# *Laboratorio 3 per fondamenti di automatica - ing. fisica*

## Esercizio 1: Legge di controllo e ricostruzione dello stato; progetto del regolatore

Innanzitutto occorre definire il sistema in spazio di stato nel workspace di matlab (vedi esercitazione 1); pertanto i comandi da digitare nel prompt sono i seguenti:

```
A=[2 1 -1; -3 -2 -1; -1 1 -4];
b=[1 0 1]';
c=[0 0 1];
d=0;                          % definizione matrici del sistema
```

a) Valutando gli autovalori della matrice A

```
autoval=eig(A)
autoval =
    1.5616
   -2.5616
   -3.0000
```

si può notare che hanno modulo superiore a 1 così che il sistema risulta instabile.

b) Affinché il sistema sia stabilizzabile mediante un regolatore lineare composto da legge di controllo *k* e ricostruttore dello stato *l*, occorre che questo sia completamente raggiungibile e completamente osservabile. I comandi matlab `ctrb()` e `obsv()` restituiscono, rispettivamente, le matrici di raggiungibilità e di osservabilità.

```
R=ctrb(A,b);                          % matrice di raggiungibilità
det_R=det(R)
det_R =
    12                                % completa raggiungibilità
O=obsv(A,c);                          % matrice di osservabilità
det_O=det(O)
det_O =
    8                                 % completa osservabilità
```

Poiché le matrici di raggiungibilità e di osservabilità hanno determinante non nullo, il sistema è, rispettivamente, completamente raggiungibile e osservabile. Pertanto, il sistema è stabilizzabile mediante un regolatore lineare e la dinamica del sistema regolato è fissabile ad arbitrio. In particolare, sarà possibile determinare una legge di controllo *k* e un ricostruttore dello stato *l* tale che tutti i transitori di annullino in tempo finito.

c) Progetto della legge di controllo *k*.
Per progettare una retroazione dello stato che assegni gli autovalori come chiesti (tutti nulli) si può utilizzare la formula di Ackermann e il comando matlab `acker()`. (digitare `help acker` per una discussione dettagliata sull'uso del comando; in particolare, il comando fa riferimento a una legge di controllo $u = -kx$ così che, nell'uso del comando, il vettore *b* andrà sostituito con il vettore $-b$).

```
autovalori=[0 0 0];
```

```
k = acker(A,-b,autovalori)
k =
    2.5000    3.0000    1.5000
eig(A + b*k)                    % verifica autovalori
```

Progetto del ricostruttore dello stato *l*.

In modo analogo si può ottenere il ricostruttore dello stato; si noti però che la formula di Ackermann permette di posizionare i poli del sistema in anello chiuso ($A+bk$), mentre la dinamica dell'osservatore è descritta dalla matrice ($A+lc$) (si noti la posizione di *l*, duale rispetto a quella di *k* – il prodotto matriciale non gode della proprietà commutativa).

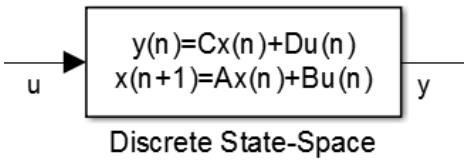
Pertanto, per risolvere il problema utilizzando la formula di Ackermann, basta osservare che $(A+lc)^T$ ha gli stessi autovalori di $(A+lc)$. Ma $(A+lc)^T = A^T + c^T l^T$

Pertanto basta applicare la formula di Ackermann alla coppia $(A^T, -c^T)$ per ottenere $l^T$.

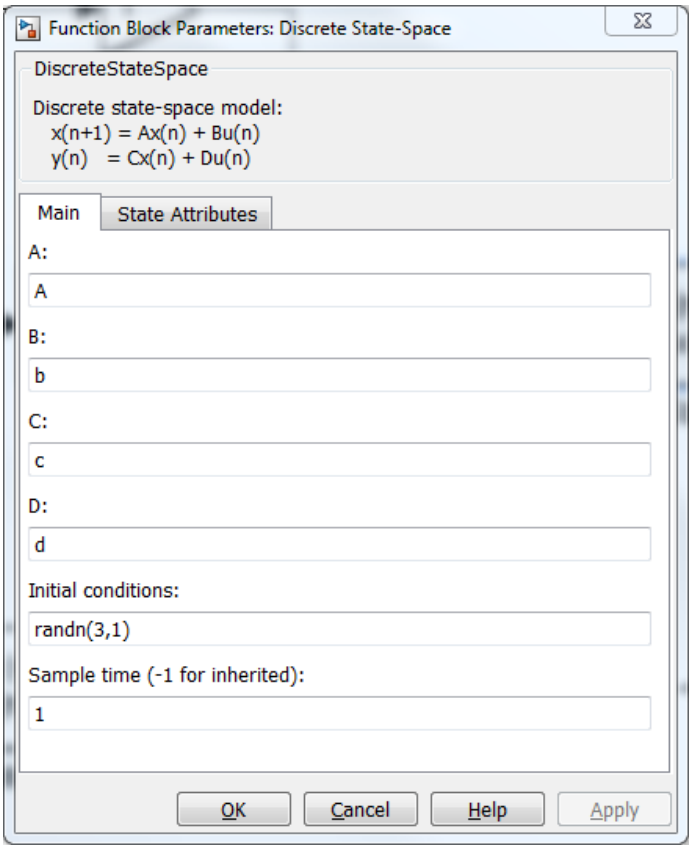
```
lT = acker(A',-c',autovalori);
l = lT'
l = 1.3750
    0.3750
    4.0000

eig(A + l*c)                    % verifica autovalori
```

d) Implementazione in simulink

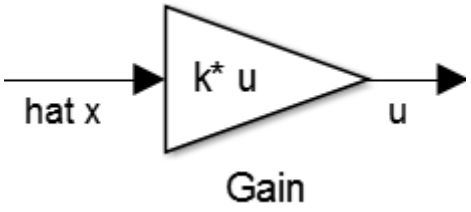
Il sistema può essere definiti tramite Simulink utilizzando il blocco Discrete State-Space che riceve in ingresso *u*, ingresso del sistema, e da cui esce *y*, uscita del sistema.

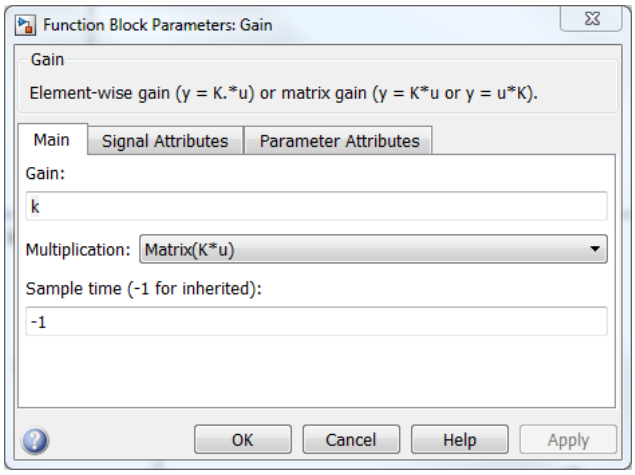



in cui la condizione iniziale è impostata come casuale (siccome noi non la conosciamo).

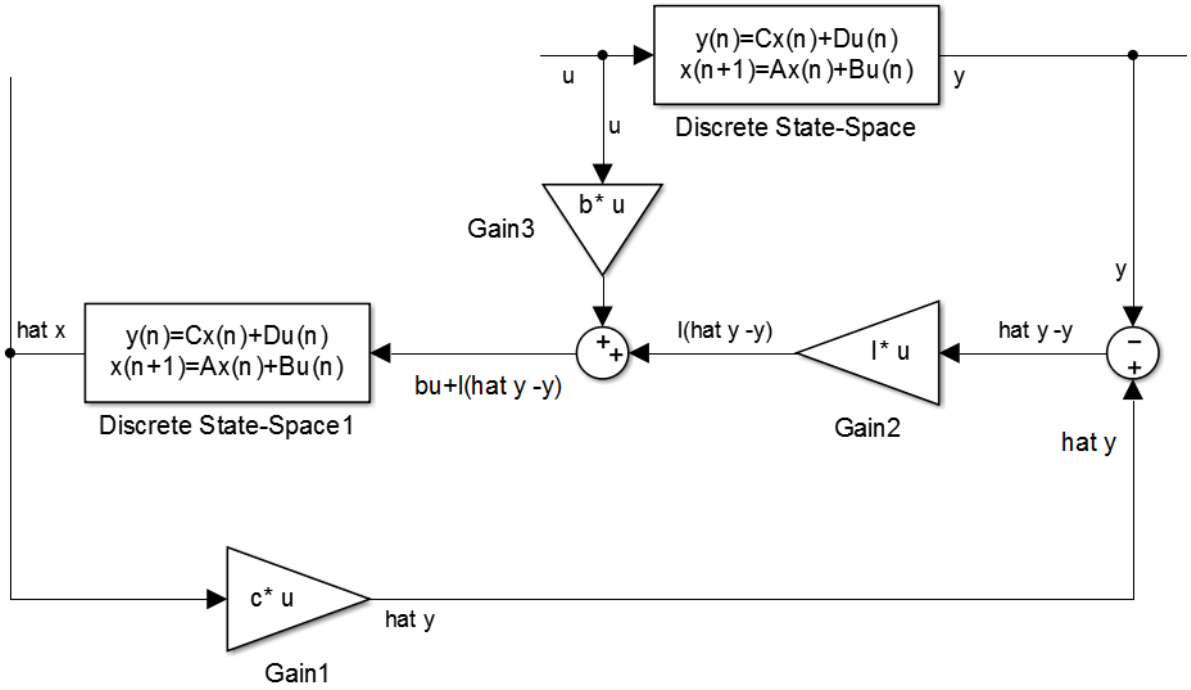


Il controllore viene implementato tramite un blocco di tipo guadagno, che riceve in ingresso lo stato ricostruito $\hat{x}$ e ha come uscita l'ingresso $u$ che vogliamo dare al sistema (si noti che il prodotto del blocco è impostato come prodotto matriciale).




Per l'implementazione del ricostruttore dobbiamo fare una copia del sistema e cambiare gli ingressi e le uscite. Esso avrà come uscita lo stato ricostruito (quindi la C del sistema sarà la matrice identità, in modo che $y = C\hat{x} = \hat{x}$, e come ingresso $bu + l(\hat{y} - y)$. Per far questo quindi possiamo costruire il segnale di ingresso all'esterno del blocco, e mettere come matrice di ingresso del nostro sistema ancora l'identità.




Unendo il blocco del controllore a quello del ricostruttore abbiamo progettato un sistema di controllo (senza ingressi) che porta il sistema a 0 in tempo finito. Volendolo portare ad un determinato valore di riferimento $y^0$ ci ricordiamo che nel progetto del regolatore imponiamo l'ingresso $u = k\hat{x} + v$: possiamo utilizzare la $v$ quindi per modulare l'uscita.

L'intero sistema regolato può essere scritto come

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = \begin{bmatrix} A & b\,k \\ -l\,c & A + bk + lc \end{bmatrix} \begin{bmatrix} x \\ \hat{x} \end{bmatrix} + \begin{bmatrix} b \\ b \end{bmatrix} v$$

$$y = \begin{bmatrix} c & 0 \end{bmatrix} \begin{bmatrix} x \\ \hat{x} \end{bmatrix} + d\,v$$

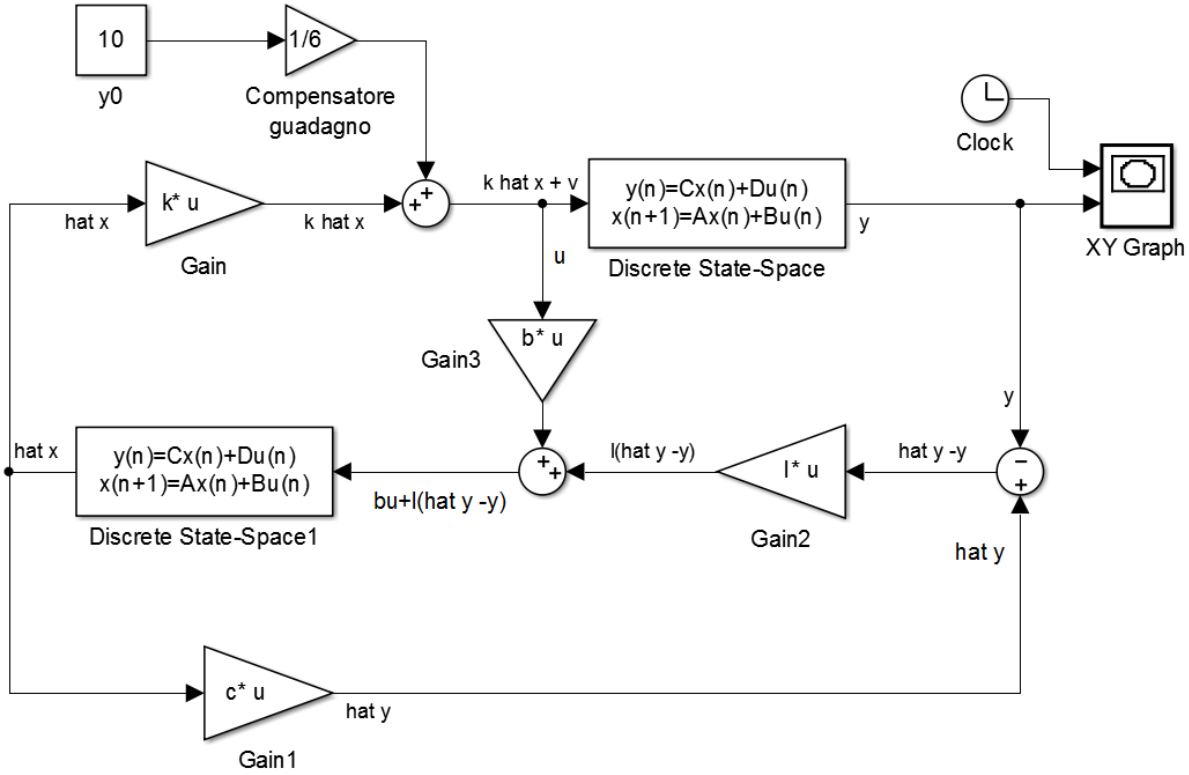
Volendo che il sistema converga a $y = y^0$ è necessario che il guadagno del sistema tra i due segnali sia unitario.
Il guagadno del sistema regolato è

$$G = [c \quad 0]\left(\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} - \begin{bmatrix}A & b\,k\\ -l\,c & A + bk + lc\end{bmatrix}\right)^{-1}\begin{bmatrix}b\\ b\end{bmatrix} + d$$

e possiamo calcolarlo tramite i comandi Matlab

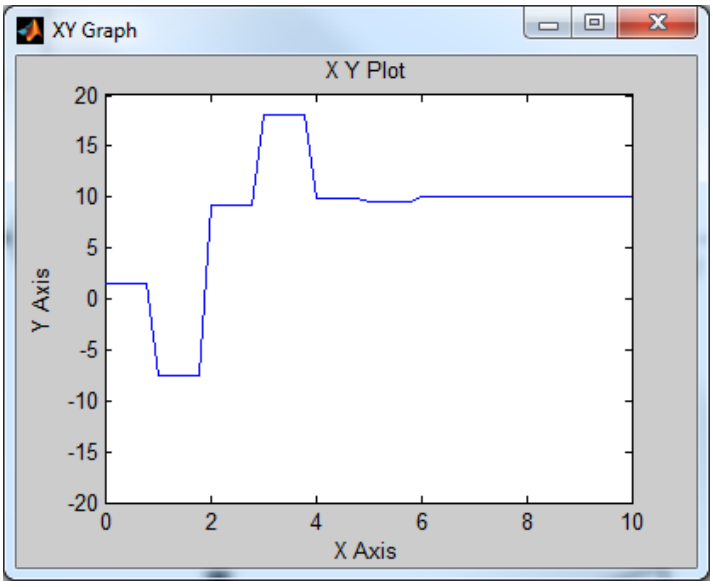
```
matriciona= [A, b*k; -l*c, A+l*c+b*k];
guadagnochiuso=[c,zeros(1,3)]*inv(eye(6)-matriciona)*[b;b]

guadagnochiuso =
   6
```

Compensando quindi il guadagno in anello chiuso otteniamo quanto richiesto




Il risutlato della simulazione sarà quindi (ad esempio, poichè la condizione iniziale è casuale) quello riportato in figura.

Si noti che il sistema converge al valore desiderato in 6 (ovvero 2 $N$) costanti di tempo.



## Esercizio 2: Legge di controllo e ricostruzione dello stato; progetto del regolatore

1. Il sistema in forma di stato, omettendo la dipendenza delle variabili dal tempo, può essere riscritto nel seguente modo:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -\frac{m}{M} g x_3 + \frac{1}{M} u_1 - \frac{1}{lM} u_2 \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = \frac{g}{l}\frac{M+m}{M} x_3 - \frac{1}{lM} u_1 + \frac{1}{l^2}\frac{M+m}{Mm} u_2 \end{cases}$$

   Pertanto, le matrici del sistema in forma di stato sono le seguenti:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{m}{M} g & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{g}{l}\frac{M+m}{M} & 0 \end{bmatrix}; B = \begin{bmatrix} 0 & 0 \\ \frac{1}{M} & -\frac{1}{lM} \\ 0 & 0 \\ -\frac{1}{lM} & \frac{1}{l^2}\frac{M+m}{Mm} \end{bmatrix}; C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}; D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix};$$

2. Per verificare che il sistema sia raggiungibile e osservabile con gli ingressi e le uscite scelte occorre mostrare che le matrici di raggiungibilità $R$ e di osservabilità $O$ abbiano rango pieno (ovvero determinante non nullo); per fare ciò occorre innanzitutto selezionare le matrici del sistema in modo corretto; si noti infatti che l'ingresso "forza" corrisponde alla prima colonna della matrice $B$, mentre l'ingresso "coppia" corrisponde alla seconda colonna della stessa. Analogamente, l'uscita "posizione" corrisponde alla prima riga della matrice $C$, mentre l'uscita "posizione angolare" corrisponde alla seconda riga della stessa; pertanto la richiesta del problema può essere risolta nel modo seguente:

```
% - Parametri
M = 10;
m = 1;
l = 1;
g = 9.81;

% - Scrittura delle equazioni di moto in forma matriciale
A = [0 1 0 0;0 0 -m/M*g 0;0 0 0 1;0 0 g/l*(m+M)/M 0];
B = [0 0 ;1/M -1/(l*M);0 0;-1/(l*M) 1/(l^2)*(M+m)/(m*M)];
C = [1 0 0 0;0 0 1 0];
D = [0 0;0 0];

% - Osservabilità e raggiungibilità

% Prima coppia: forza - posizione
disp('Controllabilità forza')
co = ctrb(A,B(:,1))             % selezione ingresso forza
det(co)
rank(co)
disp('Osservabilità posizione')
ob = obsv(A,C(1,:))             % selezione misura posizione
det(ob)
rank(ob)
```

```
% Seconda coppia: coppia - posizione angolare
disp('Controllabilità coppia')
co = ctrb(A,B(:,2))             % selezione ingresso coppia
det(co)
rank(co)
disp('Osservabilità posizione angolare')
ob = obsv(A,C(2,:))             % selezione misura angolo
det(ob)
rank(ob)


Controllabilità forza

co =

         0    0.1000         0    0.0981
    0.1000         0    0.0981         0
         0   -0.1000         0   -1.0791
   -0.1000         0   -1.0791         0


ans =

    0.0096


ans =

     4

Osservabilità posizione

ob =

    1.0000         0         0         0
         0    1.0000         0         0
         0         0   -0.9810         0
         0         0         0   -0.9810


ans =

    0.9624


ans =

     4

Controllabilità coppia

co =

         0   -0.1000         0   -1.0791
   -0.1000         0   -1.0791         0
         0    1.1000         0   11.8701
    1.1000         0   11.8701         0
```

```
ans =

     0


ans =

     2

Osservabilità posizione angolare

ob =

         0         0    1.0000         0
         0         0         0    1.0000
         0         0   10.7910         0
         0         0         0   10.7910


ans =

     0


ans =

     2
```

Il comando `rank()`, restituisce il valore del rango della matrice.
Pertanto si è dimostrato quanto chiesto dal testo al punto 2.

3. Per progettare una retroazione dello stato (legge di controllo *k*) che assegni gli autovalori come chiesti si utilizza la formula di Ackermann e il comando matlab `acker()` (vedi esercizio 1). Pertanto si procede come segue:

```
% - Assegnamento degli autovalori
% Ricavo le radici del polinomio
disp('Autovalori desiderati')
autovalori = [roots([1 1.5 1]);roots([1 2 1])]

Autovalori desiderati

autovalori =

  -0.7500 + 0.6614i
  -0.7500 - 0.6614i
  -1.0000
  -1.0000

% - Progetto la legge di controllo
disp('Legge di controllo k')
k = acker(A,-B(:,1),autovalori)

Legge di controllo k
```

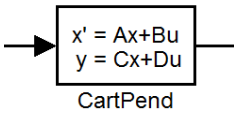
```
k =

   1.0194   3.5678   158.9294   38.5678

disp('Verifica autovalori')
eig(A + B(:,1)*k)
autovalori

Verifica autovalori

ans =

  -0.7500 + 0.6614i
  -0.7500 - 0.6614i
  -1.0000 + 0.0000i
  -1.0000 - 0.0000i


autovalori =

  -0.7500 + 0.6614i
  -0.7500 - 0.6614i
  -1.0000
  -1.0000
```

4. In modo analogo si può ottenere il ricostruttore dello stato (vedi esercizio 1 per l'uso del comando `acker()`);

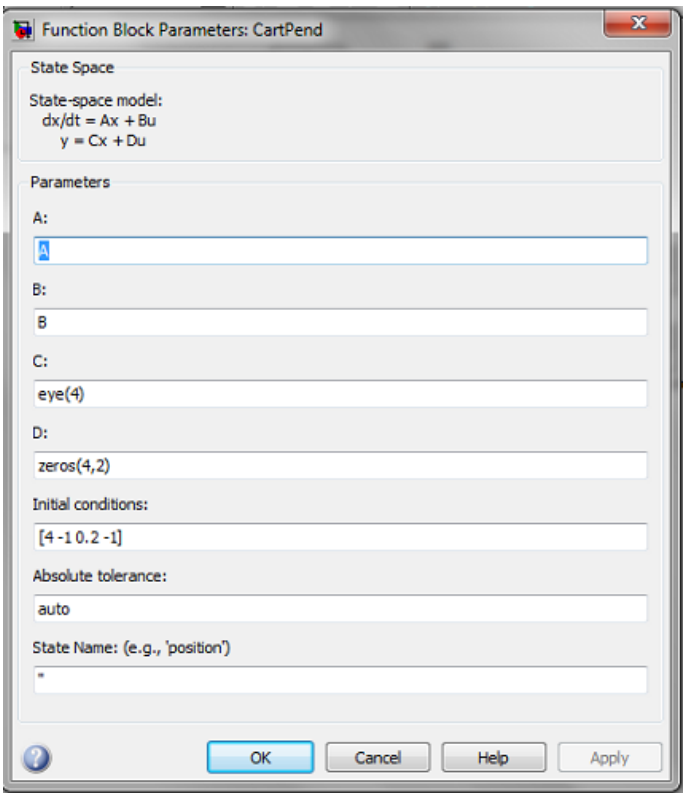
```
% - Progetto ricostruttore di stato
% Ricavo le radici del polinomio
disp('Autovalori desiderati')
autov_ricostruttore = [roots([1 15 100]);roots([1 20 100])]

Autovalori desiderati

autov_ricostruttore =

  -7.5000 + 6.6144i
  -7.5000 - 6.6144i
 -10.0000
 -10.0000

% - Ricostruttore dello stato
disp('Ricostruttore dello stato l')
l = acker(A',-C(1,:).',autov_ricostruttore)'

Ricostruttore dello stato l

l =

  1.0e+004 *

    -0.0035
    -0.0511
     0.3953
     1.5812
```

```
disp('Verifica autovalori')
eig(A + l*C(1,:))
autov_ricostruttore

Verifica autovalori

ans =

  -7.5000 + 6.6144i
  -7.5000 - 6.6144i
 -10.0000 + 0.0000i
 -10.0000 - 0.0000i

autov_ricostruttore =

  -7.5000 + 6.6144i
  -7.5000 - 6.6144i
 -10.0000
 -10.0000
```

5. Innanzitutto, per poter retroazionare il sistema bisogna definire in simulink il sistema stesso; pertanto possiamo utilizzare il blocco (già visto nell'esercitazione 2) State-Space; come uscita di tale blocco ci occorre proprio lo stato (che supponiamo tutto noto).




Pertanto le proprietà del suddetto blocco saranno le seguenti:

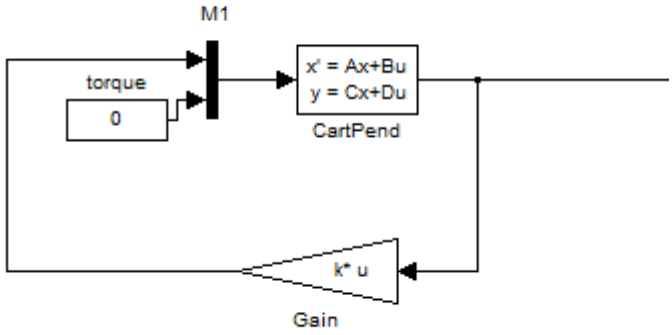


Questo perché le matrici *A* e *B* descrivono la dinamica e gli ingressi del blocco, mentre la matrice *C* descrive l'uscita dello stesso; se si vogliono ottenere le informazioni su tutto lo stato occorre che la matrice *C* sia una matrice identità e che la matrice *D* sia nulla.

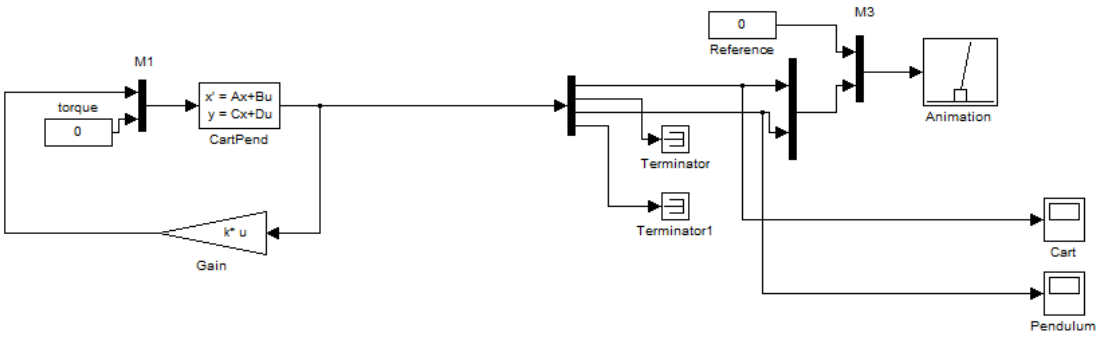
Infatti si ha:

$$Y = Cx + Du = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + 0 = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Le condizioni iniziali vengono poste diverse dal vettore nullo; pertanto il pendolo parte da una condizione fuori dall'equilibrio.
Il regolatore stabilizzante non è altro (si veda al punto 4) che una retroazione dello stato, agente sul primo dei due ingressi del sistema (la forza impressa sul carrello).
Dunque, per realizzare lo schema, basta moltiplicare lo stato con un blocco gain di simulink che abbia come parametri il vettore di guadagni trovato al punto 4; occorrerà poi considerare come secondo ingresso (coppia fornita all'asta) una costante nulla. Ricordando che per generare un vettore da più scalari/vettori si può utilizzare il blocco MUX di simulink, lo schema risulta essere:

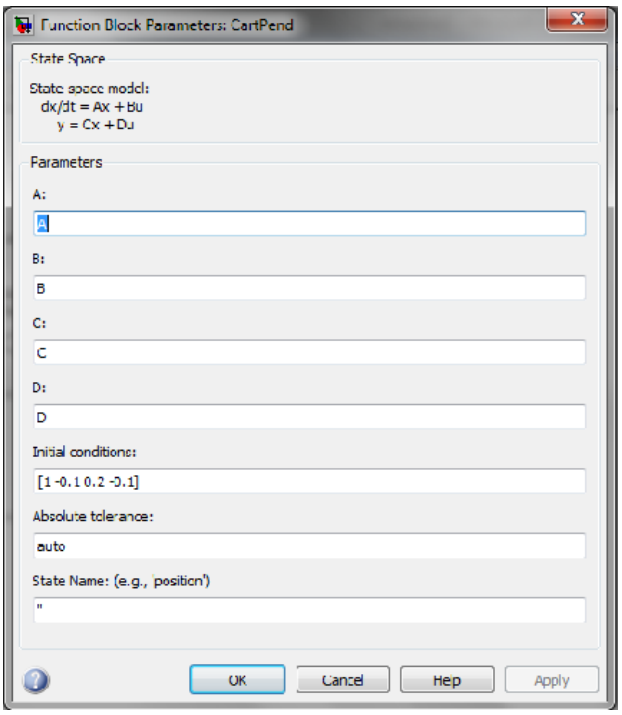



Per rappresentare ora le uscite occorre separare il vettore degli stati e prendere il primo e il terzo stato, che sono per l'appunto posizione del carrello e posizione angolare dell'asta (si veda il testo dell'esercizio). Si può utilizzare il blocco animation, che riceve proprio queste due informazioni come ingresso e simula il movimento del pendolo.
Lo schema complessivo, che rappresenta anche gli andamenti delle uscite nel tempo, risulta (file simulink simcartp_controller.mdl):




Si noti che sono presenti due blocchi "terminator": tali blocchi servono solo per evitare che simulink dia un messaggio di "warning" dovuto a un arco non connesso. Sono blocchi non necessari, che però rendono più ordinato lo schema e velocizzano la simulazione (nel caso di sistemi più complessi limitare i messaggi di "warning" risulta critico).

6. Si consideri ora lo schema del punto 5; il fatto che non si possa conoscere lo stato implica il dover ricostruire dalle misure (e quindi dall'uscita) lo stato stesso; pertanto non occorre più che la matrice C sia una matrice identità; si può dunque scrivere, nei parametri dello state space, la matrice C e la matrice D del sistema vero:



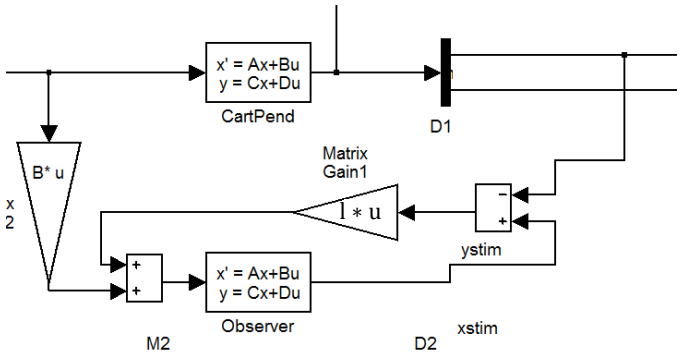

A questo punto, grazie a un DEMUX, si può selezionare la prima uscita, che è la posizione del carrello, ossia la misura a disposizione; avendo a disposizione l'arco con gli ingressi del sistema e l'arco con la misura è possibile implementare le equazioni del ricostruttore in simulink; si ricorda che le equazioni di un generico ricostruttore sono le seguenti:

$$\begin{cases} \dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) + l(y(t) - \hat{y}(t)) \\ \hat{y}(t) = C\hat{x}(t) \end{cases}$$

Dove *A*, *B* e *C* sono le matrici del sistema e l è il ricostruttore dello stato calcolato al punto 4. Pertanto si può di nuovo utilizzare il blocco state space di simulink; si noti che ora l'ingresso a tale blocco è dato da $Bu(t) + l(y(t) - \hat{y}(t))$; per costruire questo ingresso si spezzi in due la somma:

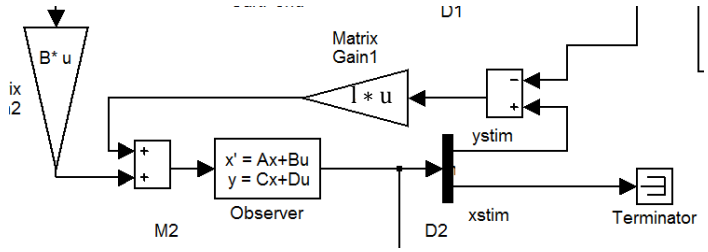
- $Bu(t)$: per questa parte basta inserire un blocco gain con i coefficienti appropriati (quelli presenti nella matrice *B*)
- $l(y(t) - \hat{y}(t))$: per questo addendo occorre inserire un nodo sommatore che sottragga alla misura $y(t)$ la sua stima; come si nota dall'equazione dell'osservatore, la stima di $y(t)$ è l'uscita del blocco stesso. Infine l'uscita del nodo sommatore va moltiplicata per *l* (blocco gain).

Dunque:

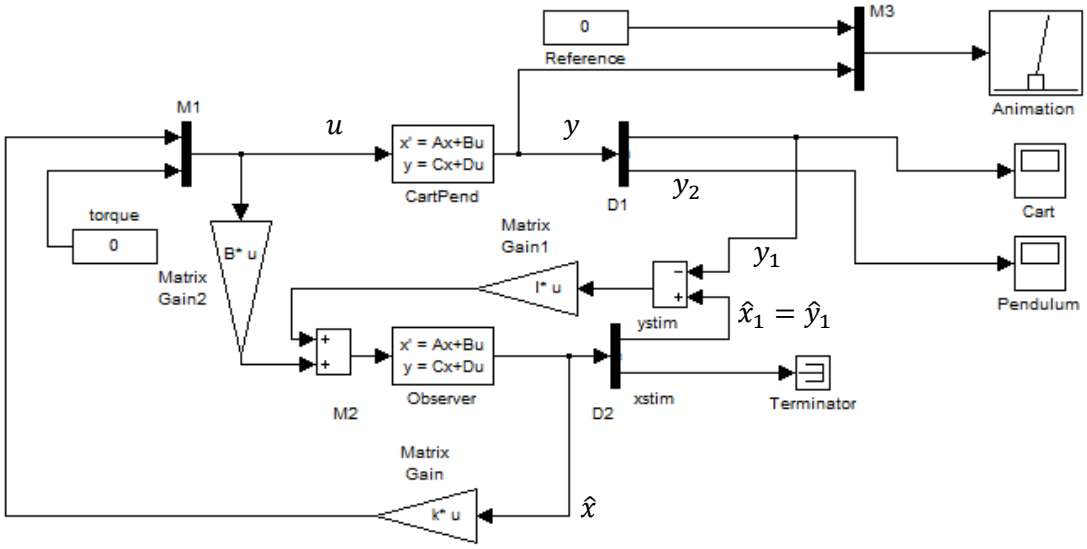



Tuttavia, si noti che occorre retroazionare TUTTO lo stato stimato; pertanto, si modifica la matrice C dell'osservatore in modo da selezionare tutto lo stato dello stesso (pertanto, come

visto al punto precedente, si utilizzi una matrice identità). Occorre ora selezionare, come stima della misura, la prima componente del vettore stato e connetterla con il nodo sommatore precedentemente visto:




A questo punto, si può retroazionare lo stato stimato attraverso il guadagno *k* calcolato al punto 3 per stabilizzare il sistema; lo schema complessivo sarà dunque (vedi schema simulink simcartp_complete.mdl):



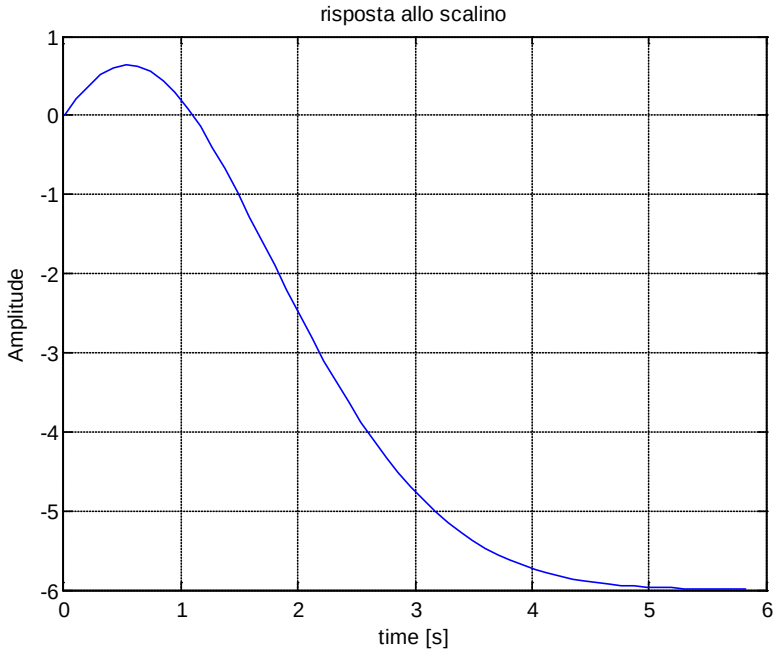

N.B. Confronto tra gli schemi dei punti 5 e 6:
Gli autovalori in anello chiuso derivanti dall'azione stabilizzante (sistema raggiungibile) inserita nello schema 5 sono gli autovalori della matrice $(A + bk)$; grazie all'assegnamento degli autovalori si può "decidere" il tempo di assestamento del sistema. Se lo stato non è accessibile (schema 6), ma il sistema è osservabile, inserendo l'osservatore del sistema si aggiungono degli autovalori, uguali a quelli della matrice $(A + lC)$. Pertanto, scegliendo tali autovalori tali che restino "dominanti" quelli di $(A + bk)$, il tempo di assestamento del sistema non viene influenzato dall'osservatore. Ciò che risulta degradato sono le prestazioni dinamiche del sistema.

## Esercizio 3: risposte canoniche

Per tracciare le risposte canoniche a scalino e impulso del sistema dato occorre definire il sistema in spazio di stato nel workspace di matlab (vedi esercitazione 1); pertanto i comandi da digitare nel prompt sono i seguenti:

```
>> A=[1 1 0; -5 -3 0; 2 3 -1];
>> b=[1; 1; 0];
>> c=[1 1 1];
>> d=0;                          % definizione matrici del sistema
>> S=ss(A,b,c,d);                % definizione del sistema
>> [out_sca,time_sca]=step(S);   % generazione dati risposta scalino
```

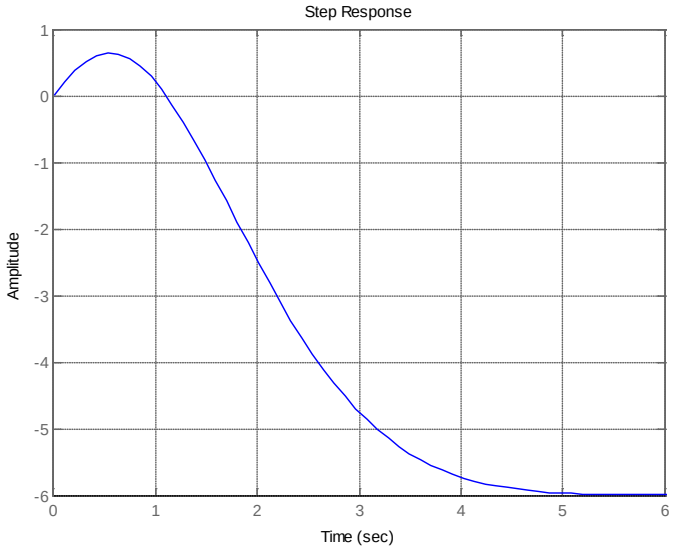
```
>> plot(time_sca,out_sca);      % rappresentazione risposta scalino
```




Alternativa: matlab mostra direttamente il grafico del risultato della risposta a scalino se il comando è dato nel seguente modo:

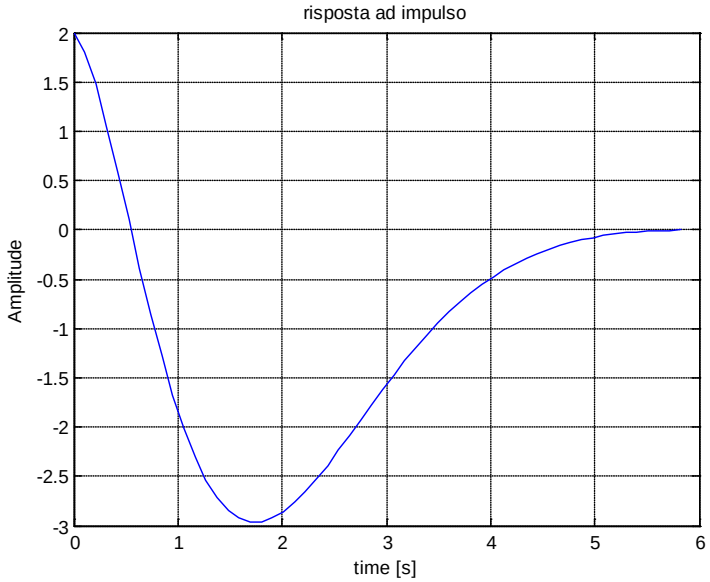
```
>> step(S);
```

Lo svantaggio in tal caso è che non si hanno a disposizione i vettori in cui sono memorizzati i valori della risposta allo scalino.




Analogamente, per la risposta all'impulso si può usare il comando `impulse()`:

```
>> [out_imp,time_imp]=impulse(S); % gener. dati risposta impulso
>> plot(time_imp,out_imp);        % rappr. risposta scalino
```




## Esercizio 4: risposte canoniche

Innanzitutto, a partire dai dati forniti dal problema ricaviamo le costanti di deflusso *k* dei tre laghi e le portate d'acqua uscenti dal lago di Segrino (entrante nel lago di Pusiano) e dal lago di Alserio (confluente nel Lambro con la portata uscente da Pusiano e regolata al Cavo Diotti).

A tale scopo si ricordi che nel modello lineare di lago, posto $x(t)$ il volume di invaso del lago, $u(t)$ le portate entranti nel lago e $y(t)$ la portata uscente dal lago, si ha:

$\dot{x}(t) = -kx(t) + u(t)$

$y(t) = kx(t)$

da cui, all'equilibrio: $\bar{u} = k\bar{x}$ e $\bar{y} = \bar{u} = k\bar{x}$.

Inoltre, il tempo di risposta (pari al tempo di ricambio del lago) è dato da $T_R$=5/*k*.

Pertanto, si ha:

– Lago di Alserio

Costante di deflusso: $k_{ALSERIO} = \frac{5}{1.9*365}\,\text{giorni}^{-1} = 0.0072\,\text{giorni}^{-1}$

Portata di uscita: $\bar{y}_{ALSERIO} = 0.0072 \cdot 6.55 \cdot 10^6\ \text{m}^3\text{giorni}^{-1} = 47160\,\text{m}^3\text{giorni}^{-1}$

– Lago di Segrino

Costante di deflusso: $k_{SEGRINO} = \frac{5}{0.8*365}\,\text{giorni}^{-1} = 0.0171\,\text{giorni}^{-1}$

Portata di uscita: $\bar{y}_{SEGRINO} = 0.0171 \cdot 1.2 \cdot 10^6\ \text{m}^3\text{giorni}^{-1} = 20520\,\text{m}^3\text{giorni}^{-1}$

– Lago di Pusiano

Portata di uscita $\bar{y}_{PUSIANO} = \bar{y}_{LAMBRONE} + \bar{y}_{FALDA} + \bar{y}_{SEGRINO} =$

$$= (0.87 \cdot 86400 + 0.51 \cdot 86400 + 20520)\ \text{m}^3\text{giorni}^{-1} = 139752\,\text{m}^3\text{giorni}^{-1}$$

Costante di deflusso: $k_{PUSIANO} = \frac{\bar{y}_{PUSIANO}}{\bar{x}_{PUSIANO}} = \frac{139752}{12.7 \cdot 10^6}\,\text{giorni}^{-1} = 0.011\text{giorni}^{-1}$

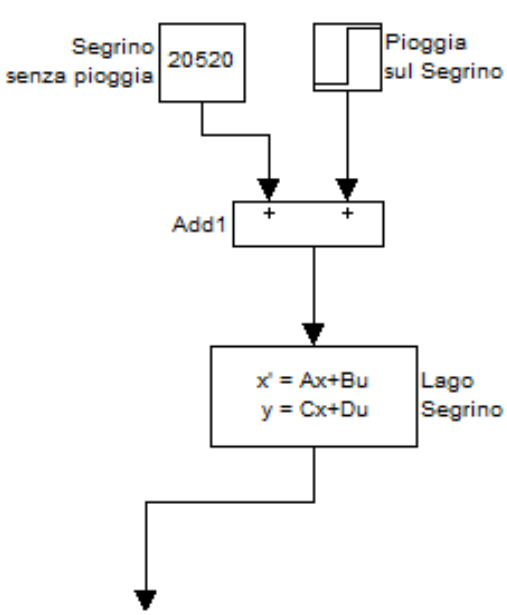
Tali dati si riferiscono al caso in cui le paratie del Cavo Diotti sono regolate in condizione nominale di funzionamento.

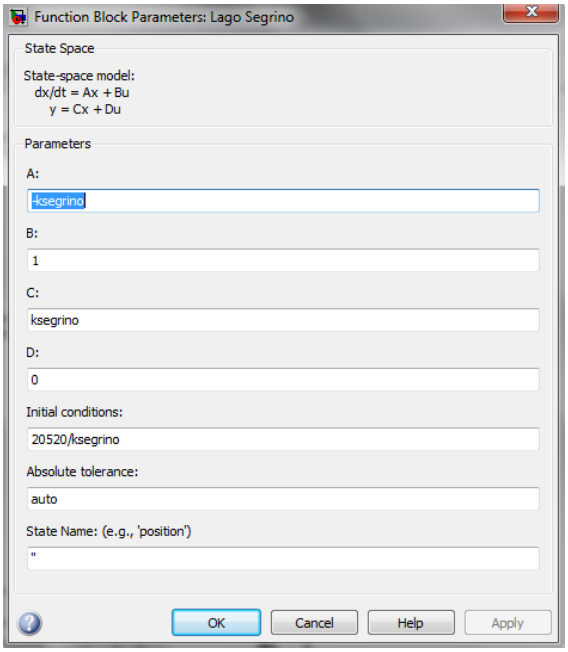
La regolazione delle paratie del Cavo Diotti comporterà la seguente variazione della costante di deflusso del lago di Pusiano:

$k_{PUSIANO\ REGOLATO} = (k_{PUSIANO} + k_{PARATIE}) = (0.011 + k_{PARATIE})$ giorni$^{-1}$

dove il parametro $k_{PARATIE}$ è positivo se le paratie vengono aperte per aumentare la portata di deflusso dal lago, negative se vengono chiuse per diminuirla.

Lo schema Simulink della rete idrica è riportato nelle pagine seguenti (laghiBrianza.mdl).
I tre laghi sono modellizzati mediante un sistema lineare del primo ordine (come sopra specificato). In particolare, lo schema per il lago Segrino è dato da:

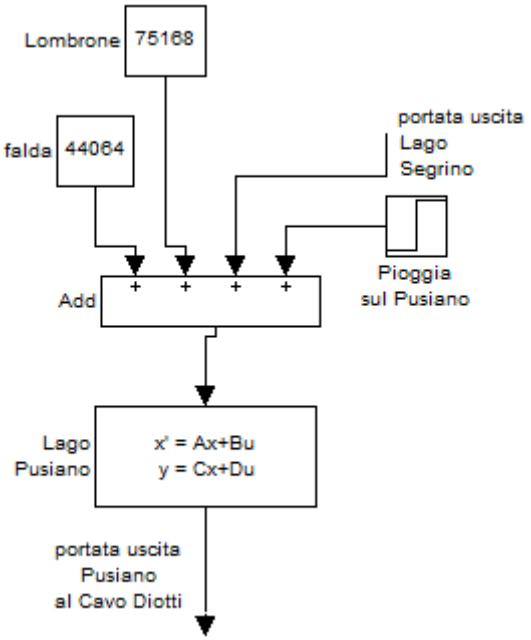


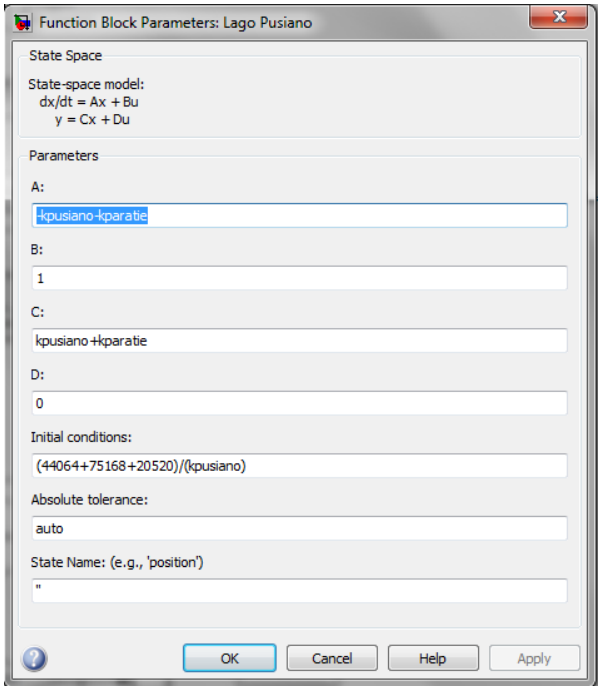




La condizione iniziale viene scelta in modo tale che, in assenza di pioggia sul Segrino, il lago si trovi già in condizioni di equilibrio (portata di uscita pari alla portata in ingresso). Infatti,

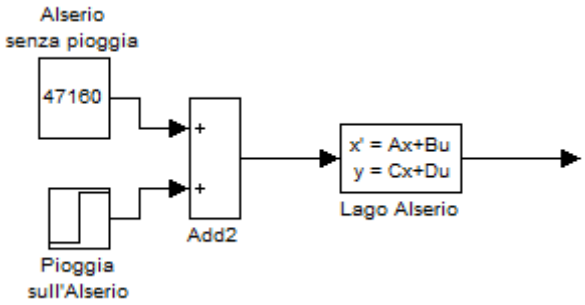
$\dot{x}_{SEGRINO}(t) = -k_{SEGRINO}\ x_{SEGRINO}(t) + 20520 + u_{PIOGGIA}(t)$

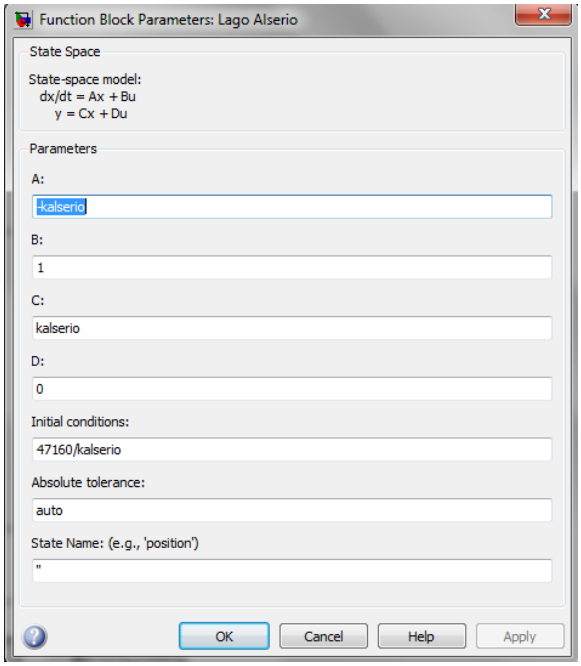
e ponendo $x_{SEGRINO}(0) = 20520 / k_{SEGRINO}$ e $u_{PIOGGIA}(t) = 0$ si ottiene $\dot{x}_{SEGRINO}(t) = 0$ cioè la situazione di equilibrio.
Lo schema per il lago di Pusiano è dato da



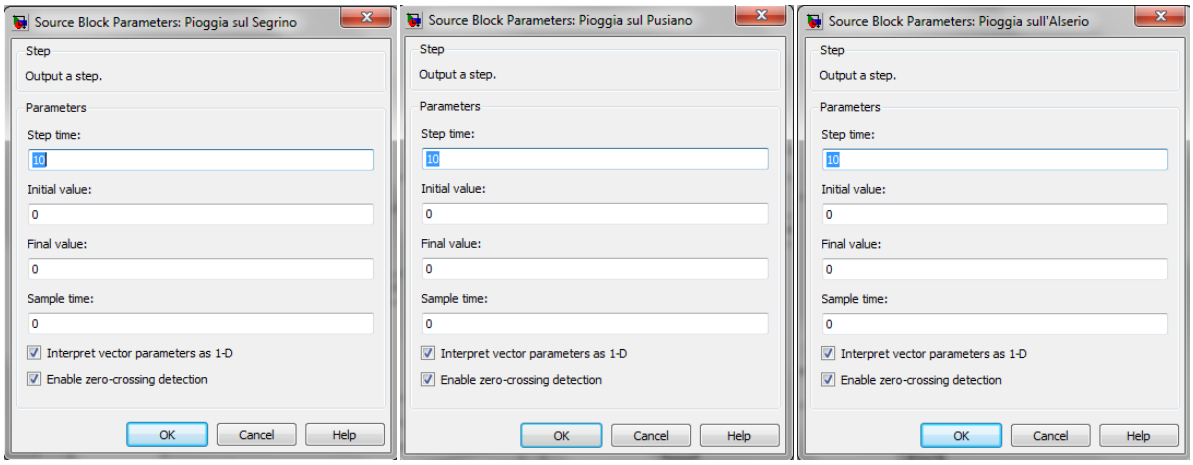




mentre quello per il lago di Alserio è




La motivazione della scelta delle condizioni iniziali per questi due laghi è analoga a quella per il lago di Segrino.
Inizializziamo gli eventi di pioggia a valore nullo



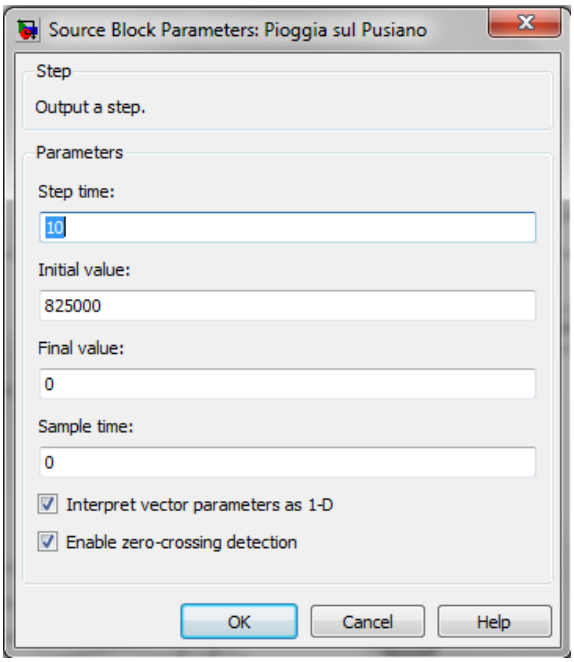

così come l'apertura delle paratie.

Al prompt del matlab dichiariamo alcuni parametri che vengono utilizzati dallo schema simulink:

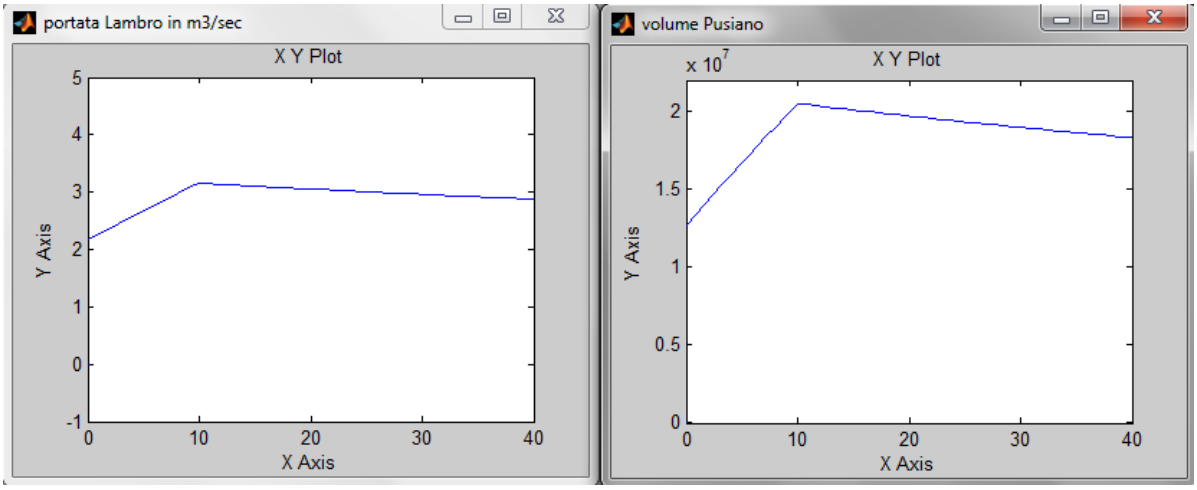
```
>> clear all; close all;
>> kalserio=5/(1.9*365);              % costante di deflusso Alserio
>> ksegrino=5/(0.8*365);              % costante di deflusso Segrino
>> kpusiano=139752/(12.7*10^6);       % costante di deflusso Pusiano
                                      % non regolato
>> kparatie=0;                        % apertura delle paratie
```

Simulando lo schema, si ottiene che la portata del Lambro (`>> portata_Lambro`) è pari a 2.1633 $m^3$ secondo $^{-1}$ = 186910 $m^3$ giorno $^{-1}$ = $\bar{y}_{PUSIANO} + \bar{y}_{ALSERIO}$ mentre il volume del lago di Pusiano (`>> volume_Pusiano`) è pari a 12,7 milioni di $m^3$. Questi valori sono inferiori ai valori di esondazione del lago e massima portata del Lambro.

Supponendo ora che sul lago di Pusiano insista un evento di pioggia pari a 150 mm/giorno e della durata di 10 giorni, è necessario sottoporre il sistema Pusiano a un impulso di durata finita (10 giorni) e ampiezza $0.150 \cdot 5.5 \cdot 10^6 = 825000\,\mathrm{m}^3\mathrm{giorni}^{-1}$.



In corrispondenza di tale evento si nota l'esondazione del lago di Pusiano (volume superiore ai 18 milioni di $m^3$ mentre la portata del Lambro è inferiore alla massima consentita (7 $m^3$/secondo).

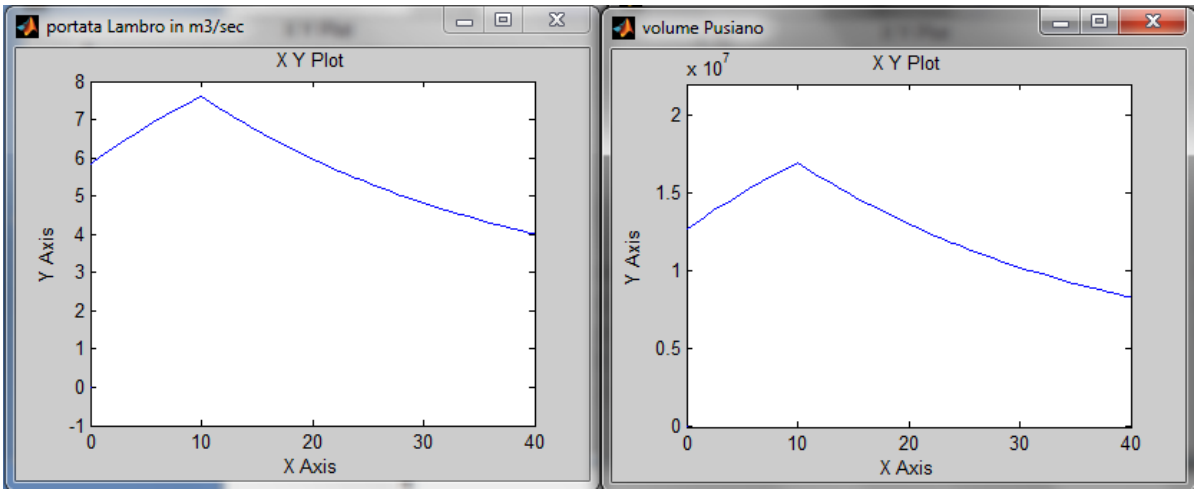



Per evitare l'esondazione del lago occorre quindi aprire le paratie al Cavo Diotti in modo da aumentarne la portata di uscita diminuendone il volume. Tuttavia, così facendo si aumenta la portata del Lambro; occorrerà quindi monitorare che questa non superi il suo valore massimo.

Ponendo ad esempio

```
>> kparatie=0.025;
```

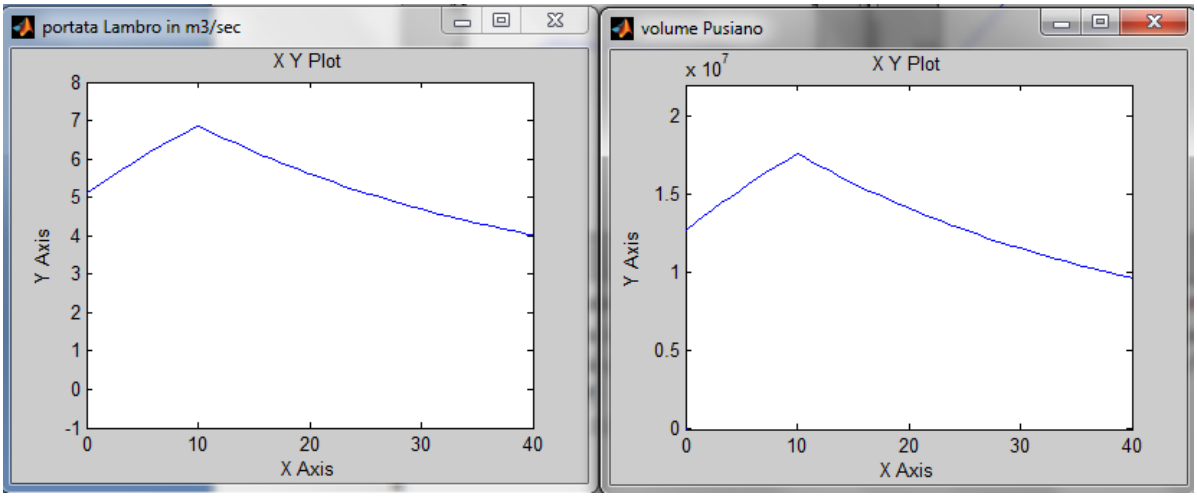
si evita l'esondazione del Pusiano ma il superamento della portata massima per il Lambro.




Occorre quindi aprire di meno le paratie. Per esempio, ponendo

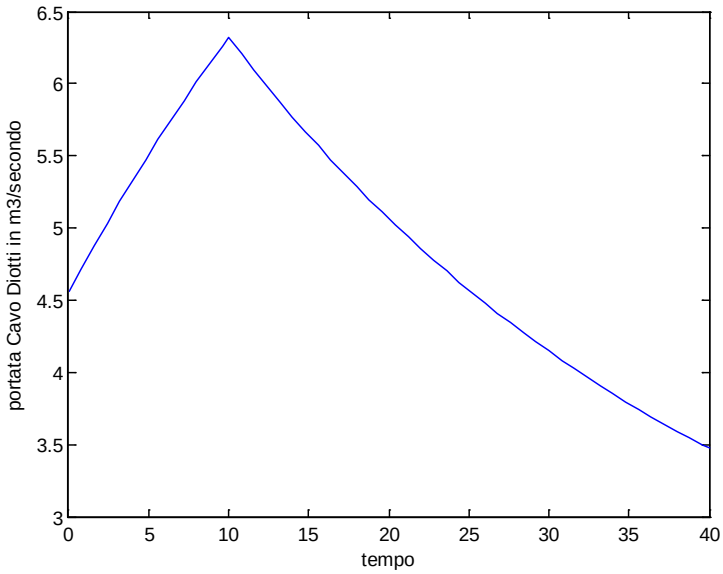
```
>> kparatie=0.02;
```

entrambi i vincoli sono soddisfatti.



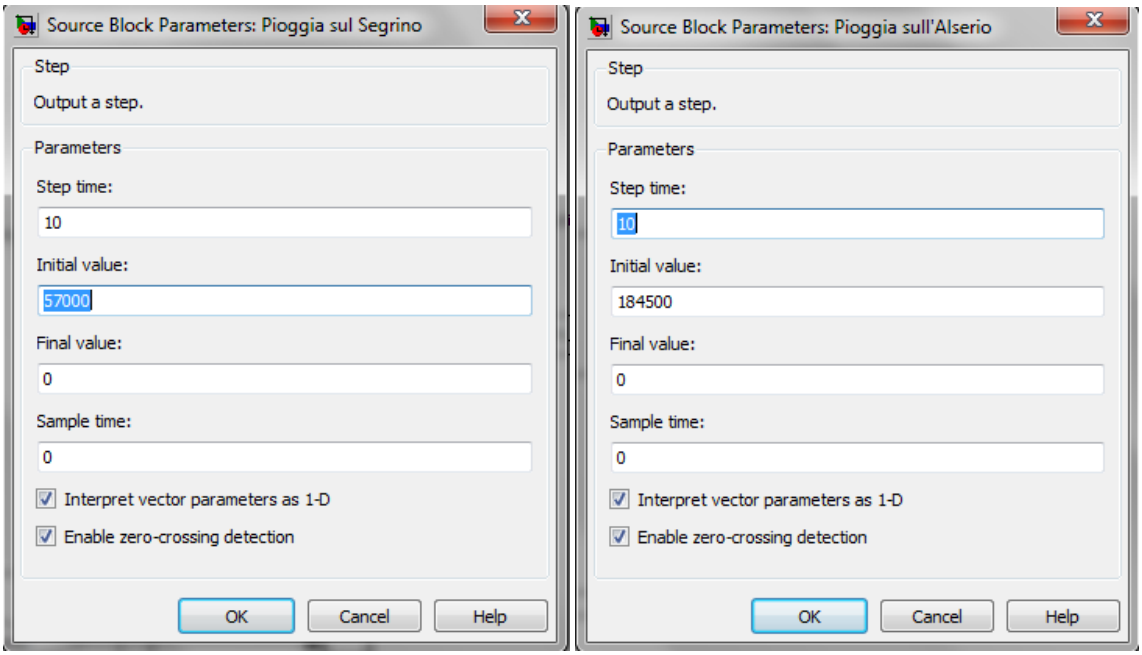
Ciò corrisponde a una portata di uscita al Cavo Diotti (in $m^3$/secondo) pari a

```
>> plot(tempo,(kpusiano+kparatie)*volume_Pusiano/86400)
```




Si ricordi che, in assenza di regolazione ($k_{PUSIANO}$ = 0), la portata di uscita dal lago di Pusiano era pari a $139752\,\mathrm{m}^3\mathrm{giorni}^{-1} = 1.6175\mathrm{m}^3\mathrm{secondo}^{-1}$.

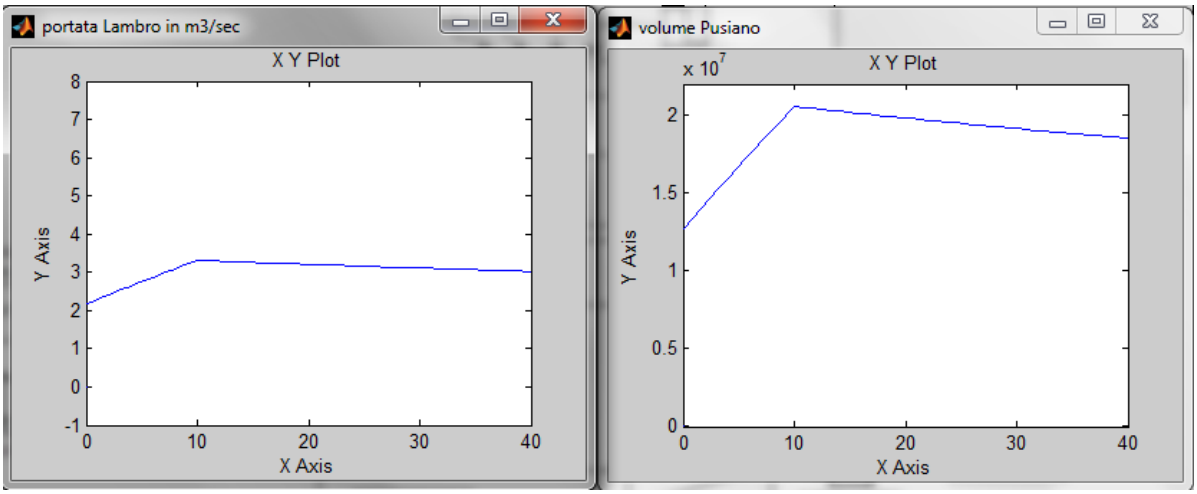
Infine, se lo stesso evento di pioggia insiste su tutti e tre i laghi, occorrere sottoporre i sistemi Segrino e Alserio a impulsi di durata finita (10 giorni) e ampiezza rispettivamente pari a $0.150 \cdot 0.38 \cdot 10^6 = 57000\,\mathrm{m}^3\mathrm{giorni}^{-1}$ e $0.150 \cdot 1.23 \cdot 10^6 = 184500\,\mathrm{m}^3\mathrm{giorni}^{-1}$.




In assenza di regolazione delle paratie

```
>> kparatie=0;
```

si osserva ancora la sola esondazione del Pusiano.




Tuttavia, la regolazione prima proposta

```
>> kparatie=0.02;
```

non garantisce più il vincolo sulla massima portata del Lambro (simulare)

```
>> max(portata_Lambro)

ans =

    7.0284
```

pur soddisfacendo quello sul rischio di esondazione del lago

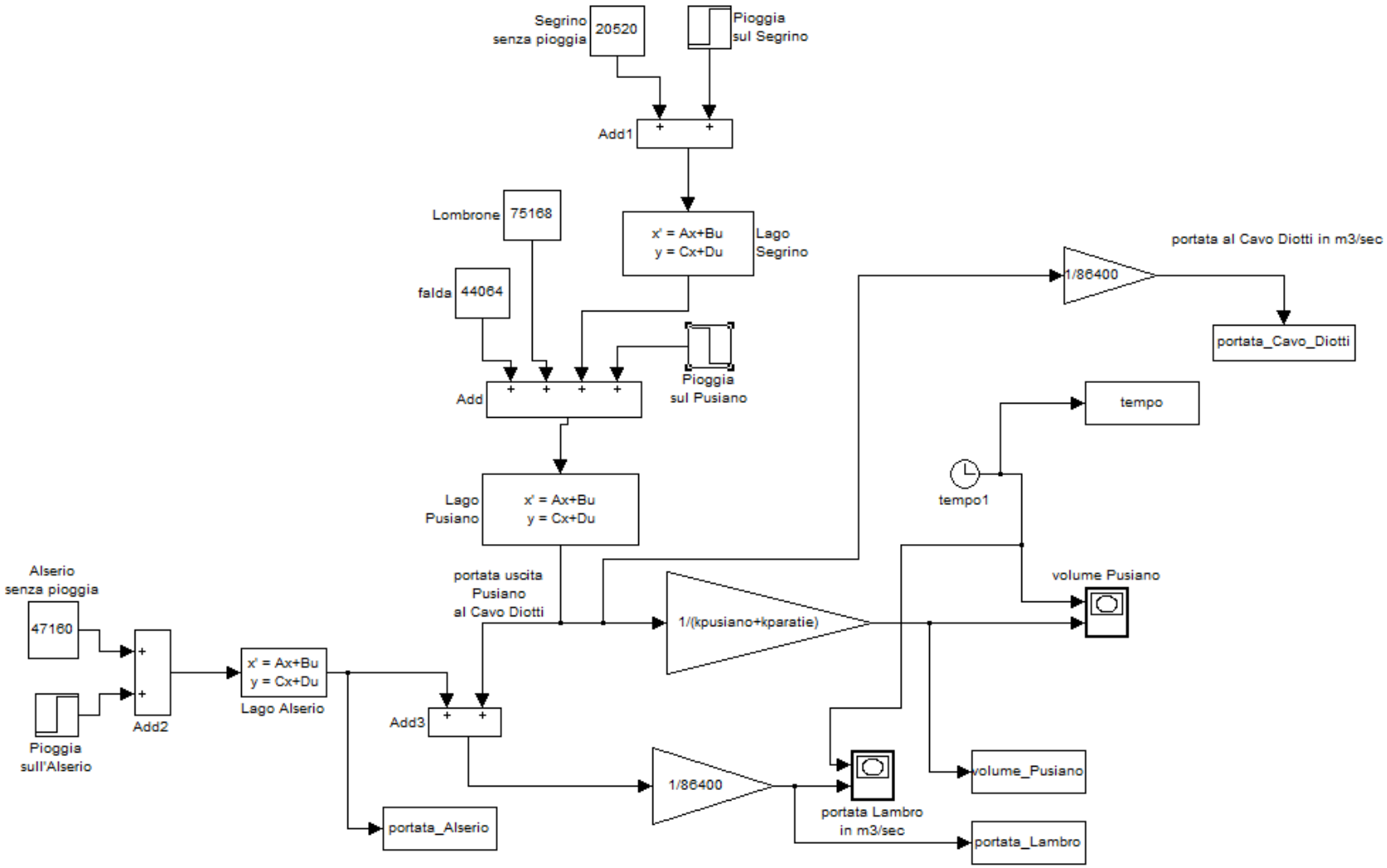
```
>> max(volume_Pusiano)

ans =

  1.7651e+007
```

Occorre chiudere leggermente le paratie per ottenere quanto voluto.

```
>> kparatie=0.0185;
```

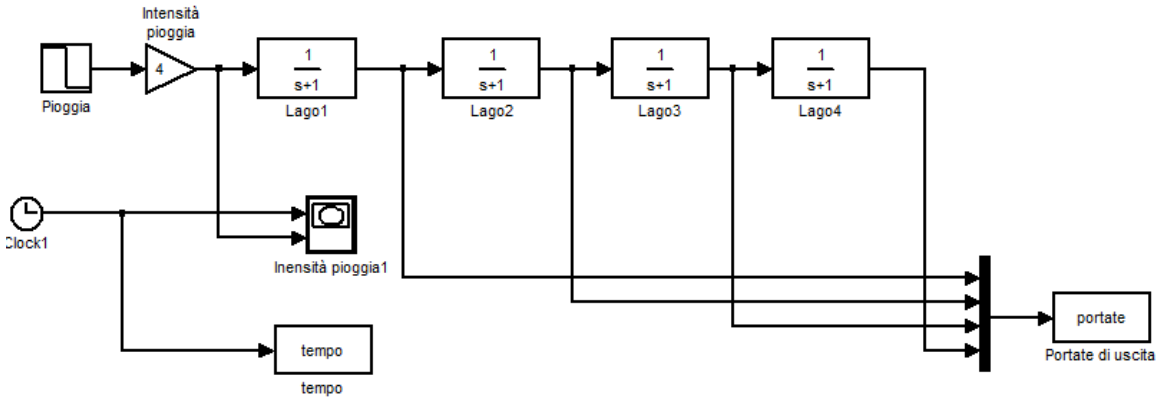
(simulare)

```
>> max(portata_Lambro)

ans =

    6.7904

>> max(volume_Pusiano)

ans =

  1.7852e+007
```


Segrino
senza pioggia
20520
Pioggia
sul Segrino
Add1
Lombrone
75168
x' = Ax+Bu
y = Cx+Du
Lago
Segrino
falda
44064
Pioggia
sul Pusiano
Add
Lago
Pusiano
x' = Ax+Bu
y = Cx+Du
portata uscita
Pusiano
al Cavo Diotti
1/86400
portata al Cavo Diotti in m3/sec
portata_Cavo_Diotti
tempo
tempo1
volume Pusiano
1/(kpusiano+kparatie)
Alserio
senza pioggia
47160
Add2
Pioggia
sull'Alserio
x' = Ax+Bu
y = Cx+Du
Lago Alserio
Add3
portata_Alserio
1/86400
portata Lambro
in m3/sec
volume_Pusiano
portata_Lambro

**Esercizio 5: Risposte canoniche**

Lo schema simulink della rete idrica composta, per esempio, da quattro laghi in cascata (identici e con costante di deflusso unitaria) è mostrato nella figura seguente (file laghi.mdl)




Il grafico seguente (`plot(tempo,portate)`) dimostra quanto richiesto