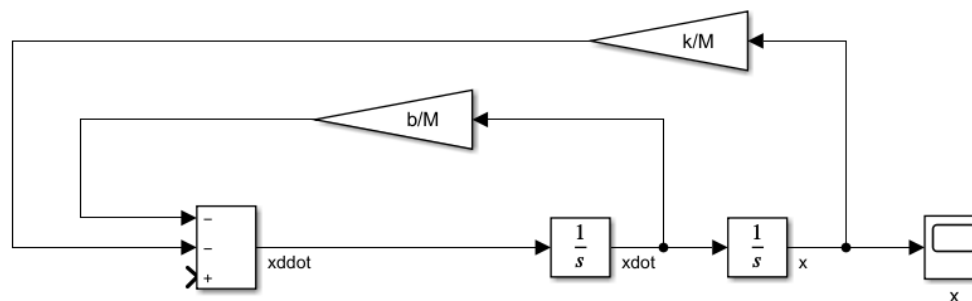
**File:** `MassaMollaSmorzatore_dati.m`

```
M=10; % Massa [kg]
b=5; % Coeff. attrito viscoso [N s/m]
k=20; % Costante elastica [N/m]

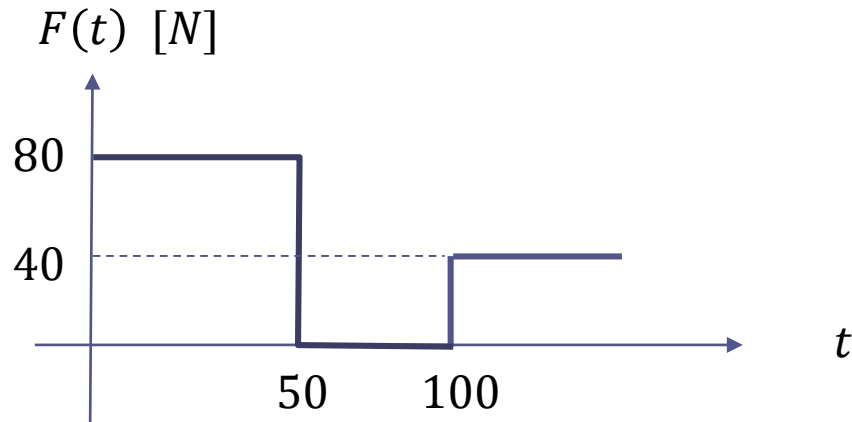
x0=0; % posizione iniziale [m]
v0=0; % velocita iniziale [m/s]
```

$$\ddot{x}(t) = -\frac{b}{M} \dot{x}(t) - \frac{k}{M} x(t) + \frac{1}{M} F(t)$$

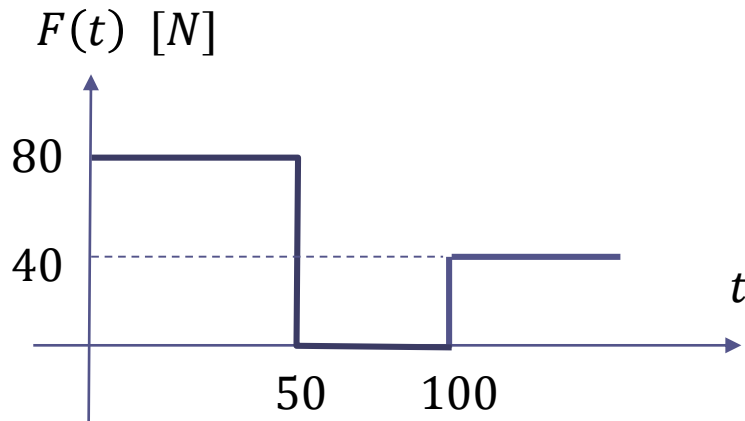
Schema parziale con i primi due contributi  $-\frac{b}{M} \dot{x}(t)$  e  $-\frac{k}{M} x(t)$



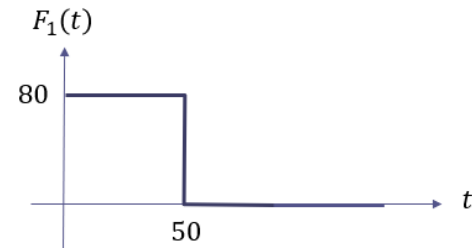
Ipotizziamo per la forza applicata  $F(t)$  il seguente profilo:



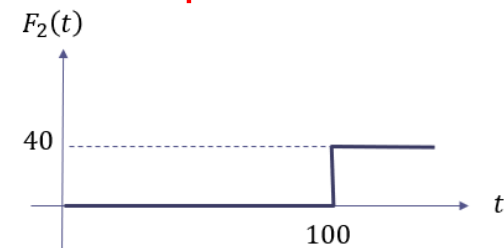
Possiamo creare questo profilo sommando fra loro le uscite di due blocchi step parametrizzati opportunamente



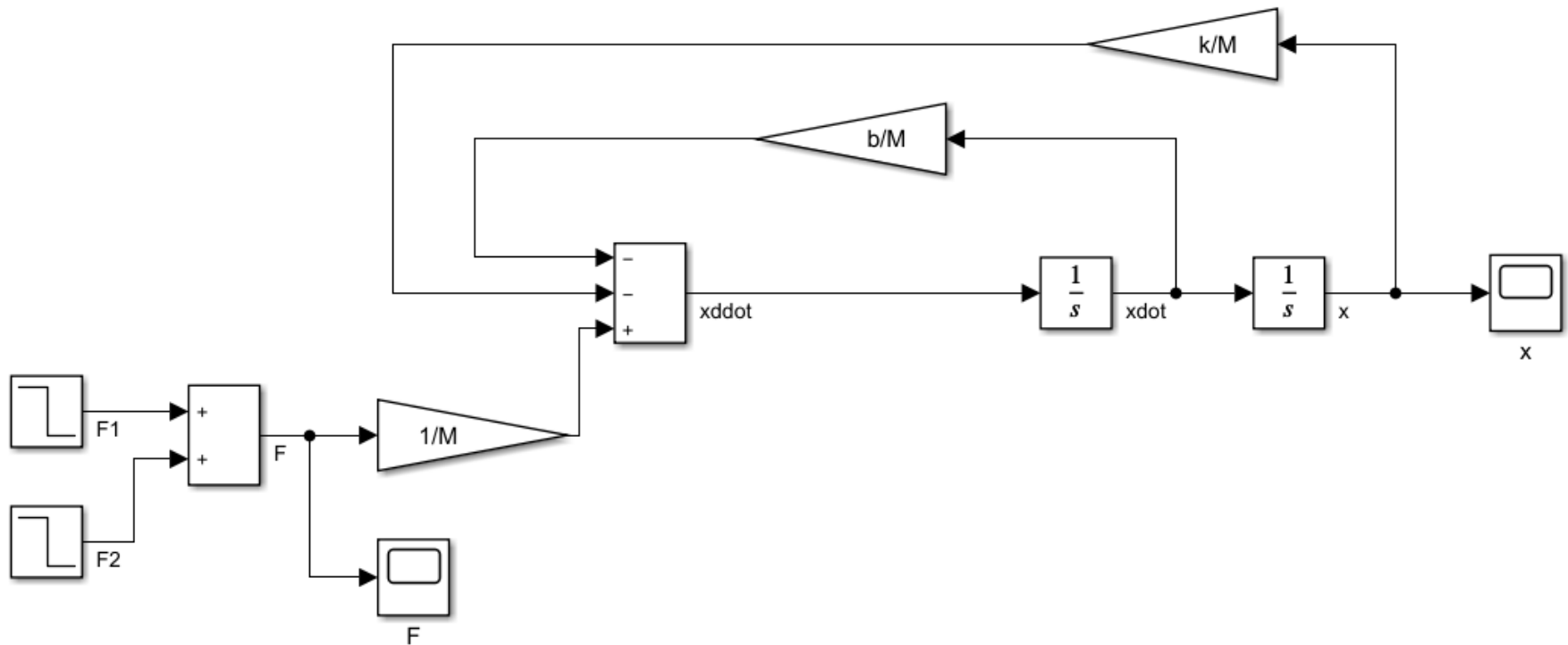
=



+



## Schema completo

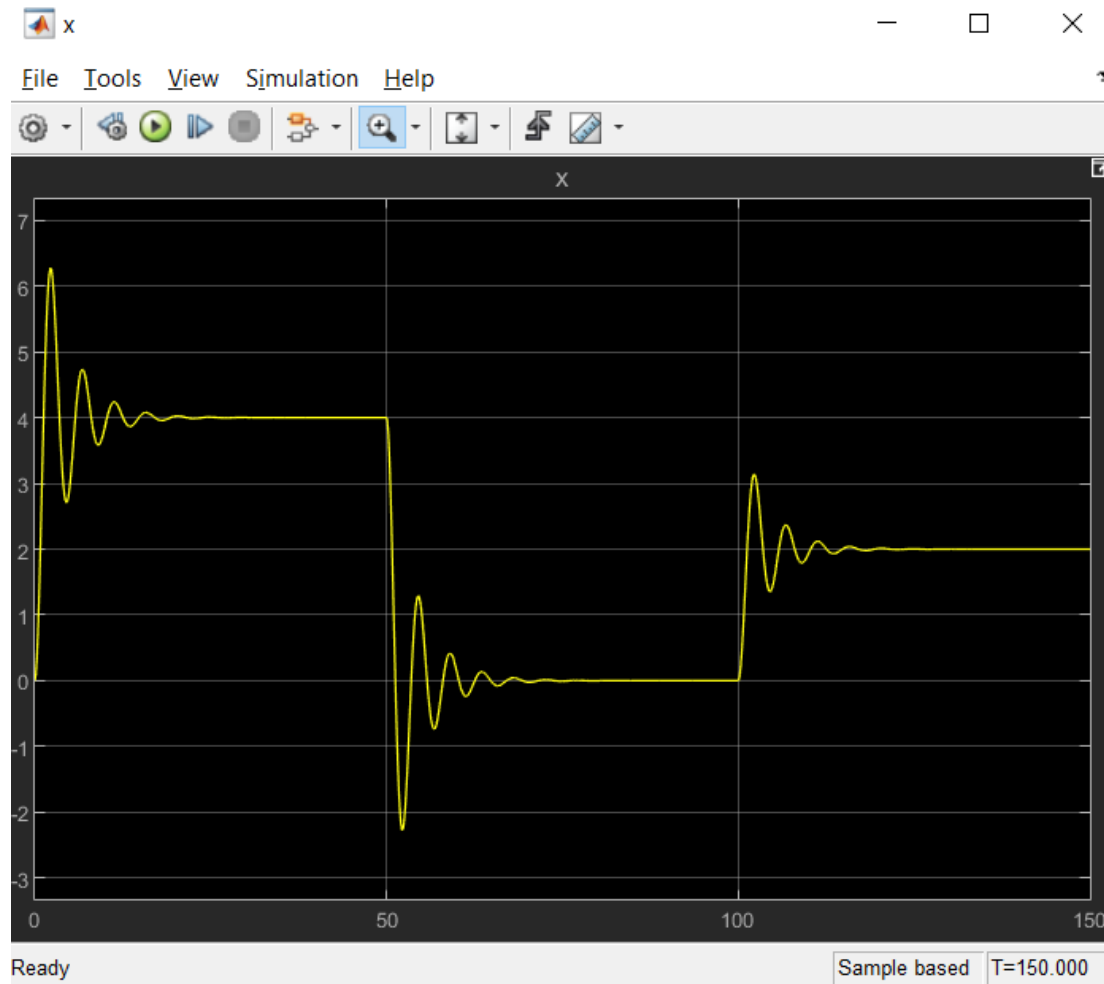


File: MassaMollaSmorzatore.slx



Parametrizziamo i blocchi Integratori con le condizioni iniziali della posizione e della velocità del carrello.

Fissiamo la durata della simulazione a 150 secondi.

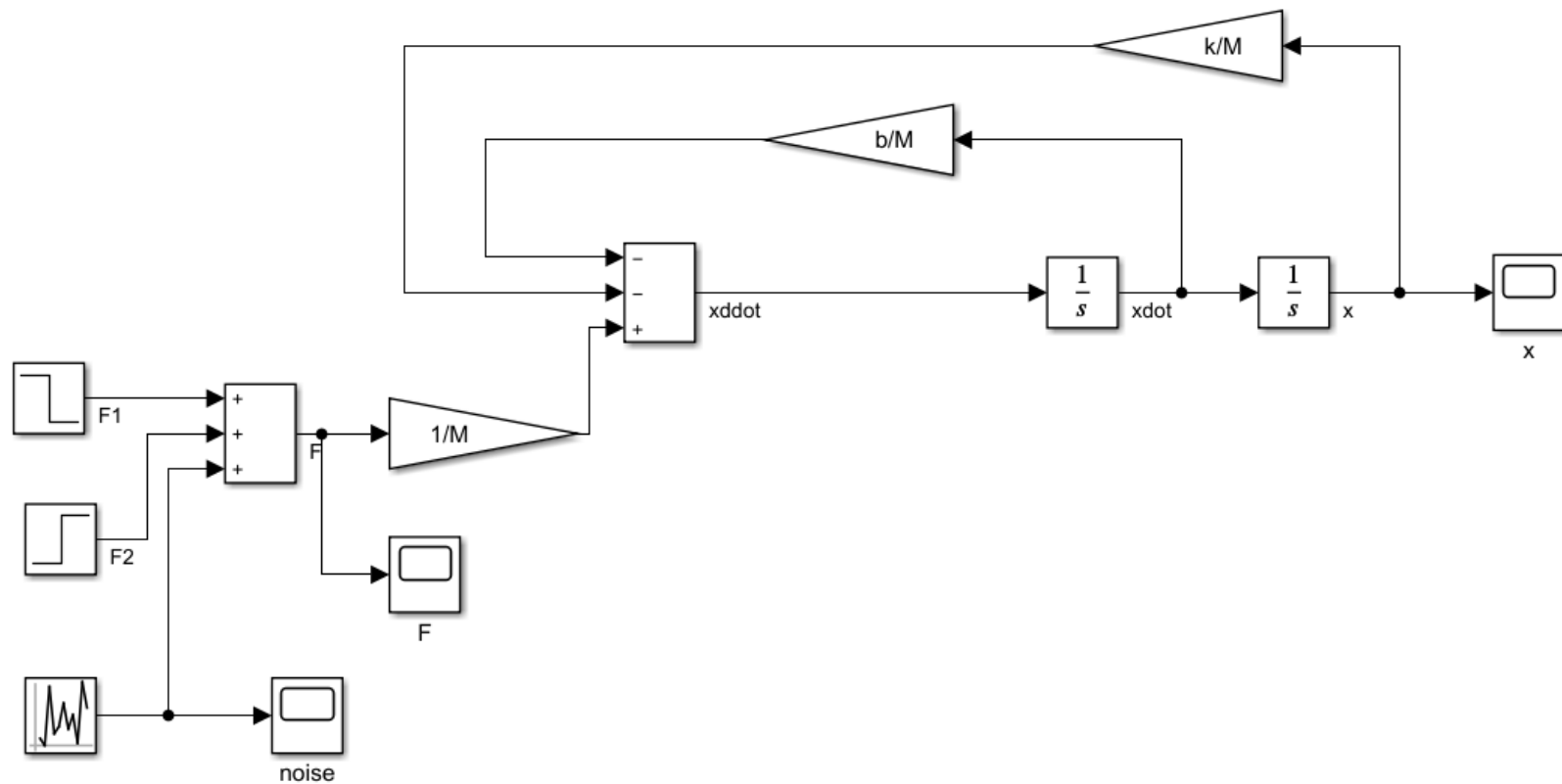


Modifichiamo il modello appena realizzato **sovrapponendo al segnale di forza una componente random**. Tale segnale consente di rappresentare fluttuazioni casuali della forza sviluppata dal sistema propulsivo.

Per generare in simulink un segnale random si hanno a disposizione i due blocchi «Random Number» e «Uniform Random Number» (libreria Sources). Il primo genera un segnale con distribuzione gaussiana con media e varianza assegnate, il secondo genera un segnale con distribuzione uniforme in un intervallo assegnato.

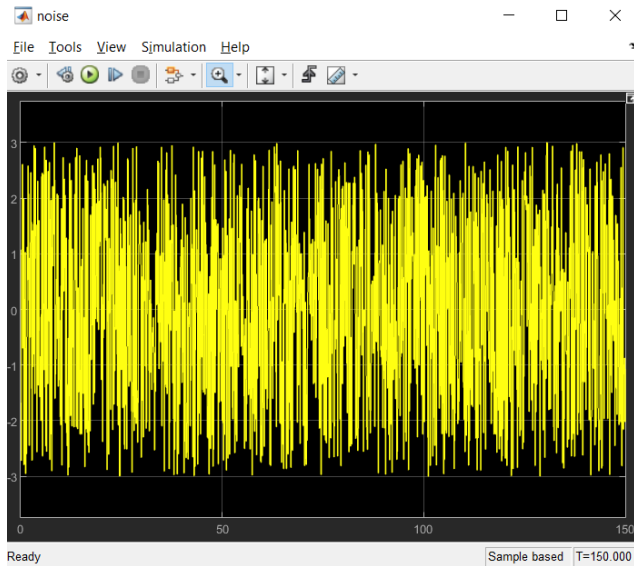
Utilizziamo il blocco «Uniform Random Number» con intervallo  $[-3,3]$  N (minimum = -3, maximum = 3)

## Schema completo con rumore random sulla forza applicata

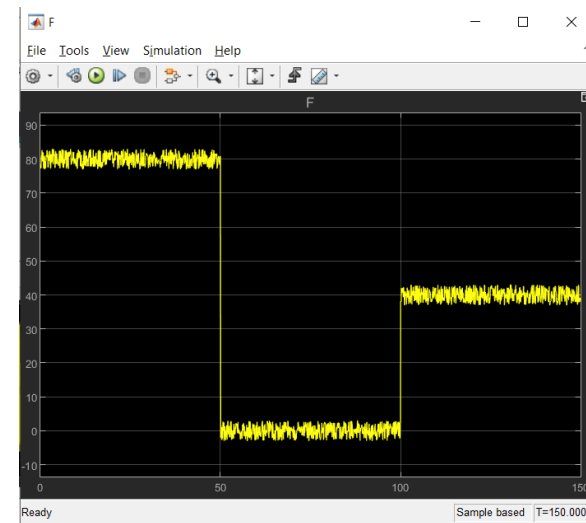


**File:** MassaMollaSmorzatore\_noise.slx

## Rumore sulla forza



## Forza applicata totale



## Posizione del carrello



Modifichiamo ulteriormente il modello inserendo dei termini **non lineari** nelle espressioni della forza di attrito e della forza di richiamo elastico.

## **Sistema massa-molla-smorzatore non lineare**

$$M \ddot{x}(t) + b_1 \dot{x}(t) + b_2 \dot{x}(t)|\dot{x}(t)| + k_1 x(t) + k_2 x^3(t) = F(t) + \eta_F(t)$$

Il termine di attrito  $b_2 \dot{x}(t)|\dot{x}(t)|$  viene denominato «drag». La caratteristica forza-deformazione della molla include un termine addizionale proporzionale al cubo della deformazione ( $k_1 x(t) + k_2 x^3(t)$ )

Nell'equazione sopraripotata è incluso anche il termine  $\eta_F(t)$  che rappresenta il rumore gaussiano che si sovrappone alla forza applicata

Abbiamo ora a che fare con una **equazione differenziale non lineare**.

Il ragionamento e la procedura per la realizzazione dello schema a blocchi Simulink restano invariati. Sarà più problematico modellare i vari termini non lineari. I blocchi Gain e Sommatore non sono più sufficienti. Serviranno blocchi ad hoc in grado di implementare funzioni non lineari coinvolte nelle equazioni del modello.

$$M \ddot{x}(t) + b_1 \dot{x}(t) + b_2 \dot{x}(t)|\dot{x}(t)| + k_1 x(t) + k_2 x^3(t) = F(t) + \eta_F(t)$$



equazione differenziale in **forma esplicita**

$$\ddot{x}(t) = -\frac{b_1}{M} \dot{x}(t) - \frac{b_2}{M} \dot{x}(t)|\dot{x}(t)| - \frac{k_1}{M} x(t) - \frac{k_2}{M} x^3(t) + \frac{1}{M} [F(t) + \eta_F(t)]$$

Si dovranno sempre importare nella pagina di lavoro due integratori da collegarsi in serie. Il sommatore a monte del primo integratore, mediante il quale si sommano fra loro i 5 contributi presenti alla destra dell'uguale nella eq. differenziale in forma esplicita, dovrà ora avere 5 terminali di ingresso.

```
clc, close all, clear all
```

```
M=10; % Massa [kg]
```

```
b1=5; % Coeff. attrito viscoso [N s/m]
```

```
b2=3; % Drag coefficient [N s^2/m^2]
```

```
k1=30; % Costante elastica [N/m]
```

```
k2=-0.04; % Costante elastica della componente non lineare [N/m^3]
```

```
x0=0; % posizione iniziale [m]
```

```
v0=0; % velocita iniziale [m/s]
```

```
%Parametri del profilo di forza F(t)
```

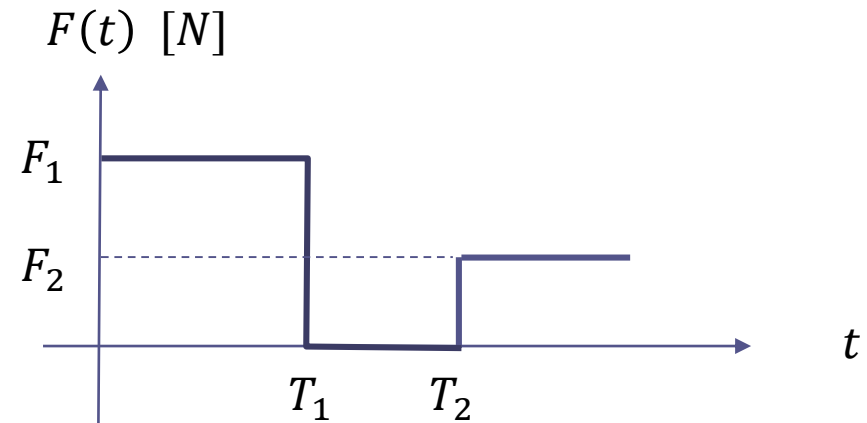
```
% F(t)= F1 per 0<t<T1
```

```
% F(t)=0 per T1<t<T2;
```

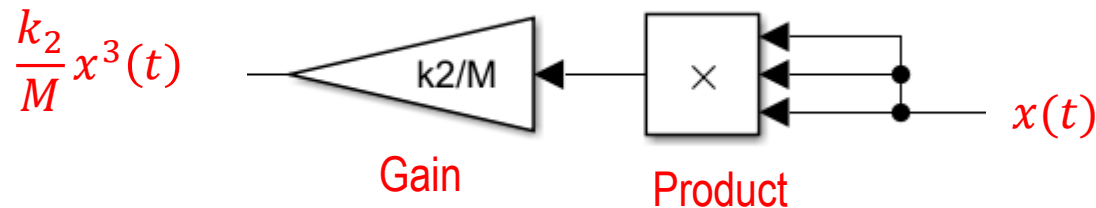
```
% F(t)=F2 per t > T2;
```

```
F1=80; F2= 40;
```

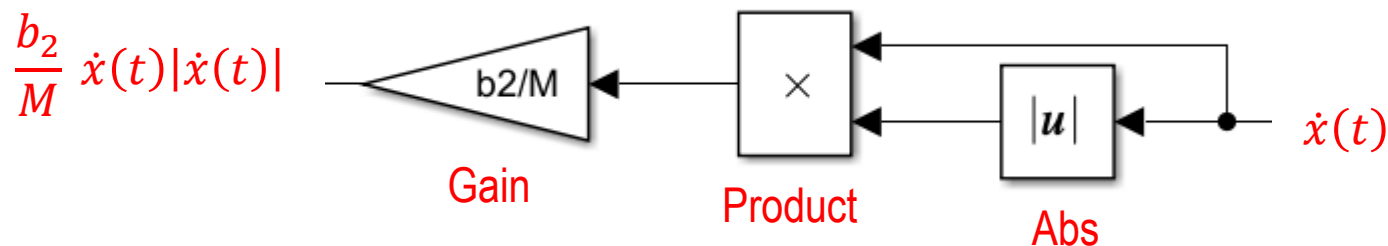
```
T1=50; T2=100;
```



L'aliquota proporzionale ad  $x^3(t)$  la si può realizzare mediante un blocco «Product», selezionando il numero di ingressi pari a 3 (v. figura sottostante)



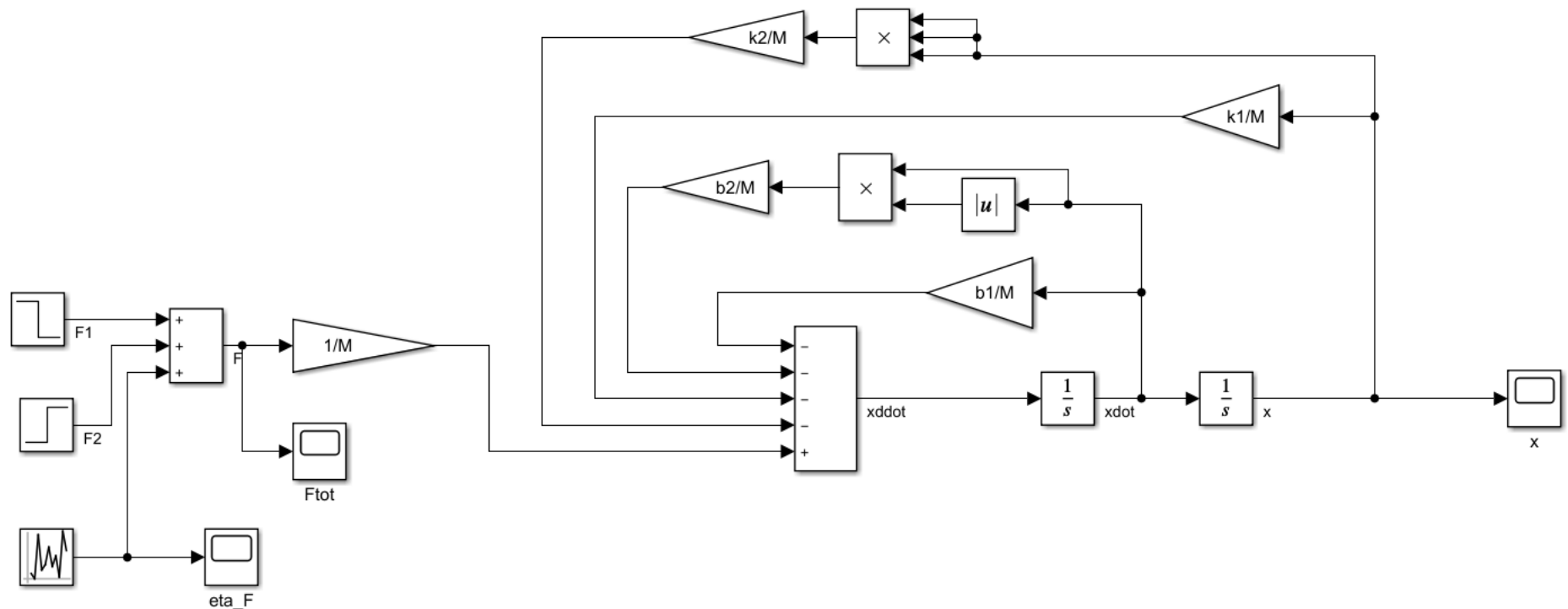
Per realizzare invece il termine di drag  $\frac{b_2}{M} \dot{x}(t)|\dot{x}(t)|$  è necessario impiegare un blocco Simulink denominato Abs, che estrae il valore assoluto del segnale in ingresso, disponibile nella libreria «Math Operations» (v. figura sottostante)





## Schema completo sistema MMS non lineare

$$\ddot{x}(t) = -\frac{b_1}{M} \dot{x}(t) - \frac{b_2}{M} \dot{x}(t)|\dot{x}(t)| - \frac{k_1}{M} x(t) - \frac{k_2}{M} x^3(t) + \frac{1}{M} [F(t) + \eta_F(t)]$$



File: MassaMollaSmorzatore\_noiseNonlineare.slx

## Evoluzione temporale della posizione del carrello (realizzato in Simulink)

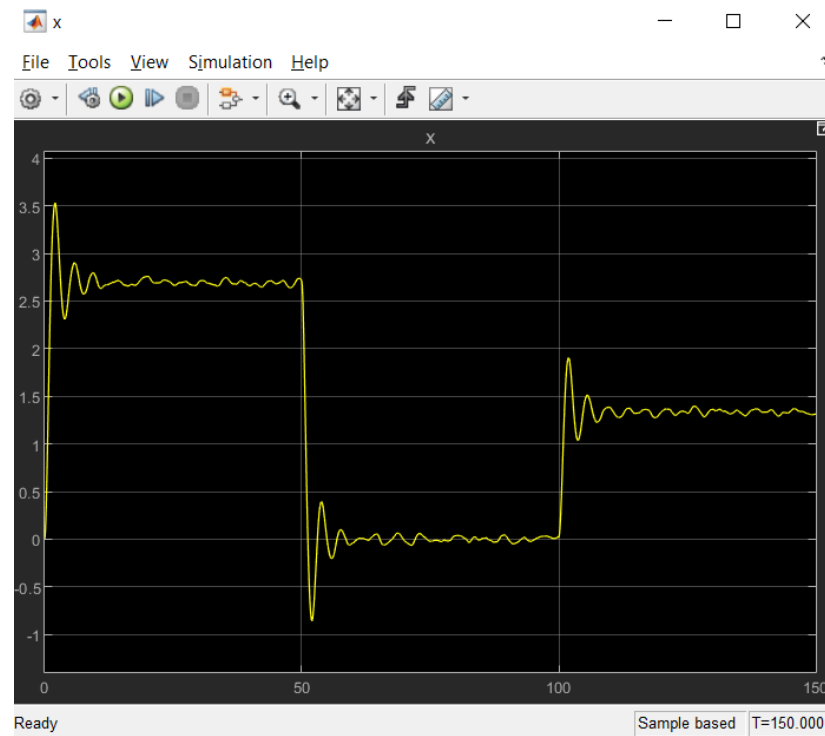
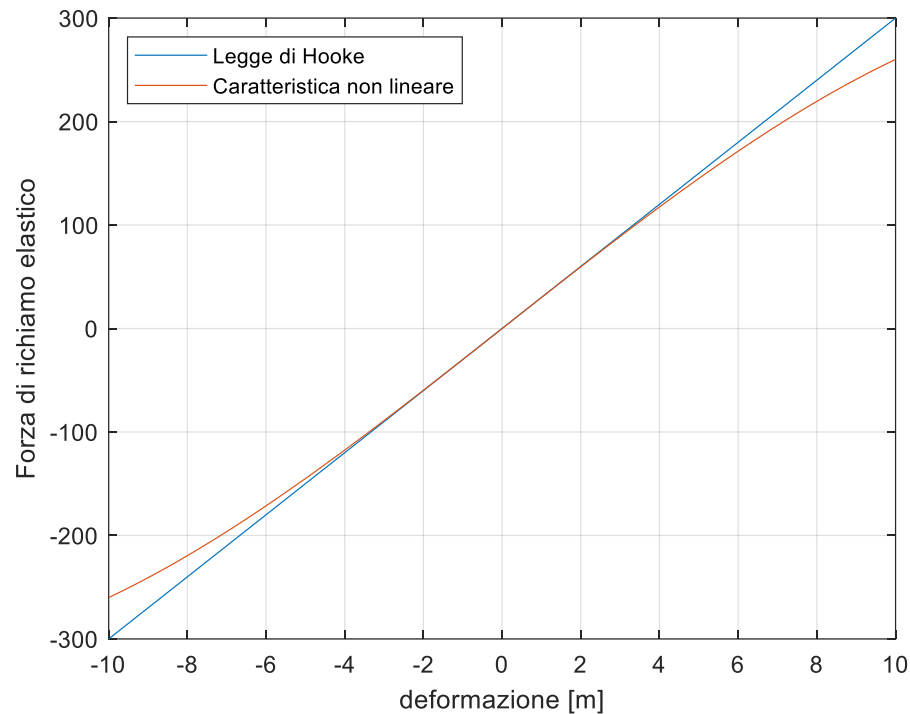


Grafico che confronta le caratteristiche forza-deformazione della molla nel caso lineare (legge di Hooke,  $k_2=0$ ) e non lineare (realizzato in matlab all'interno dello script di parametrizzazione)



## Esercizio: dinamica verticale di un razzo

$$m(t)\ddot{z}(t) + k\dot{z}|\dot{z}| = F(t) - m(t)g$$

$$m(t) = m_0 - m_1(1 - e^{-0.1t}) \quad m_0 > m_1$$

$$F(t) = F = \cos t$$

$$\ddot{z}(t) = \frac{1}{m(t)} [-k\dot{z}|\dot{z}| + F] - g$$

`m0=200;`

`m1=150;`

`k=4;`

`g=9.81;`

`F=5000;`



**Files:**     `Razzo.slx`

`Razzo_dati.m`

## Modellazione di un servomotore in corrente continua

Modelliamo il comportamento dinamico di un sistema elettromeccanico: un servomotore in corrente continua



Applicando una differenza di potenziale ai morsetti di alimentazione si provoca la circolazione di una corrente elettrica che induce una coppia di natura elettromagnetica che mette in rotazione l'albero del motore.

La dinamica del sistema è descritta da un **sistema di equazioni differenziali**.

## Modello matematico di un servomotore DC

$$V(t) = Ri(t) + L \frac{di(t)}{dt} + K_V \omega(t)$$

$$T_{em}(t) - T_{res}(t) = J \frac{d\omega(t)}{dt} + B\omega(t)$$

$$T_{em}(t) = K_T i(t)$$

### VARIABILI DEL SISTEMA

$v(t)$	tensione di alimentazione	$\omega(t)$	velocità angolare
$T_{em}(t)$	coppia elettromagnetica	$T_{res}(t)$	coppia resistente
$i(t)$	corrente di fase		

### PARAMETRI ELETTROMECCANICI

$K_V$	costante di forza contro-elettromotrice	$k_T$	costante di coppia
$R$	resistenza dell'avvolgimento	$J$	momento di inerzia all'albero
$L$	induttanza di avvolgimento	$B$	coefficiente di attrito viscoso

Ricaviamo le equazioni differenziali **in forma esplicita**

$$V(t) = Ri(t) + L \frac{di(t)}{dt} + K_V \omega(t)$$

$$T_{em}(t) - T_{res}(t) = J \frac{d\omega(t)}{dt} + B\omega(t)$$

$$T_{em}(t) = K_T i(t)$$



$$V(t) = Ri(t) + L \frac{di(t)}{dt} + K_V \omega(t)$$

$$K_T i(t) - T_{res}(t) = J \frac{d\omega(t)}{dt} + B\omega(t)$$



$$\frac{di(t)}{dt} = \frac{1}{L} [V(t) - Ri(t) - K_V \omega(t)]$$

$$\frac{d\omega(t)}{dt} = \frac{1}{J} [K_T i(t) - B\omega(t) - T_{res}(t)]$$

Così come per la rappresentazione in Simulink di una equazione differenziale, anche per quanto concerne la rappresentazione in Simulink di un sistema di equazioni differenziali il punto di partenza è la **risrittura del sistema di equazioni in forma esplicita**

$$\begin{aligned}\frac{di(t)}{dt} &= \frac{1}{L} [V(t) - Ri(t) - K_V \omega(t)] \\ \frac{d\omega(t)}{dt} &= \frac{1}{J} [K_T i(t) - B\omega(t) - T_{res}(t)]\end{aligned}$$

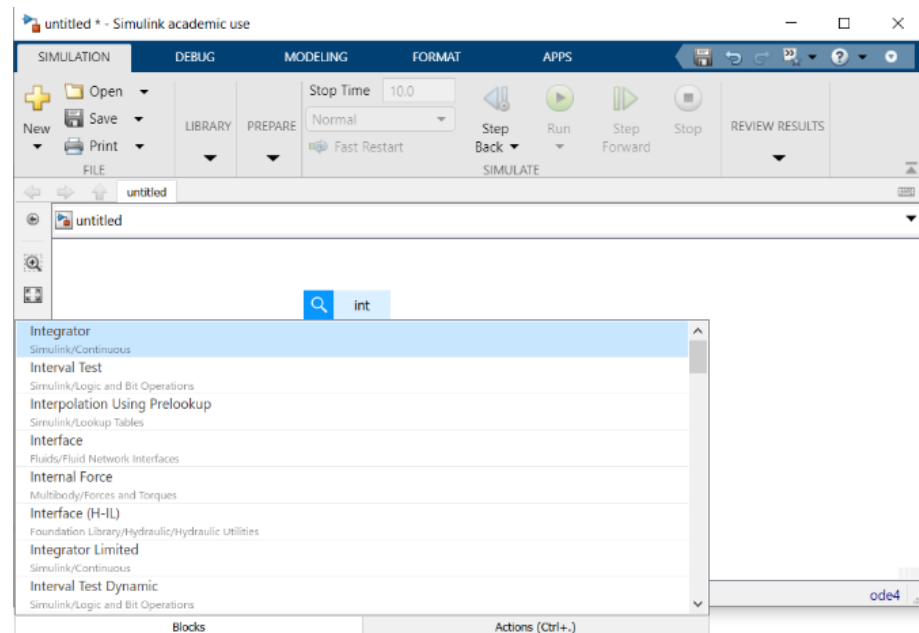
All'interno delle equazioni del modello la tensione applicata  $V(t)$  e la coppia resistente  $T_{res}(t)$  (che dipende dal carico applicato all'albero del motore) sono grandezze note (ingressi al sistema). Le grandezze incognite sono la velocità angolare  $\omega(t)$  e la corrente di fase  $i(t)$ .

Modelliamo in Simulink il sistema di equazioni. La procedura, come si vedrà, è pressochè identica a quella seguita per modellare il sistema termico ed il sistema massa-molla-smorzatore, con lievi differenze.

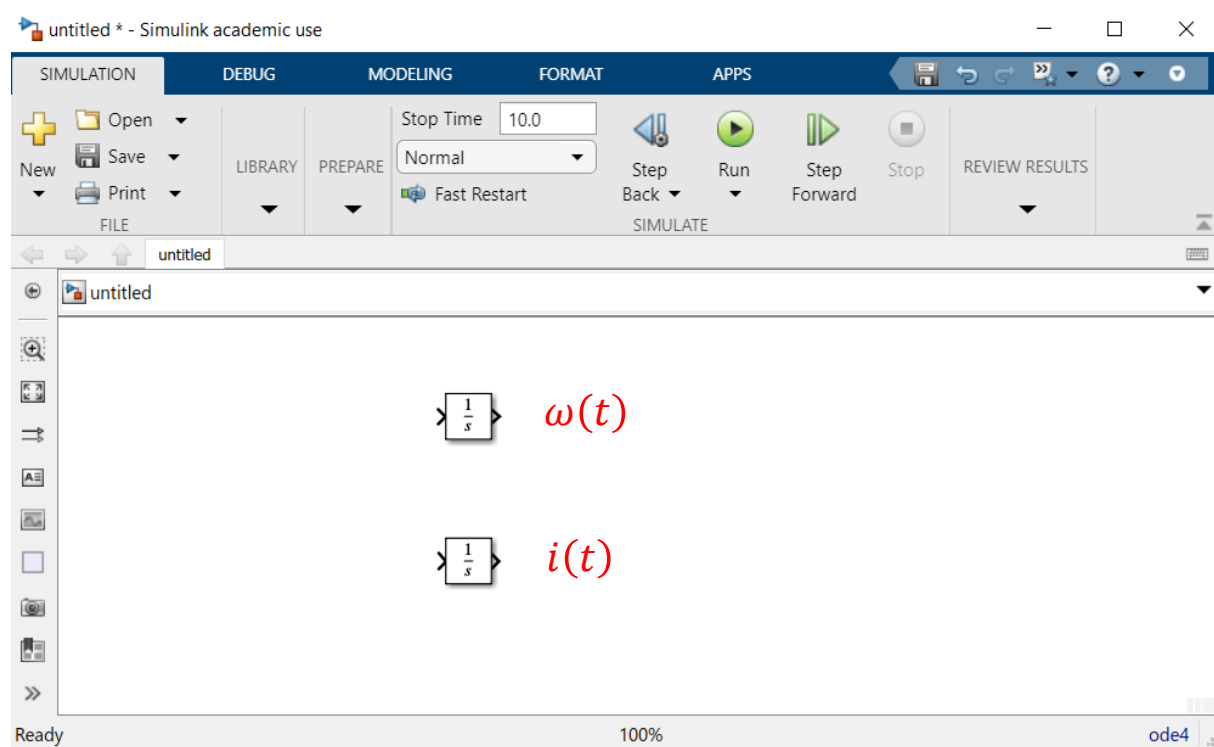


Dobbiamo importare nella pagina di lavoro un numero di blocchi «Integrator» pari all'ordine complessivo del sistema (in questo caso 2, perchè il sistema si compone di due equazioni accoppiate del primo ordine)

Per importare dei blocchi all'interno della pagina di lavoro è possibile, in alternativa alla procedura di drag-and-drop dalle rispettive librerie, fare doppio click in un punto qualunque dello schema e scrivere il nome del blocco (o la parte iniziale del nome). In esito a ciò, compare una lista di opzioni da cui scegliere il blocco che ci interessa

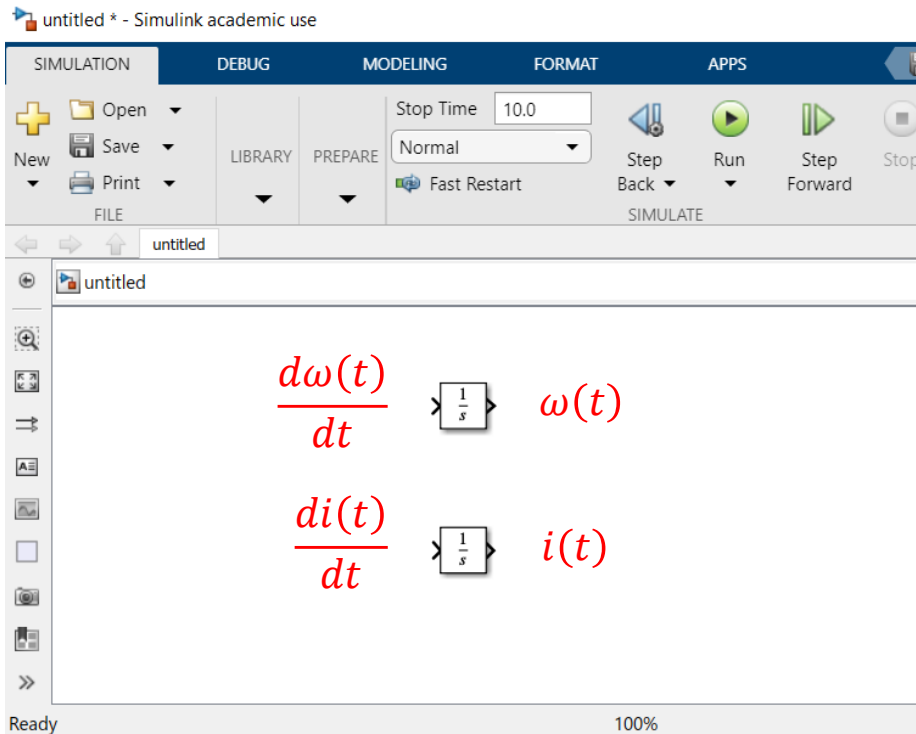


A differenza da quanto fatto in precedenza, non collegheremo più un serie gli integratori, ma li disporremo su righe differenti.



Ora completeremo lo schema in modo tale che le due variabili incognite, la velocità angolare e la corrente di fase, siano generate ai terminali di uscita dei due blocchi integratori

Chiaramente, se ai terminali di uscita degli integratori sono presenti la velocità angolare  $\omega(t)$  e la corrente di fase  $i(t)$  ai rispettivi terminali di ingresso saranno presenti le rispettive derivate temporali



$$\frac{di(t)}{dt} = \frac{1}{L} [V(t) - Ri(t) - K_V \omega(t)]$$

$$\frac{d\omega(t)}{dt} = \frac{1}{J} [K_T i(t) - B\omega(t) - T_{res}(t)]$$

Per modellare il funzionamento del motore debbono essere «costruite» ed applicate in ingresso ai due integratori le espressioni dei segnali  $\frac{d\omega(t)}{dt}$  e  $\frac{di(t)}{dt}$ , definite dalle equazioni in forma esplicita

## Realizziamo ed eseguiamo il seguente Script

```
% Parametri modello motore DC
% Parametri elettromeccanici
R=0.8;      % resistenza
L=1e-3;     % induttanza
KT=0.3;     % costante di coppia
Kv=0.3;     % costante di forza c.e.m.
J=1;        % momento di inerzia
B=0.8;      % coefficiente di attrito viscoso

% Ingressi
V=10;       % tensione di alimentazione
Tres=0;     % coppia resistente

% Condizioni iniziali
omega0=0;   % condizione iniziale della velocita
i0=0;       % condizione iniziale della corrente
```

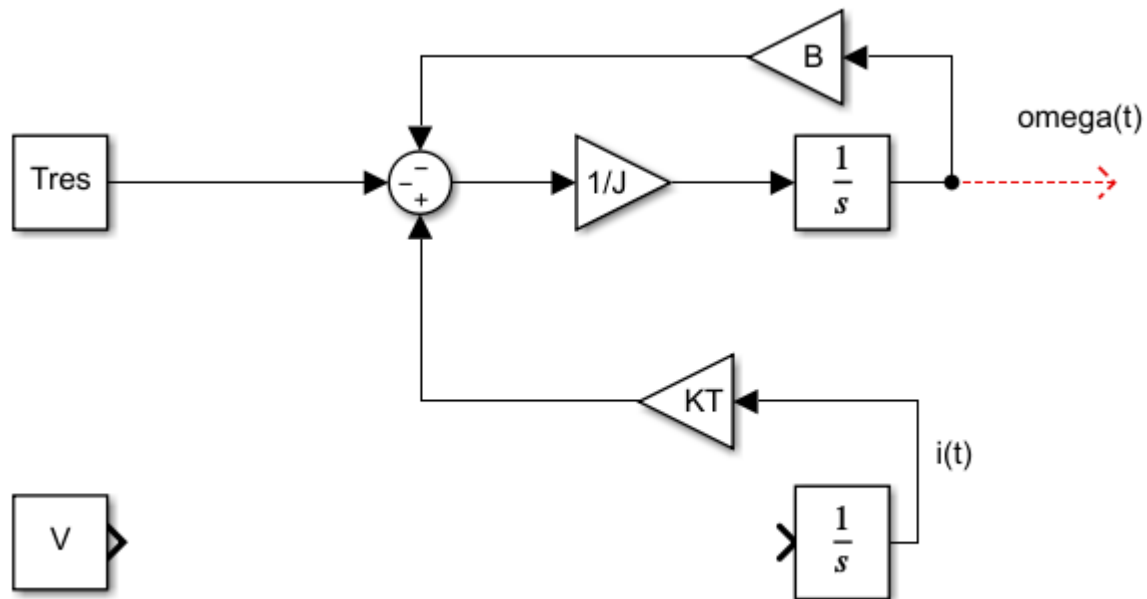
**File:** MotoreDC\_dati.m

Configuriamo i due blocchi integrator inserendo la condizione iniziale del segnale prodotto in uscita. All'interno delle rispettive finestre di parametrizzazione, inserire nel campo «Initial condition» le variabili  $\omega_0$  ed  $i_0$

Si desidera realizzare un test di funzionamento a vuoto (cioè con  $T_{res}(t) = 0$ ) con l'applicazione di una tensione di alimentazione  $V(t)$  costante.

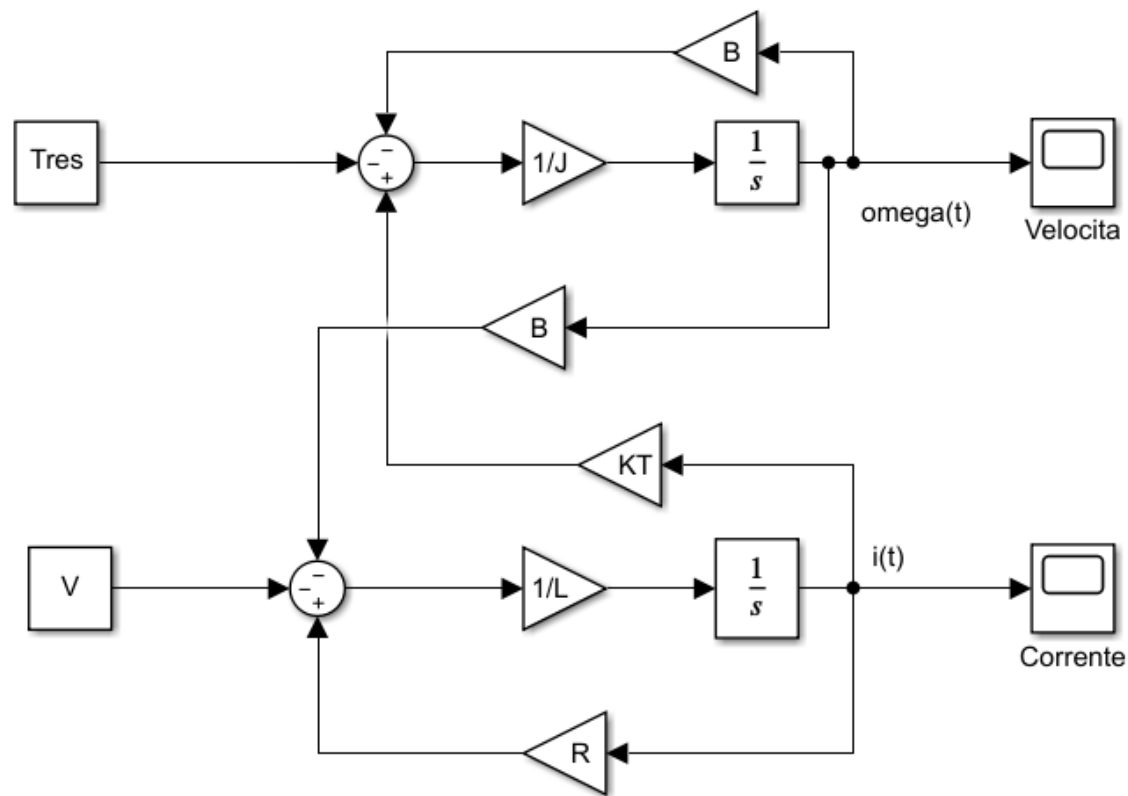
Inseriamo nella pagina di lavoro due blocchi `Constant` che impiegheremo per generare la tensione  $V(t)$  applicata al motore e la coppia resistente. Tali blocchi saranno parametrizzati in termini di ampiezza del segnale generato («Constant value») dalle variabili  $V$  e  $T_{res}$  che sono state definite nello script.

Modello della equazione meccanica  $\frac{d\omega(t)}{dt} = \frac{1}{J} [K_T i(t) - B\omega(t) - T_{res}(t)]$



Modello comprensivo anche della equazione elettrica

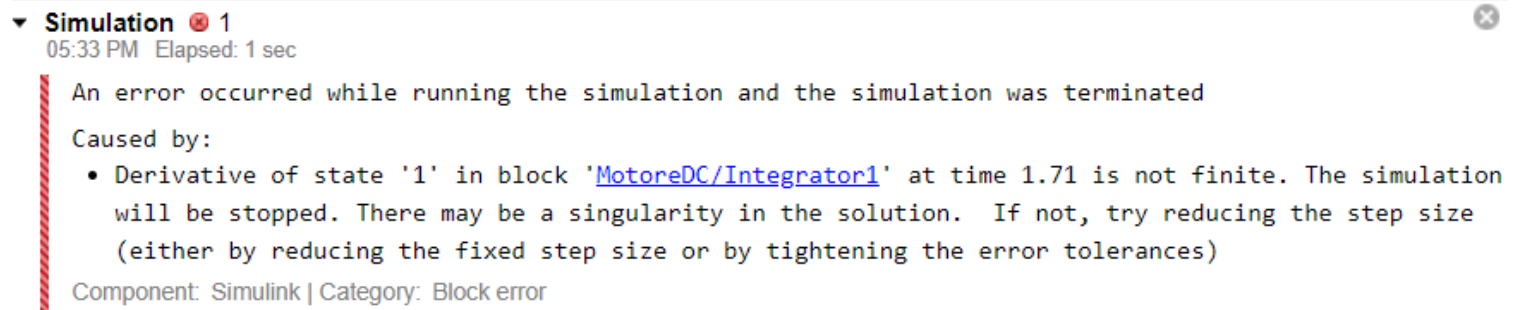
$$\frac{di(t)}{dt} = \frac{1}{L} [V(t) - Ri(t) - K_V \omega(t)]$$



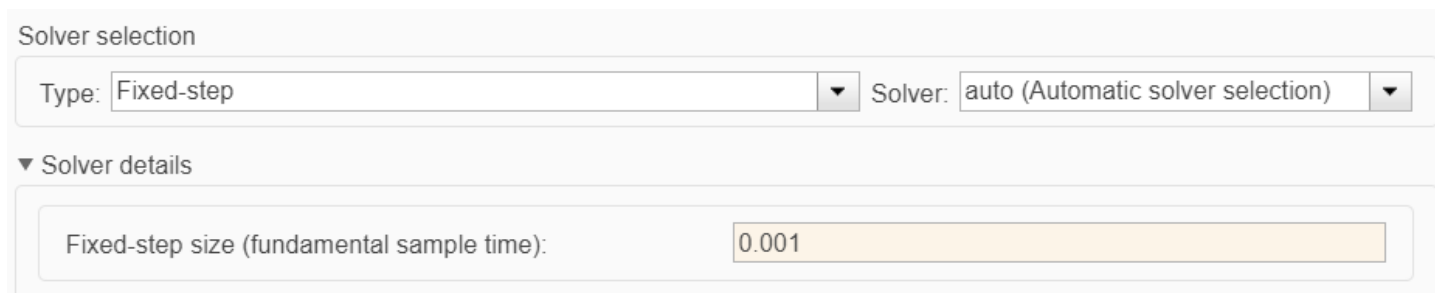
**File:** MotoreDC.slx

Si desidera realizzare un test di funzionamento a vuoto (cioè con  $T_{res}(t) = 0$ ) con l'applicazione di una tensione di alimentazione  $V(t)$  costante di ampiezza pari a  $10\text{ V}$ .

Se si avvia la simulazione utilizzando come fixed step size del solutore numerico il valore di 0.01 (suggerito in precedenza) si ottiene il seguente messaggio di errore

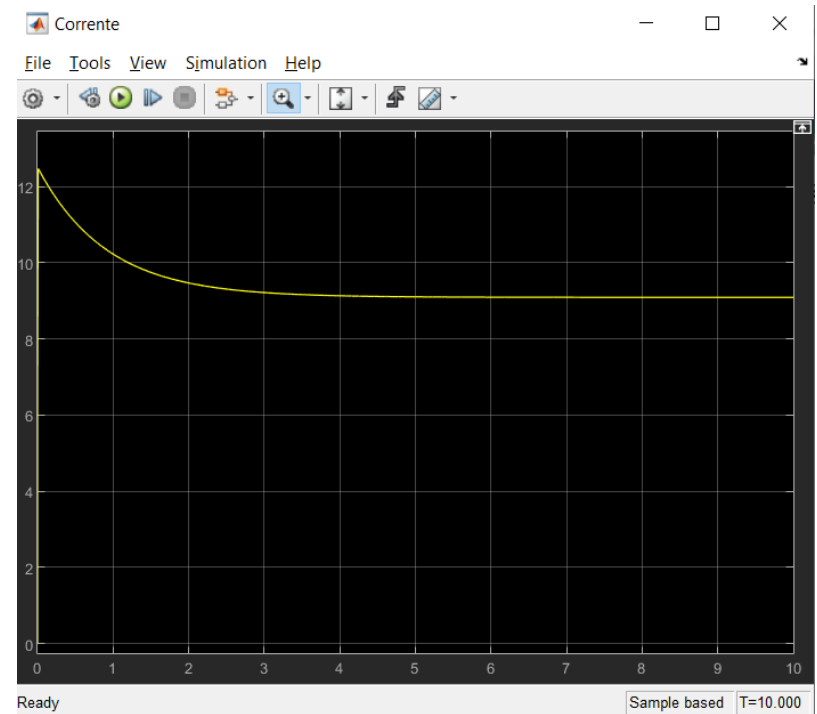
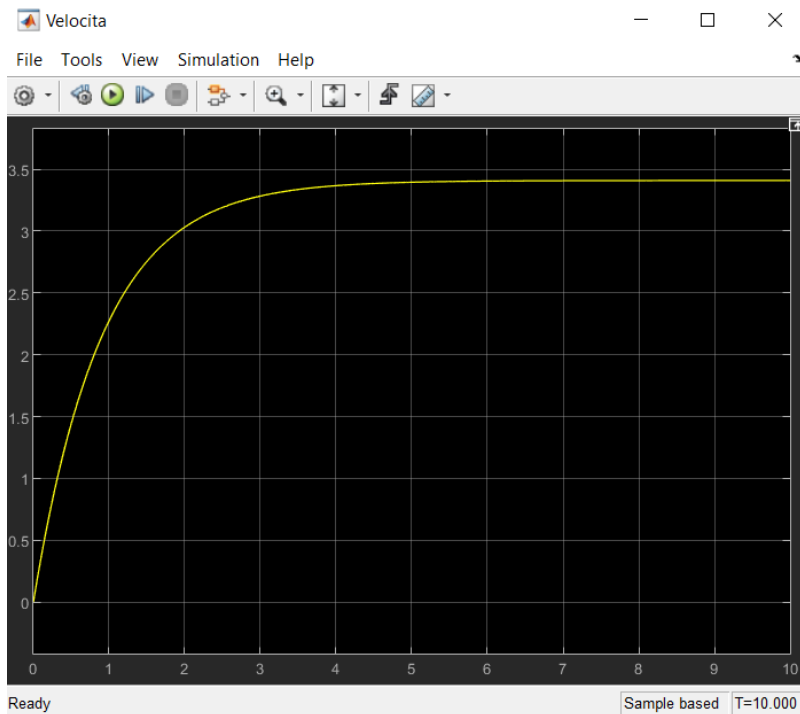


Pur nella sua semplicità, la simulazione numerica di questo modello matematico è resa problematica dal fatto che coesistono nel modello variabili di natura elettrica e di natura meccanica, aventi dinamiche molto differenti. Il solutore numerico con passo 0.01 diverge, e per risolvere il problema bisogna ridurlo fino a 0.001.



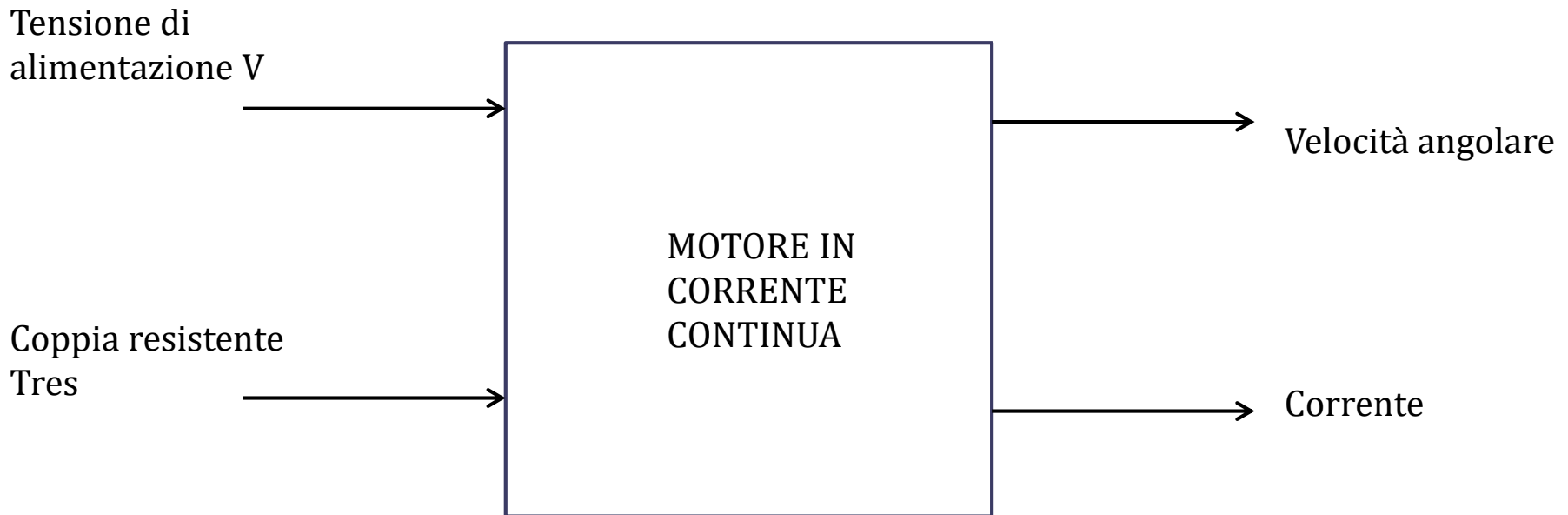


Ora possiamo avviare la simulazione e visualizzare i grafici della velocità angolare e della corrente di fase



## Sottosistemi (Subsystem)

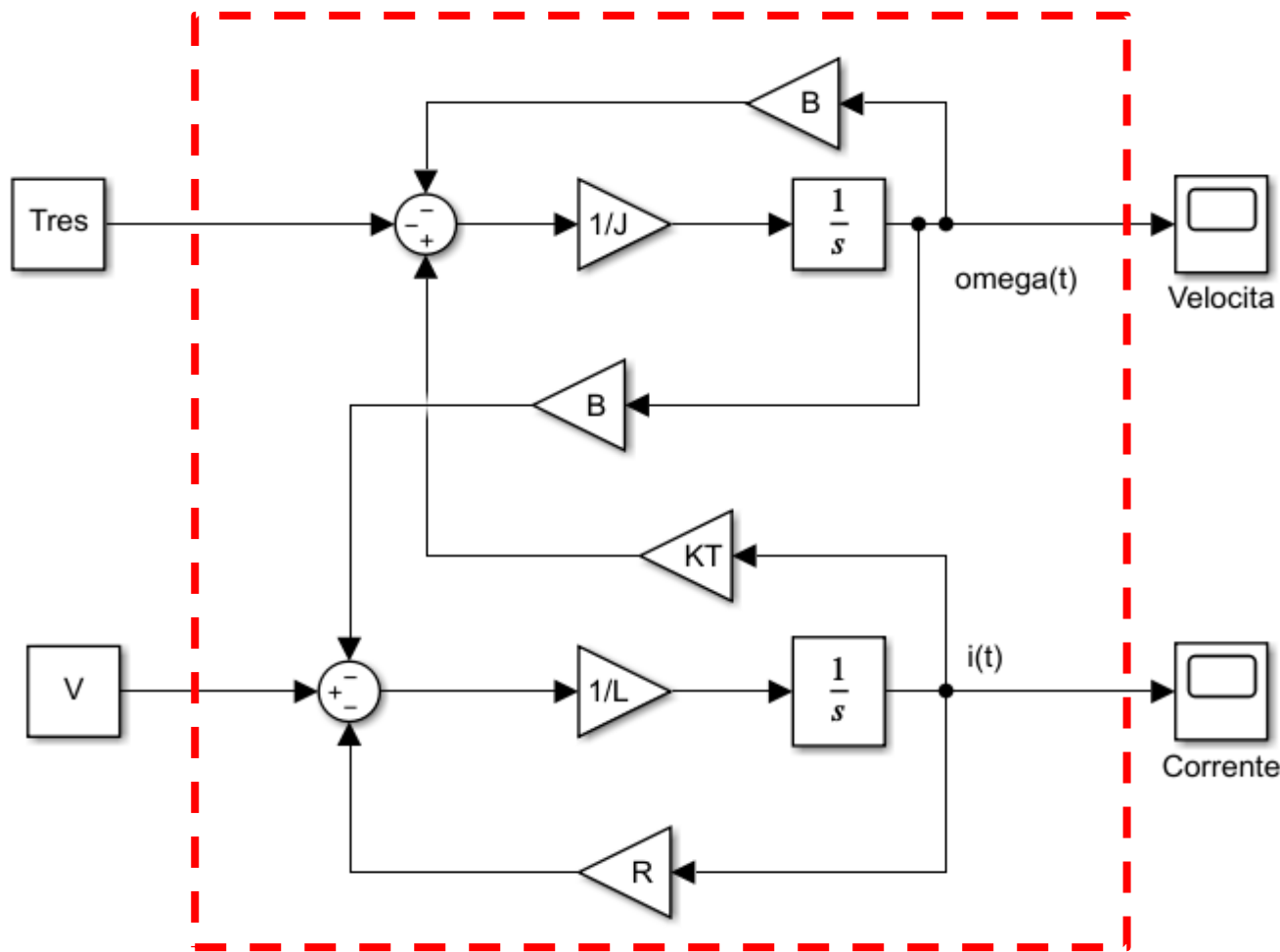
Ora vogliamo compattare il modello simulink del motore DC all'interno di un sottosistema compatto con ingressi e uscite, come mostrato nella figura seguente



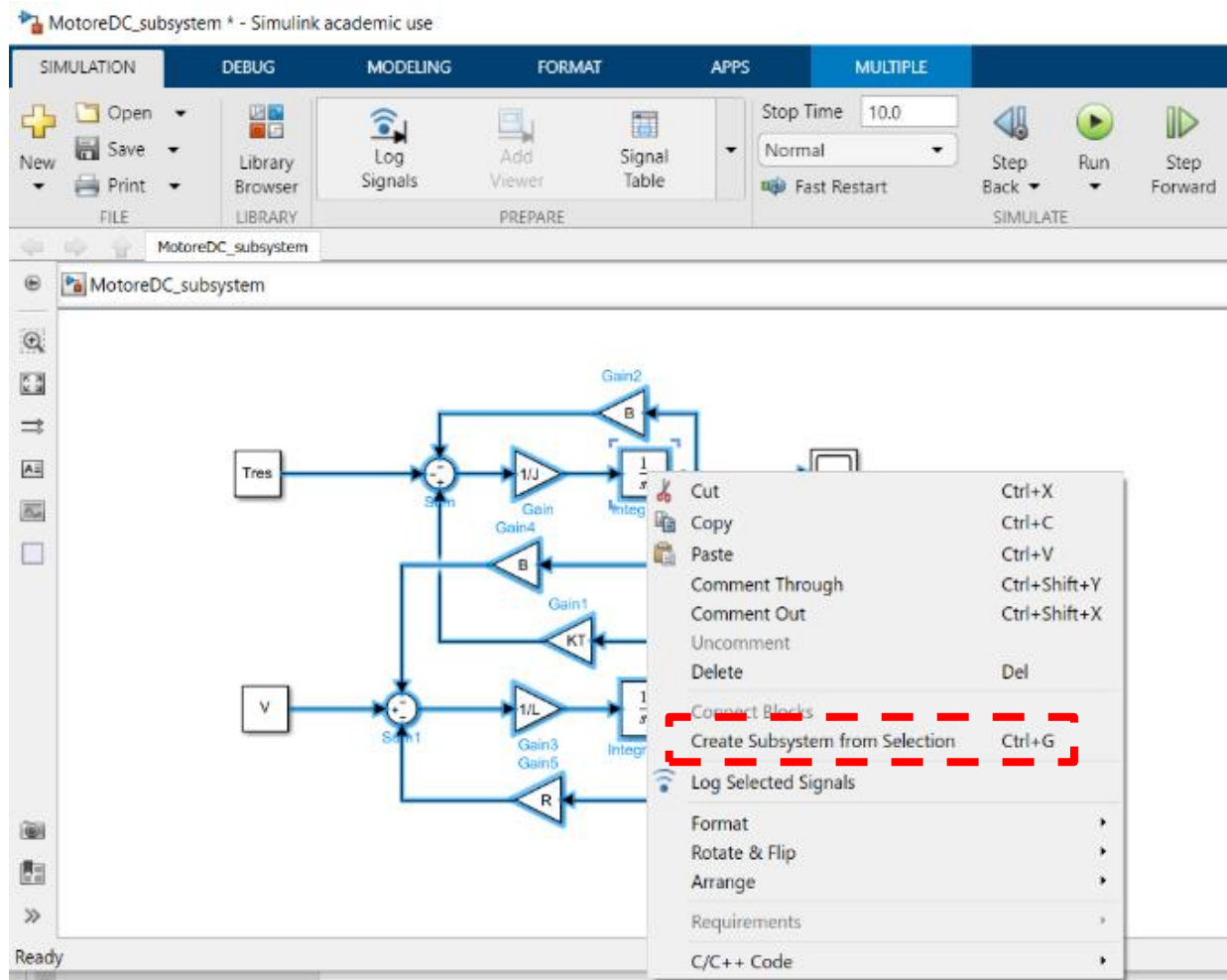
**Segnali di ingresso**

**Segnali di uscita**

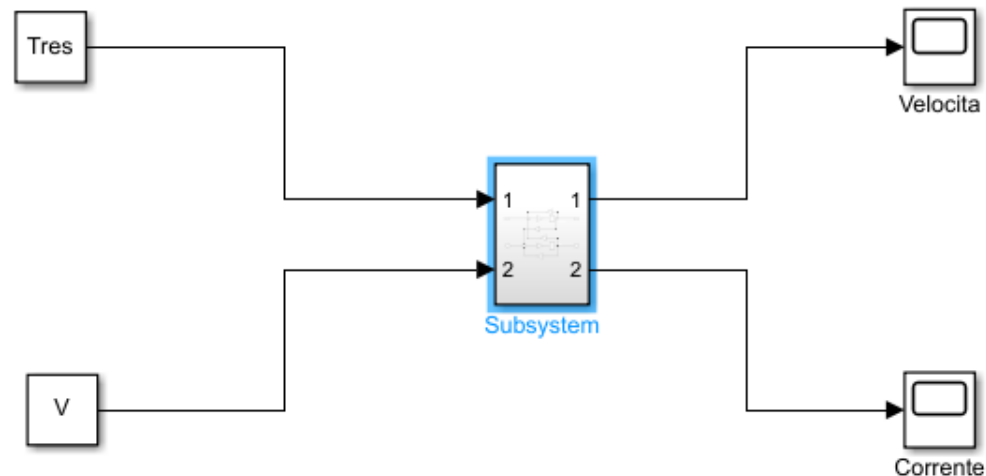
Bisogna tracciare sopra il modello un rettangolo come quello in figura, nel quale «entrino» i segnali di ingresso e dal quale «escano» i segnali di uscita del sottosistema che si sta creando. Ciò può richiedere, in taluni casi, di «ricollocare» i blocchi del modello spostandoli in modo che sia possibile tracciare il rettangolo citato.



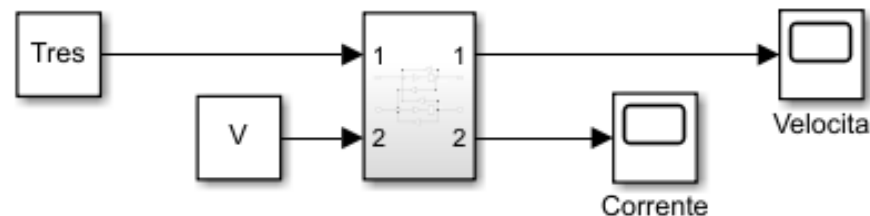
Individuare con il mouse un'area analoga a quella individuata dal rettangolo rosso della slide precedente, successivamente puntare il mouse su un qualunque blocco selezionato interno all'area, premere il tasto destro e selezionare "Create Subsystem from Selection" (in alternativa premere Ctrl+G)



Il risultato è il seguente

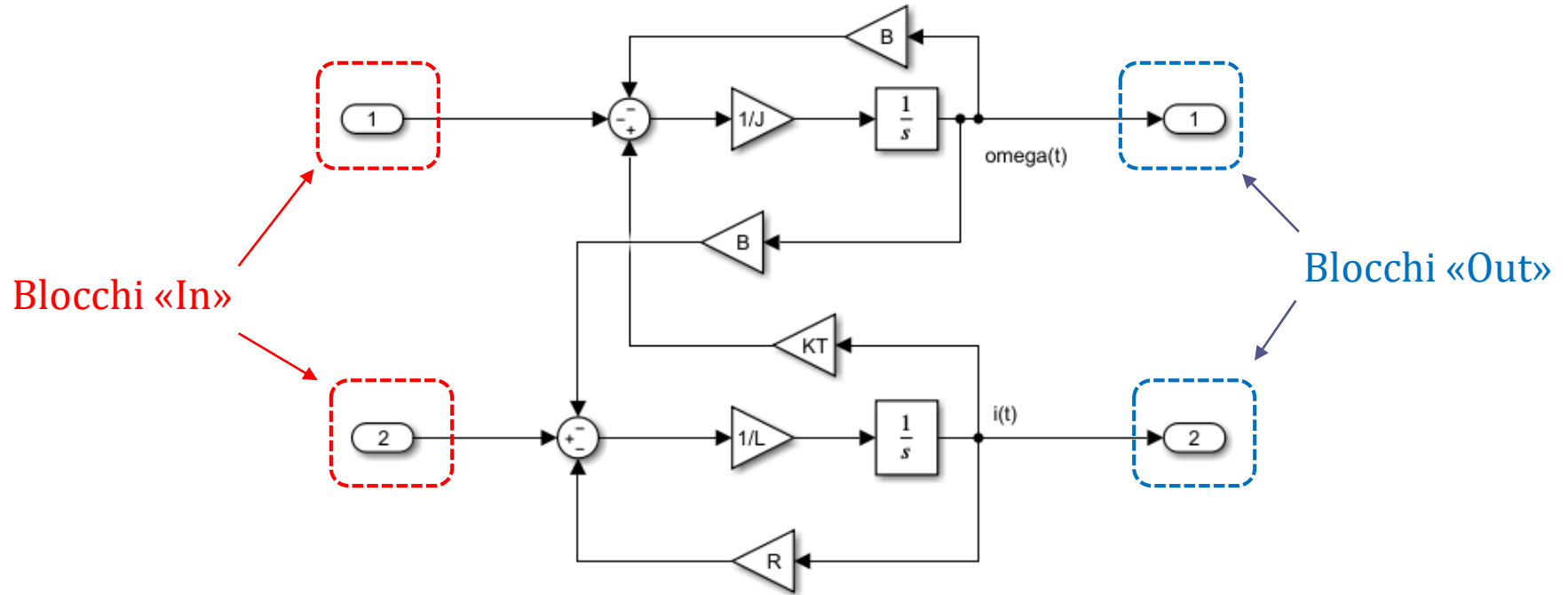


Possiamo riordinare e ridimensionare i blocchi per creare una struttura più ordinata



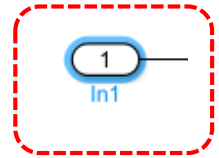
Ora vogliamo che nei terminali di ingresso e uscita del subsystem compaiano dei nomi rappresentativi dei segnali che entrano ed escono dal blocco.

Per accedere al contenuto del Subsystem bisogna fare doppio click su di esso

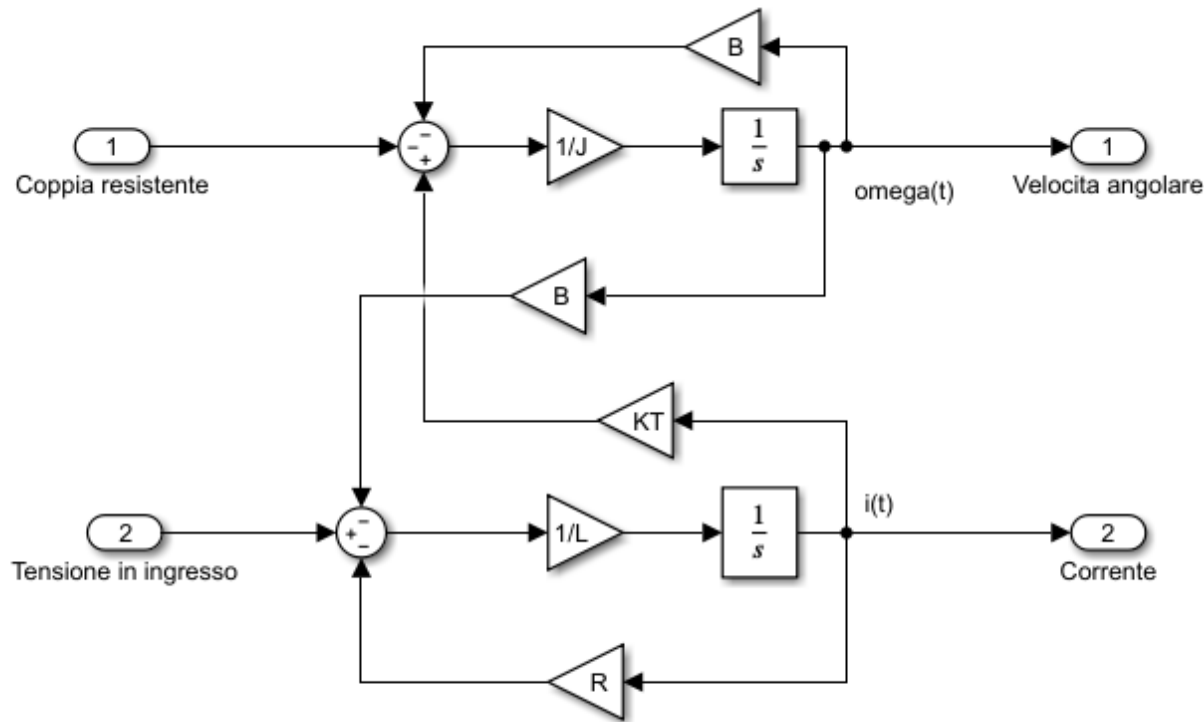


All'interno di un subsystem ciascun segnale di ingresso è rappresentato da un blocco In, mentre ciascun segnale di uscita è collegato ad un blocco Out. Se si aggiungono, o rimuovono, dei blocchi In o Out varierà in maniera corrispondente il numero di terminali di ingresso e uscita del subsystem.

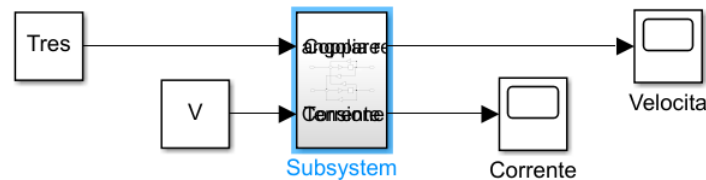
Per assegnare ad un terminale di ingresso o uscita una label si deve selezionare il corrispondente blocco In e Out, e si deve sovrascrivere il nome del blocchetto che compare sotto lo stesso.



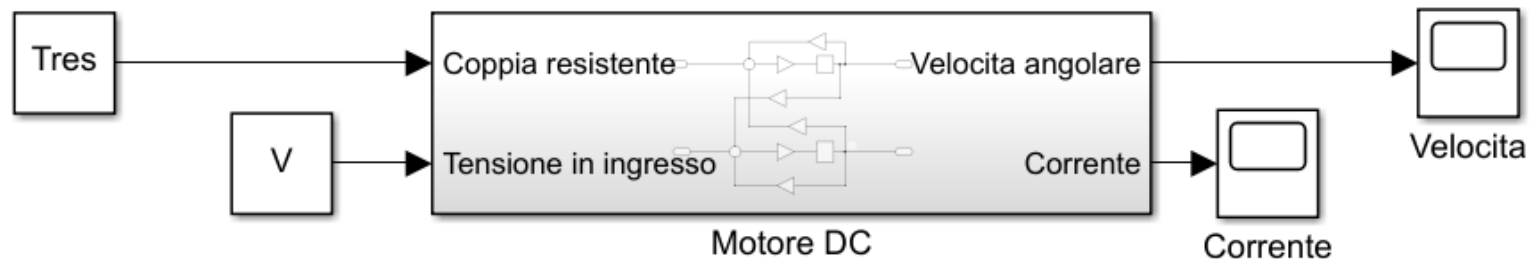
Svolgendo questa operazione per tutti i blocchi In e Out si giunge al seguente schema



L'aspetto del sottosistema risulta modificato.



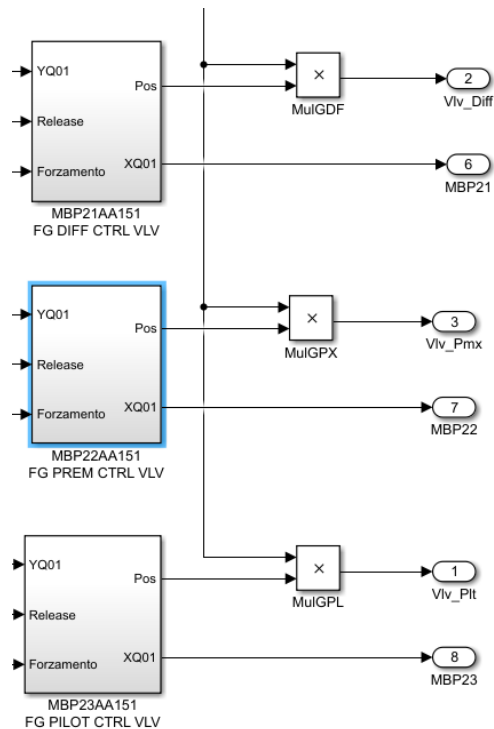
Allargando il sottosistema si visualizzano correttamente le labels. E' possibile attribuire un nome al Subsystem selezionandolo e sovrascrivendo il nome del blocchetto che compare sotto lo stesso.



In trasparenza è mostrato il contenuto del sottosistema. Per disabilitare tale visualizzazione cliccare con il tasto destro sul sottosistema e successivamente disselezionare la voce «Content Preview» dal menù «Format».



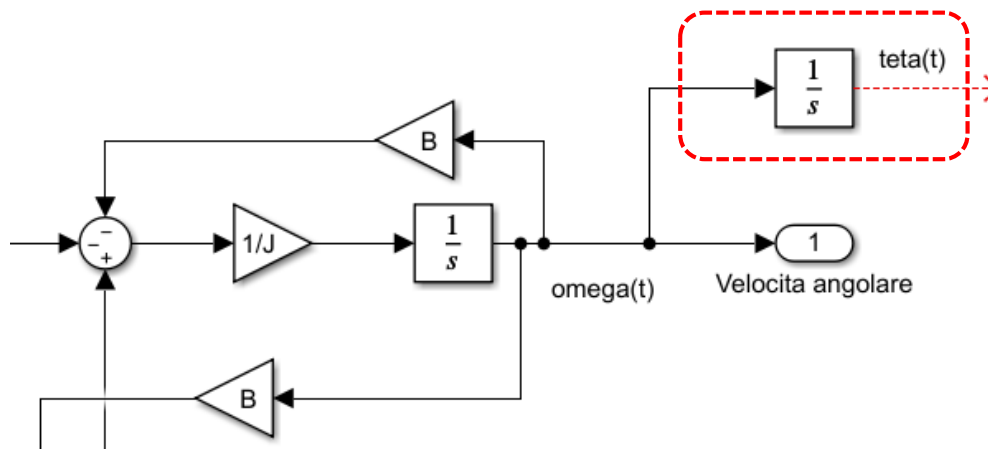
Il sottosistema può ora essere duplicato all'interno del modello o importato all'interno di altri modelli



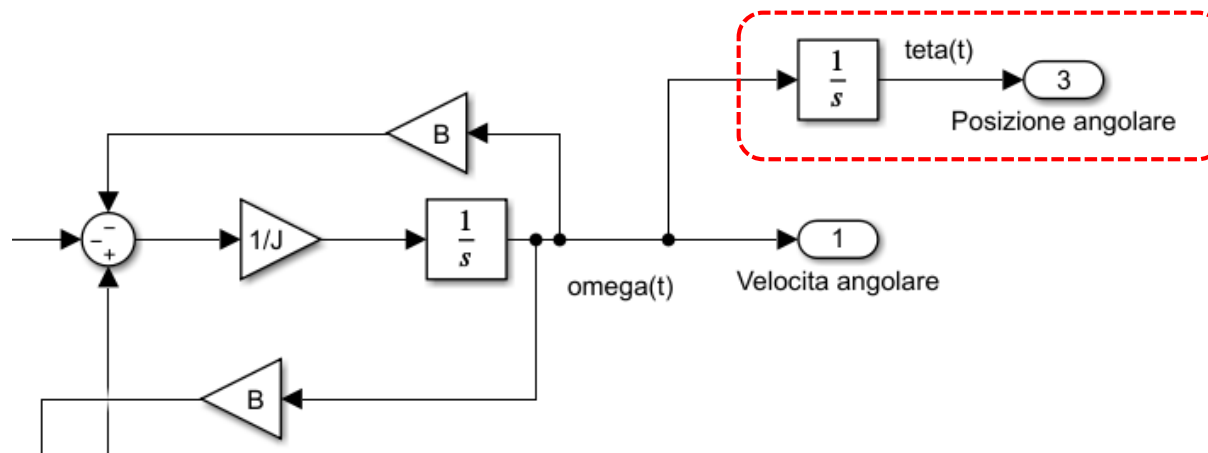
Porzione di un modello di simulazione di un impianto, con tre sottosistemi identici in parallelo che rappresentano 3 valvole motorizzate con le stesse caratteristiche.

Modifichiamo il subsystem in modo che venga restituita in uscita anche la **posizione angolare** del motore.

Dobbiamo preliminarmente rendere accessibile la posizione angolare integrando la velocità (si inserisca quindi un nuovo blocco integratore che riceve in ingresso la velocità)



Ora importiamo nello schema dalla libreria «Commonly Used Blocks» una istanza del blocco elementare «Out», e colleghiamone il segnale della posizione angolare al terminale di ingresso del blocco Out in modo che tale segnale venga mandato all'esterno del sottosistema attraverso una nuova porta di uscita. Associamo anche una Label al blocco Out in modo che tale label sia inserita nella porta di uscita



Ecco come si presenta il blocco Subsystem dopo avere apportato tale modifica:

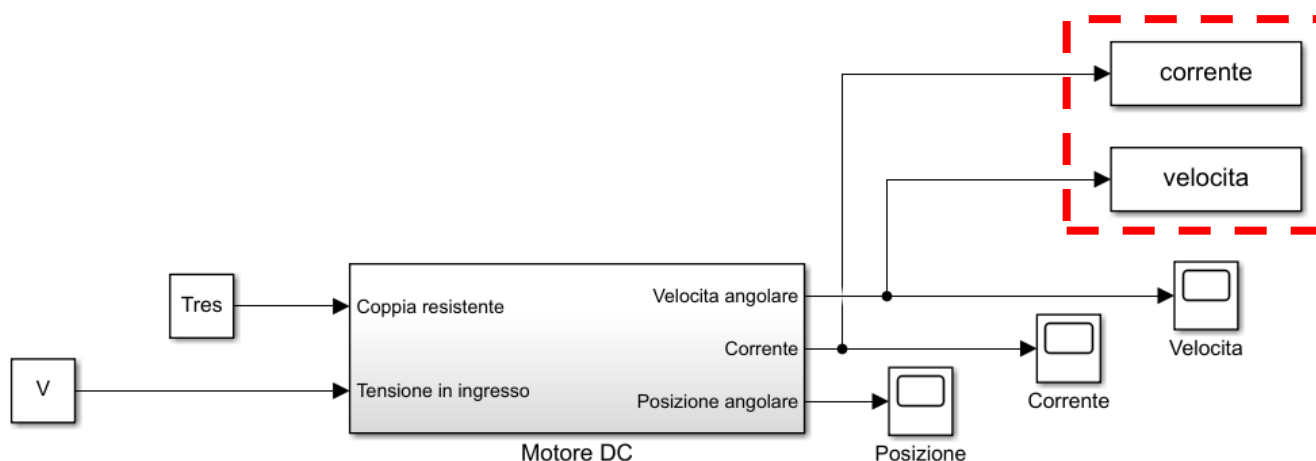


**File:** MotoreDC\_subsystemPosizione.slx

## Esportazione verso il workspace di Matlab di dati generati da un modello Simulink

Mostriamo come esportare dati generati da un modello Simulink verso il workspace di Matlab. Ciò ne consente l'elaborazione numerica o, ad esempio, la creazione con il comando **plot** di grafici di qualità corredati da etichette esplicative

L'esportazione di dati da Simulink verso il workspace di Matlab avviene mediante il blocco elementare "To workspace" che si trova nella libreria Sinks. Importiamone 2 istanze onde esportare nel workspace i dati associati alla velocità angolare ed alla corrente di fase.



Blocchi  
"To workspace"  
collegati ai segnali  
da esportare nel  
workspace

**File:** MotoreDC\_subsystem2Workspace.slx

## Finestra di parametrizzazione del primo blocco “To Workspace”

Nome che sarà attribuito alla variabile esportata nel workspace

Possibilità di sottocampionare i dati esportati nel workspace attraverso il parametro «Decimation»

Formato di esportazione: modalita di default “**Timeseries**”

Block Parameters: To Workspace

To Workspace

Write input to specified timeseries, array, or structure in a workspace. For menu-based simulation, data is written in the MATLAB base workspace. Data is not available until the simulation is stopped or paused.

To log a bus signal, use "Timeseries" save format.

Parameters

Variable name:

corrente

Limit data points to last:

inf

Decimation:

1

Save format: Timeseries

☒ Log fixed-point data as a TI object

Sample time (-1 for inherited):

-1

OK Cancel Help Apply

Eseguire la simulazione, e verificare quali nuovi variabili compaiano nel workspace dopo che la simulazione è terminata

Al termine della simulazione sono presenti nel workspace le due variabili «corrente» e «velocita» aventi come formato «Timeseries»

Il formato «Timeseries» contiene al proprio interno per ciascun segnale sia gli istanti di campionamento che i valori del segnale.

Variables - velocita

velocita

1x1 double timeseries

Time series name:

Time	Data:1
0	0
1.0000e-03	0.0011
0.0020	0.0037
0.0030	0.0069
0.0040	0.0105
0.0050	0.0141
0.0060	0.0178
0.0070	0.0215
0.0080	0.0252
0.0090	0.0289

Workspace

Name	Value	Size
B	0.8000	1x1
corrente	1x1 double timeseries	1x1
i0	0	1x1
J	1	1x1
KT	0.3000	1x1
KV	0.3000	1x1
L	1.0000e-03	1x1
omega0	0	1x1
R	0.8000	1x1
tout	10001x1 double	10001x1
Tres	0	1x1
V	10	1x1
velocita	1x1 double timeseries	1x1

In una variabile di tipo “Timeseries” avente nome generico “var” si accede ai vettori degli istanti di campionamento ed al vettore dei valori del segnale con le sintassi:

**var.Time**  
**var.Data**

Potremo quindi accedere ai dati di velocità e corrente con le sintassi

`corrente.Time`  
`corrente.Data`

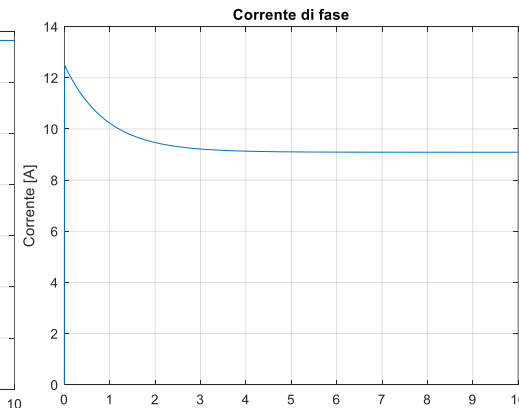
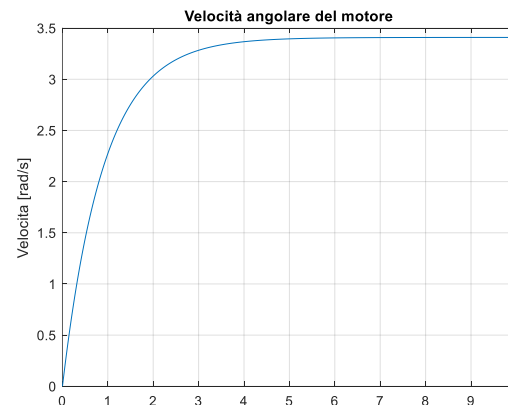
`velocita.Time`  
`velocita.Data`

Si noti che tali vettori sono esattamente gli argomenti da passare alla funzione plot per costruire il relativo grafico.

**Dopo avere eseguito la simulazione**, si può eseguire il seguente script di creazione dei grafici

```
figure(1)
plot(velocita.Time,velocita.Data),
grid,
title('Velocità angolare del motore')
ylabel('Velocita [rad/s]')
```

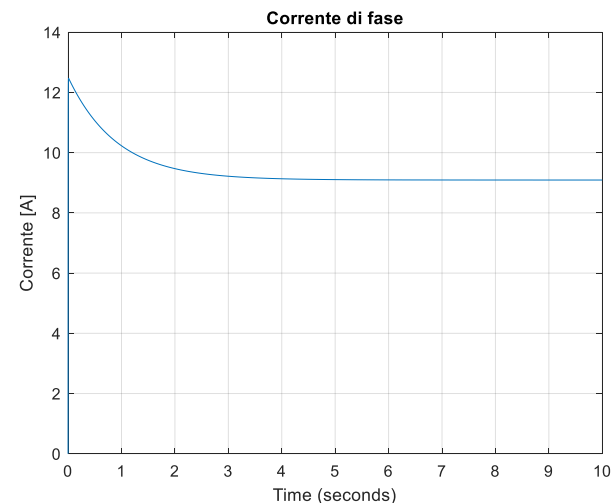
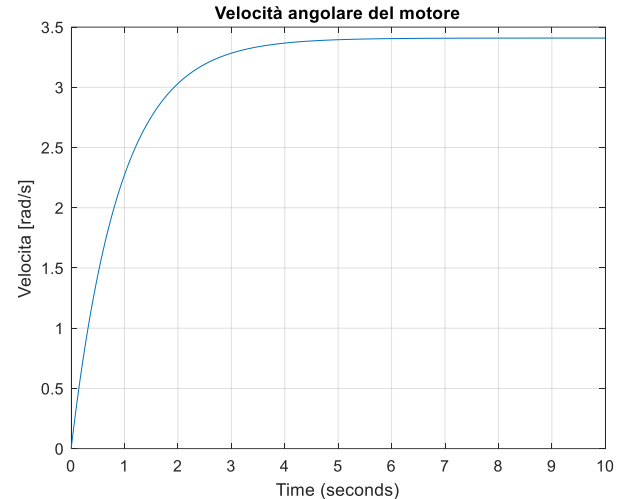
```
figure(2)
plot(corrente.Time,corrente.Data),
grid,
title('Corrente di fase')
ylabel('Corrente [A]')
```



Per la creazione di **grafici singoli** a partire da variabili di tipo Timeseries, si può utilizzare una sintassi semplificata in cui viene passata alla funzione plot la sola variabile di tipo Timeseries.

```
figure(1)
plot(velocita),
grid,
title('Velocità angolare del motore')
ylabel('Velocita [rad/s]')
```

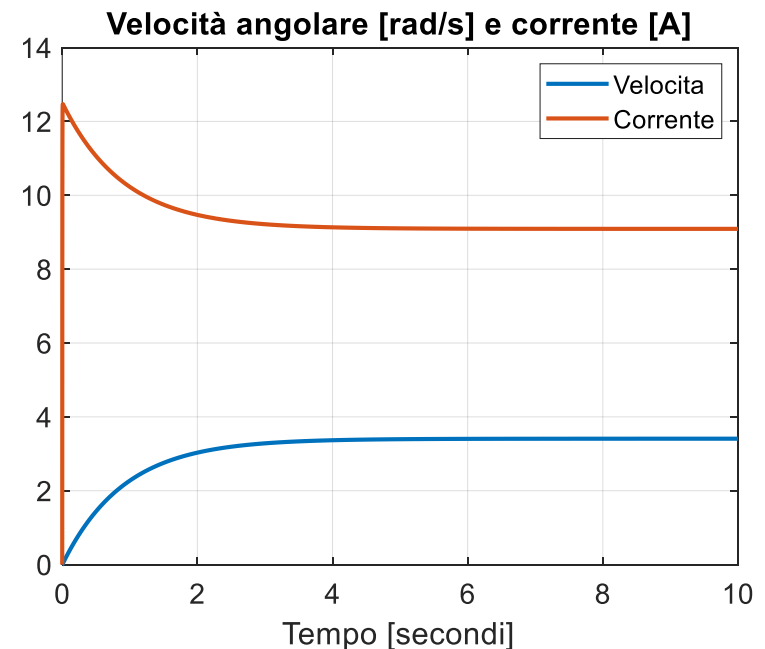
```
figure(2)
plot(corrente),
grid,
title('Corrente di fase')
ylabel('Corrente [A]')
```





Per la creazione di **grafici con curve sovrapposte** a partire da variabili di tipo Timeseries, si deve invece necessariamente accedere ai vettori con la sintassi .Time e .Data. Il seguente codice crea un grafico in cui la velocità e la corrente risultano sovrapposte

```
figure(3)
plot(velocita.Time,velocita.Data,corrente.Time,corrente.Data,'LineWidth',2),
grid,
title('Velocità angolare [rad/s] e corrente [A]','FontSize',14)
xlabel('Tempo [secondi]','FontSize',14)
set(gca,'FontSize',14)
legend('Velocita','Corrente')
```



## Gestione di modelli Simulink mediante script (funzione `sim`)

Vogliamo imparare ad **aprire** ed **eseguire** i file Simulink per mezzo di uno script Matlab che impartisca i relativi comandi. Realizziamo uno script all'interno del quale si operi la parametrizzazione del modello, l'avvio della simulazione e la successiva creazione di grafici.

Sviluppiamo questo esempio con riferimento al file  
`MotoreDC_subsystem2Workspace.slx`

Impiegheremo all'interno di tale Script le funzioni Matlab **`open_system`** e **`sim`** che servono rispettivamente per aprire il file simulink e per eseguirlo.

```
%% Parametrizzazione del modello
```

**Inizio di una nuova «Section»**

```
clear all,clc
```

```
% Parametri modello motore DC
```

```
% Parametri elettromeccanici
```

```
R=0.8;      % resistenza
```

```
L=1e-3;     % induttanza
```

```
KT=0.3;     % costante di coppia
```

```
KV=0.3;     % costante di forza c.e.m.
```

```
J=1;        % momento di inerzia
```

```
B=0.8;      % coefficiente di attrito viscoso
```

```
% Ingressi
```

```
V=10;       % tensione di alimentazione
```

```
Tres=0;     % coppia resistente
```

```
% Condizioni iniziali
```

```
omega0=0;   %condizione iniziale della velocita
```

```
i0=0;       %condizione iniziale della corrente
```

```
%% Apertura ed avvio del modello
```

**Inizio di una nuova «Section»**

```
open_system('MotoreDC_subsystem2Workspace')
```

**Apri il modello Simulink**

```
sim('MotoreDC_subsystem2Workspace')
```

**Esegui il modello**

**(continua)**

**(continua)**

`%% Creazione grafici`

**Inizio di una nuova «Section»**

```
figure(1)
plot(velocita),
grid,
title('Velocità angolare del motore')
ylabel('Velocita [rad/s]')
```

```
figure(2)
plot(corrente),
grid,
title('Corrente di fase')
ylabel('Corrente [A]')
```

```
figure(3)
plot(velocita.Time,velocita.Data,corrente.Time,corrente.Data,'LineWidth',2),grid,
title('Velocità angolare [rad/s] e corrente [A]','FontSize',14)
xlabel('Tempo [secondi]','FontSize',14)
set(gca,'FontSize',14)
legend('Velocita','Corrente')
```

**Files:** MotoreDC\_ScriptGestione.m

Lo script riportato nelle due slides precedenti implementa la completa gestione del modello Simulink senza che l'utente debba operare direttamente su di esso.

Lo script salva dapprima nel workspace le variabili simboliche utilizzate dal modello Simulink, quindi apre ed esegue il modello, ed in ultimo crea dei grafici utilizzando le variabili di tipo Timeseries che vengono esportate nel workspace dal modello Simulink al termine della esecuzione.

Si noti come nello Script sono state inserite alcune righe di codice che hanno **due caratteri percentuali consecutivi a inizio riga**.

Tali righe di codice inducono la suddivisione dello script in varie **Sezioni** (Sections)

```
%% Parametrizzazione del modello
```

Prima section

```
%% Apertura ed avvio del modello
```

Seconda section

```
%% Creazione grafici
```

Terza section

Lo script contiene tre  
distinte sezioni

L'utilità delle sections è duplice. Non solo individua e separa parti dello Script in cui vengono eseguite operazioni di natura differente ma consente facilmente di eseguire la porzione di codice contenuta in una sezione senza dover necessariamente eseguire tutto lo script.

Per fare ciò si deve rendere «attiva» una particolare sezione cliccando su di essa (la section attiva è evidenziata con un colore particolare, come mostrato nella figura sottostante) e successivamente premere Ctrl+Invio. In tal modo si eseguono unicamente le istruzioni Matlab contenute nella sezione attiva.

```
omega0=0; %condizione iniziale della velocita
i0=0;      %condizione iniziale della corrente
%% Apertura ed avvio del modello
open_system('MotoreDC_subsystem2Workspace')
sim('MotoreDC_subsystem2Workspace')
%% Creazione grafici
figure(1)
plot(velocita),
grid,
title('Velocità angolare del motore')
ylabel('Velocita [rad/s]')
```

Ad esempio, una volta che il modello sia stato aperto ed eseguito (operazione che può richiedere un tempo anche molto lungo in funzione della complessità del modello) si andrà unicamente a lavorare sulla sezione di creazione dei grafici, modificandola e rieseguendola in maniera selettiva finché non si sia ottenuto l'output desiderato.

## Esercizio 1

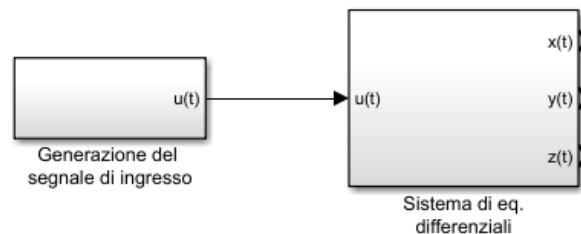
Si consideri il sistema di equazioni differenziali

$$\dot{x}(t) + 3x(t) - 2|y(t)| = 0$$

$$\ddot{y}(t) + y(t) + |x(t)| = z(t)$$

$$\dot{z}(t) + 2z^3(t) - |y(t)| = u(t) \quad u(t) = t - 1$$

Interpretando il segnale  $u(t) = t - 1$  come un ingresso esterno applicato al sistema di equazioni differenziali, si costruisca un modello Simulink del sistema che contenga due Subsystems come mostrato nella figura seguente



Dopo avere realizzato il modello, scrivere uno script che lo esegua in automatico e crei un grafico che riporti sovrapposti i segnali  $y(t)$  e  $\dot{y}(t)$  nell'intervallo temporale  $t \in [0,20]$  a partire dalle condizioni iniziali  $x(0) = y(0) = 1$ ,  $\dot{y}(0) = -1$ ,  $z(0) = 2$ .

## Traccia della soluzione

Sistema in forma esplicita

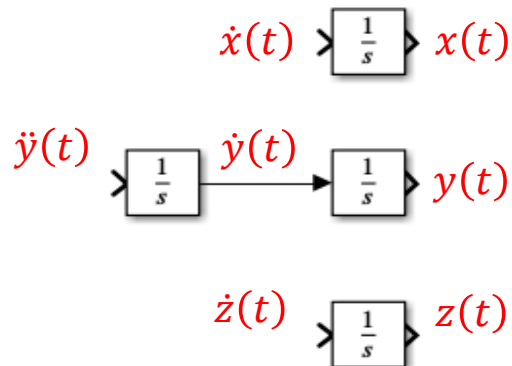
$$\dot{x}(t) = -3x(t) + 2|y(t)|$$

$$\ddot{y}(t) = -y(t) - |x(t)| + z(t)$$

$$\dot{z}(t) = -2z^3(t) + |y(t)| + u(t)$$

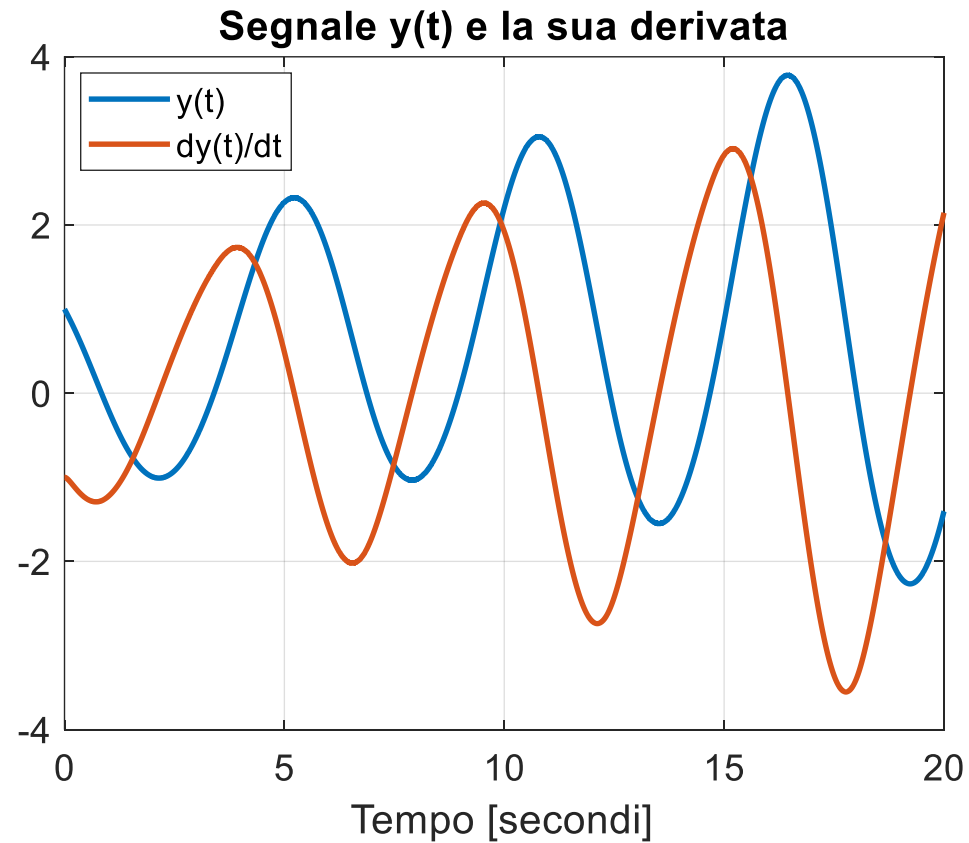
Sistema composto da 3 eq. differenziali accoppiate: una equazione del secondo ordine e due equazioni del primo ordine (ordine complessivo del sistema = 4)

Importiamo nella pagina di lavoro **4 integratori**, e li disponiamo su **3 righe distinte**, ciascuna associata ad una delle tre equazioni differenziali. In ciascuna riga andremo a collocare un numero di integratori pari all'ordine della corrispondente equazione, disponendo fra loro in serie gli integratori delle righe associate ad una equazione di ordine maggiore di uno.



Ora si completi lo schema, in analogia con la procedura seguita nei precedenti esempi, costruendo gli ingressi ai vari integratori in accordo con le varie espressioni delle 3 equazioni differenziali in forma esplicita



**Files:**

Esercizio01.slx

Esercizio01\_script.m