

**Laborator Prelucrarea Numerică a Semnalelor**  
**Proprietățile sistemelor**  
**Implementare în Matlab / Simulink**

**Ștefan-Adrian Toma**  
**V1 2020**

## Cuprins

1. Introducere.....	3
2. Memoria.....	3
2.1    Sisteme fără memorie.....	3
2.2    Sisteme cu memorie .....	4
3. Inversabilitatea.....	4
3.1    Exemplu Matlab .....	4
4. Stabilitatea .....	5
4.1    Exemplu .....	5
4.2    Implementare Matlab .....	5
4.3    Implementare Simulink.....	7
5. Cauzalitatea .....	10
5.1    Exercițiu Matlab .....	10
5.2    Exemplu model Simulink .....	11
6. Liniaritatea.....	16
6.1    Exercițiu Matlab .....	17
6.2    Exemplu model Simulink.....	18
7. Invarianța în timp.....	22
7.1    Exercițiu Matlab .....	22
7.2    Exemplu model Simulink.....	24

## 1. Introducere

Un sistem în timp discret (Figura 1) transformă secvența de intrare  $x[n]$ , într-o altă secvență de ieșire  $y[n]$ . Fie  $S\{x[n]\}$  operatorul sistemului. Răspunsul sistemului este ca în relația (1).

$$y[n] = S\{x[n]\} \quad (1)$$

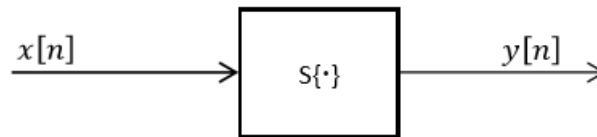


Figura 1. Sistem numeric

Ne propunem să evidențiem în Matlab și Simulink proprietățile sistemelor:

1. Memoria
2. Inversabilitatea
3. Stabilitatea
4. Cauzalitatea
5. Liniaritatea
6. Invarianța în timp

## 2. Memoria

### 2.1 Sisteme fără memorie

Sistemele fără memorie sunt acelea pentru care răspunsul  $y[n]$  depinde doar de valoarea curentă  $x[n]$ , nu și de valori anterioare ale lui  $x$ . Ieșirea sistemului la momentul de timp  $t$  depinde numai de intrarea la acel moment de timp  $t$ .

#### Exemplu

```
x=[0 3 1 3 4 -2 5 -7];  
for i=1:length(x)  
    y(i)=2*x(i);  
end  
stem(y);
```

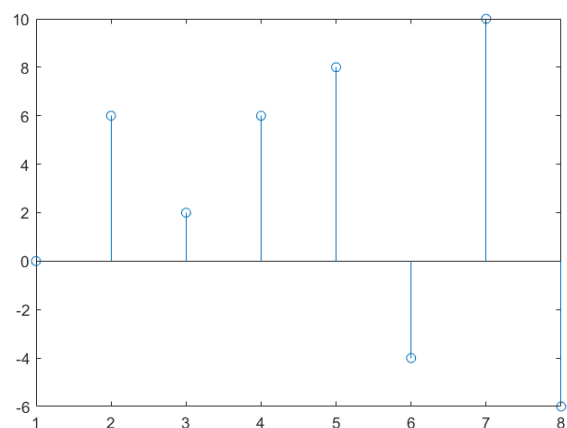


Figura 1. Sistem fără memorie

## 2.2 Sisteme cu memorie

În cazul sistemelor cu memorie, răspunsul acestora depinde și de valorile lui  $x$ , anterioare momentului curent de timp.

### Exemplu

```
x=[0 3 1 3 4 -2 5 -7];  
for i=2:length(x)  
    y(i)=(x(i-1)+x(i))/2;  
end  
stem(y);
```

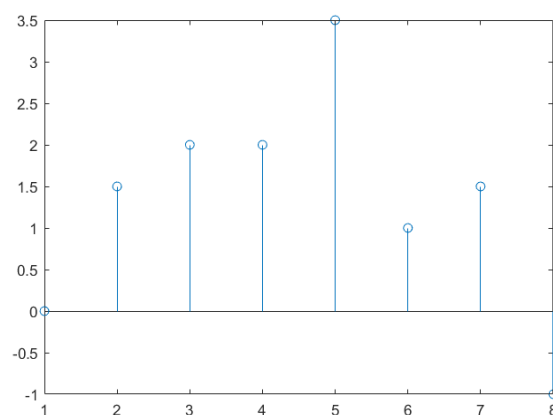
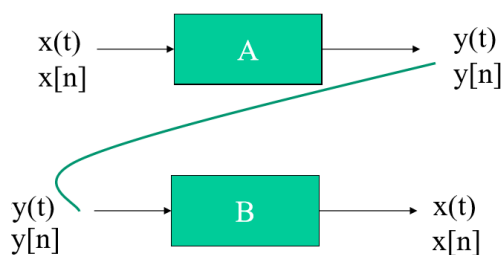


Figura 2. Sistem cu memorie

## 3. Inversabilitatea

Un sistem se numește **inversabil** dacă există un alt sistem (sistem invers) care, conectat în cascadă cu primul, reface semnalul de excitație.



$$\begin{aligned}y[n] &= A\{x[n]\} \\ x[n] &= B\{y[n]\} \\ A &= B^{-1}\end{aligned}$$

### 3.1 Exemplu Matlab

```
n=0:40;  
x1=sin(pi/3*n);  
y=3*x1;  
x2=(1/3)*y;  
subplot(2,1,1);  
plot(x1);  
title('Semnal x1');  
subplot(2,1,2);  
plot(x2);  
title('Semnal x2');
```

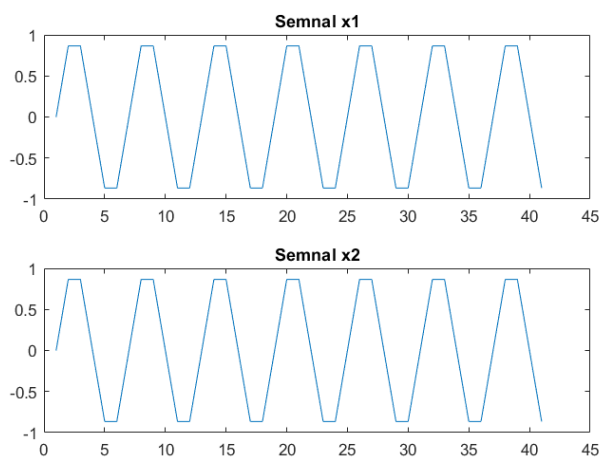


Figura 3. Semnalul de ieșire  $x_2$  din B este același cu semnalul de intrare  $x_1$  în A

Observăm că semnalul  $y$  se poate inversa, astfel încât în  $x_2$  să obținem semnalul inițial  $x_1$ .

## 4. Stabilitatea

Un sistem cauzal, determinist este stabil dacă răspunsurile care apar în sistem pentru intrări de modul mărginit sunt la rândul lor mărginite din punct de vedere energetic.

Pentru un sistem analogic stabil, polii sunt poziționați în semiplanul stâng al planului complex al transformatei Laplace. În cazul unui sistem numeric, condiția de stabilitate este îndeplinită dacă polii se găsesc în interiorul cercului de rază unitate din planul transformatei Z.

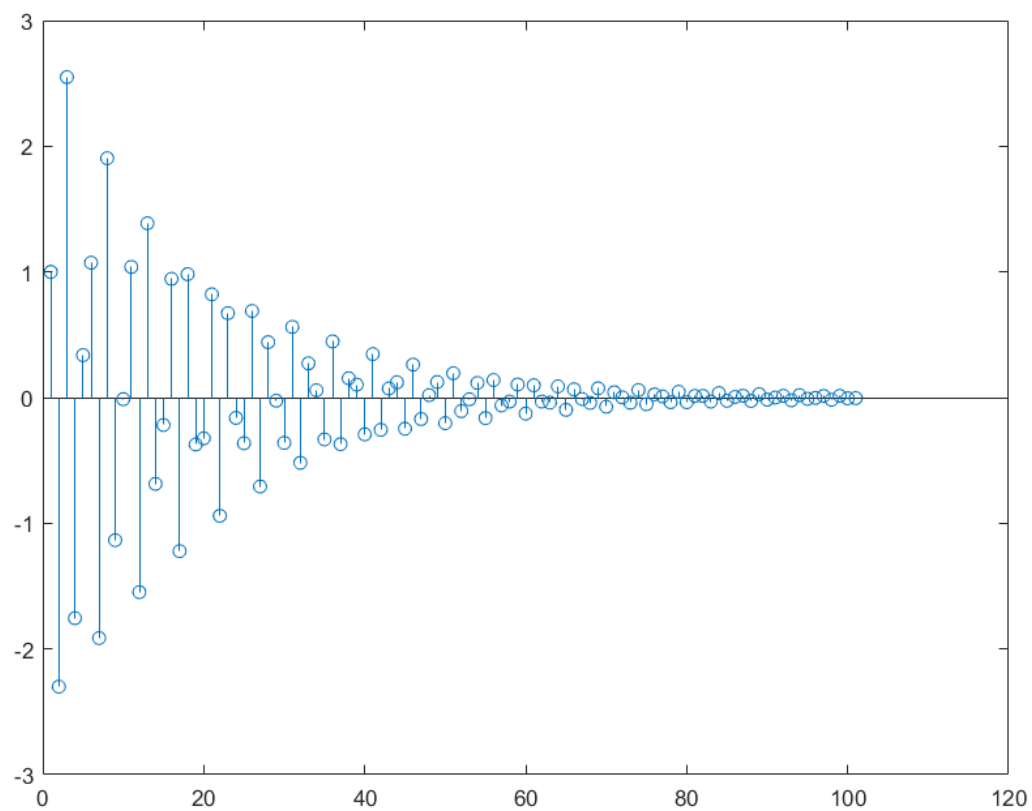
### 4.1 Exemplu

- se consideră sistemul caracterizat prin funcția de transfer din (3)
- se calculează răspunsul la impuls al sistemului (cu funcția impz)
- se calculează energia răspunsului la impuls
- se afișează pozițiile polilor și zerourilor în planul Z (funcția zplane)

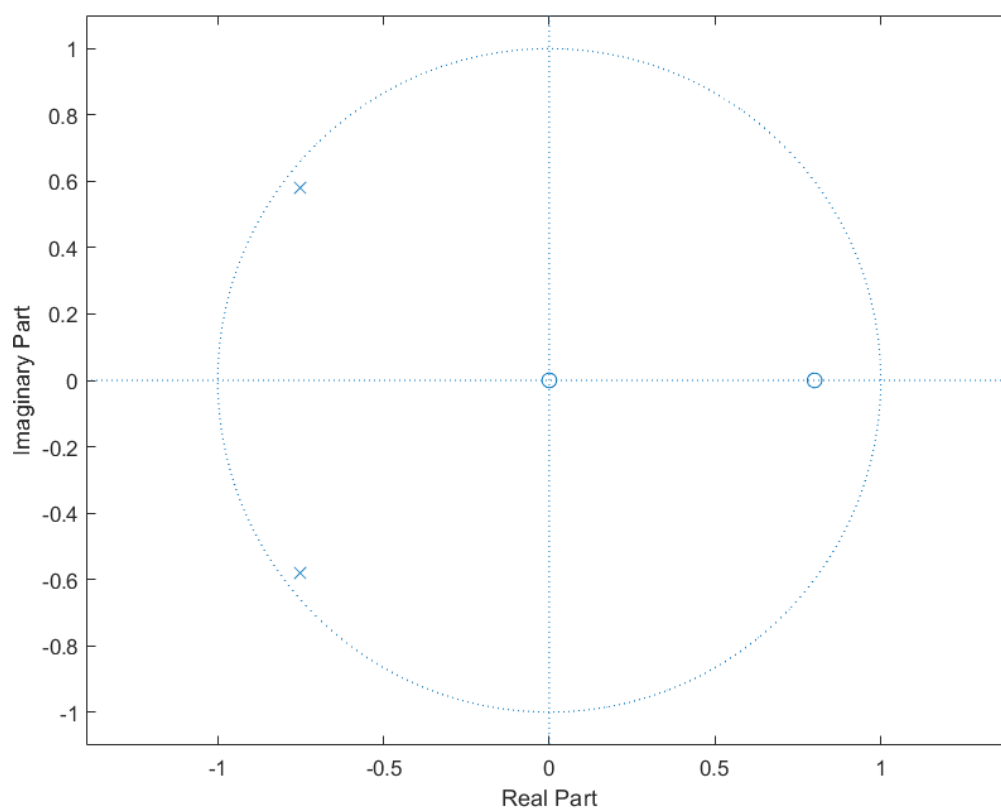
$$H(z) = \frac{1-0.8z^{-1}}{1+1.5z^{-1}+0.9z^{-2}} \quad (3)$$

### 4.2 Implementare Matlab

```
num=[1 -0.8];  
den=[1 1.5 0.9];  
n=100;  
h=impz(num,den,n+1);  
stem(h);  
figure  
zplane(num,den);
```



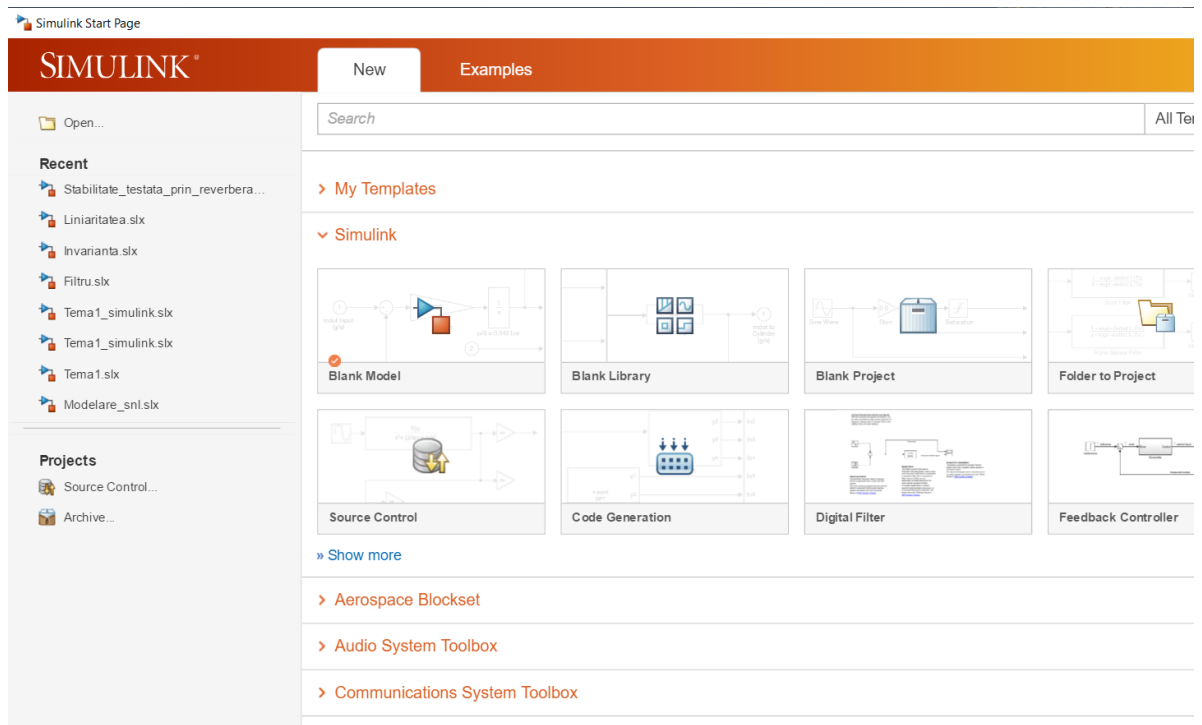
*Figura 5. Răspunsul la impuls al sistemului*



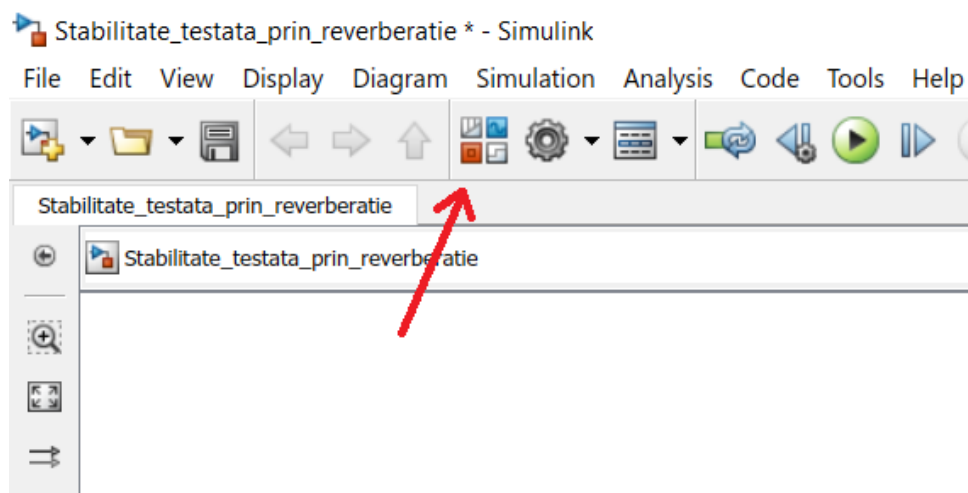
*Figura 6. Pozițiile polilor și zerourilor în planul Z*

### 4.3 Implementare Simulink

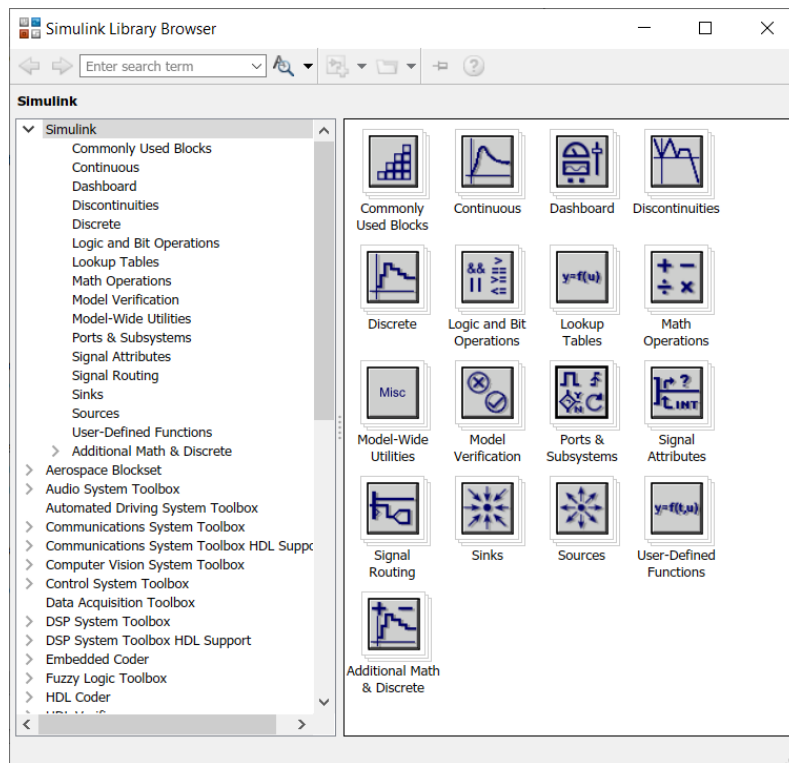
Din pagina de start Simulink, se creează un nou model tip *Blank Model*.



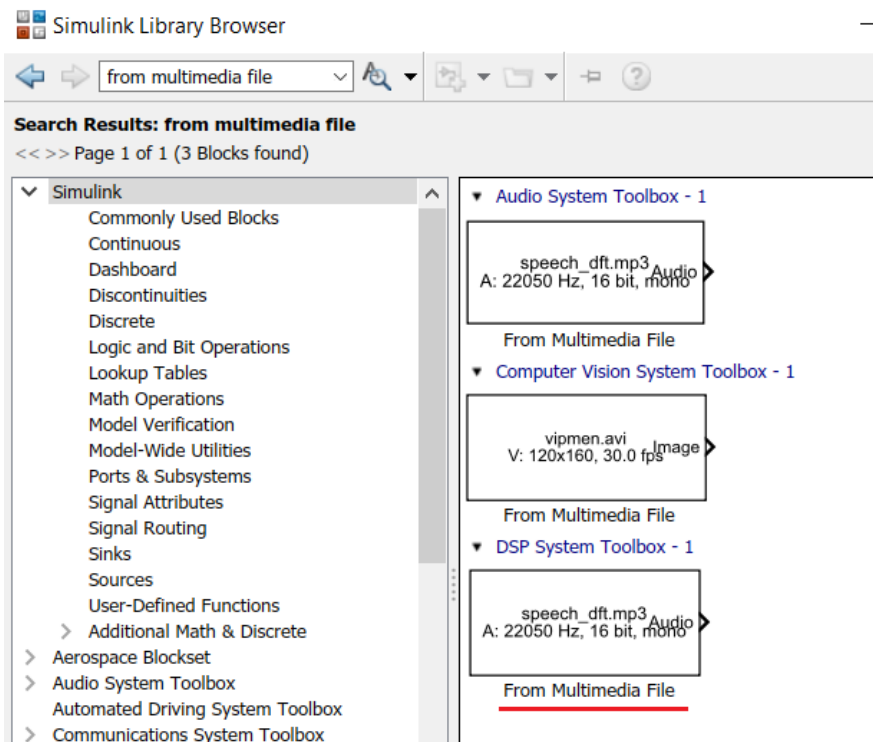
După salvarea inițială și redenumirea proiectului se deschide Library Browser-ul pentru a adăuga componentele necesare simulării.



Blocurile necesare se pot căuta ușor în funcție de categorie, sau după nume folosind caseta text din colțul stânga-sus.



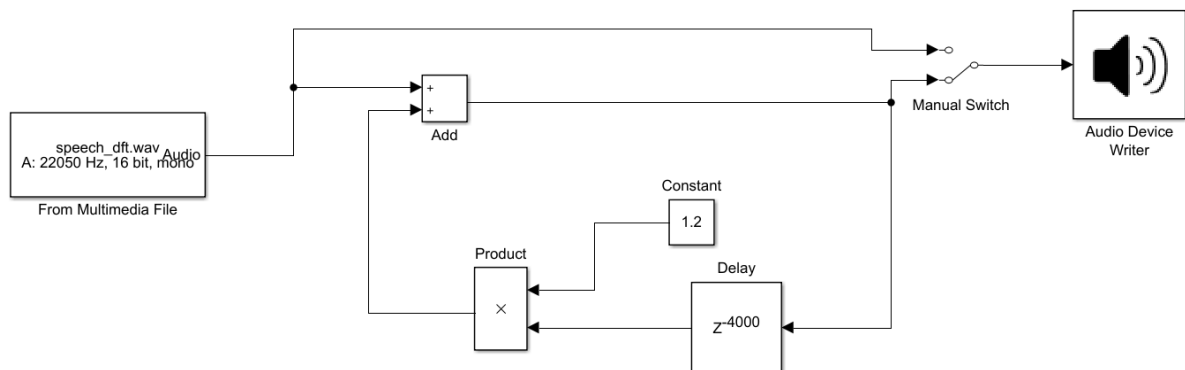
Pentru adăugarea unui bloc necesar în spațiul de lucru se face click dreapta pe acesta > *Add block to model*, sau un simplu drag & drop.





Vom adăuga pe rând blocurile din figura de mai jos și le vom conecta conform schemei.

- *From Multimedia File* (fie din DSP System Toolbox sau Audio System Toolbox)
- *Add*
- *Product*
- *Constant*
- *Delay* ( $z^{-1}$ )
- *Manual Switch*
- *Audio Device Writer*

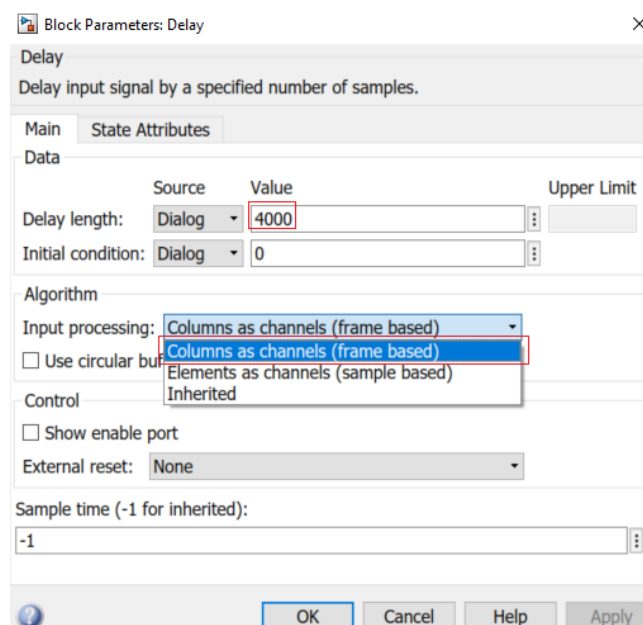


*Scurtături utile & vizualizare:*

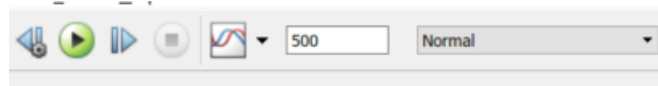
**CTRL + R** = rotirea unui bloc selectat  
**CTRL + drag & drop** pe un bloc = **copy – paste** pe locul de *release*  
**Scroll up/ down** pe workspace = **zoom in/ out**  
**Space + drag** = pan (deplasare in workspace)

Prin dublu click pe blocul *From Multimedia File* se poate selecta pentru parametrul *File Name* un alt fișier de intrare (.mp3) sau se păstrează setările implicite.

Pentru *Delay*, se setează lungimea întârzierii la 4000, iar algoritmul de procesare al input-ului va fi *frame based*.



Se alege durata maximă a simulării și se apasă *Play*.



Se obține un efect de reverberație a cărei intensitate se modifică prin varierea constantei (deci a intensității cu care semnalul întârziat, trecut prin *Delay* se adaugă la semnalul inițial, prin bucla de feedback).

Se observă că dacă factorul (constanta) este:

< 1 – atunci

> 1 – atunci

## 5. Cauzalitatea

Un sistem este **cauzal** dacă ieșirea la orice moment de timp  $t$  depinde numai de intrările anterioare momentului  $t$  și/sau de intrările la momentul de timp  $t$ . Pentru un sistem cauzal, răspunsul nu poate să preceadă excitația.

### 5.1 Exercițiu Matlab

Să se scrie un script Matlab care evidențiază un sistem cauzal care realizează medierea eșantioanelor anterioare, în scopul atenuării efectelor unui zgomot perturbator. Se presupune că zgomotul are medie nulă.

#### Indicații

- se generează o bază de timp  $n = 1:100$
- se generează un semnal sinusoidal  $x = \sin(2\pi \cdot 0.05 \cdot n)$
- se generează pe post de zgomot un semnal aleator  $z = \text{rand}(1, \text{length}(n))/2$ ;
- se calculează semnalul sumă dintre sinus și zgomot ( $s = x + z$ )
- se aplică relația de mediere:

$$y[n] = \frac{x[n-1] + x[n-2] + x[n-3]}{3} \quad (2)$$

#### Rezolvare

```
n=1:100;  
x=sin(2*pi*0.05*n);  
z=rand(1,length(n))/2;  
s=x+z;  
rez=zeros(1,length(n));
```

```

for (i=4:100)
    rez(i)=(s(i-3)+s(i-2)+s(i-1))/3;
end

figure
subplot(311),plot(n,x), grid on, title ('Semnal initial
(x)')
subplot(312),plot(n,s), grid on, title ('Semnal cu zgomot')
subplot(313),plot(n,rez), grid on, title ('Semnal cu zgomot
atenuat prin mediere')

```

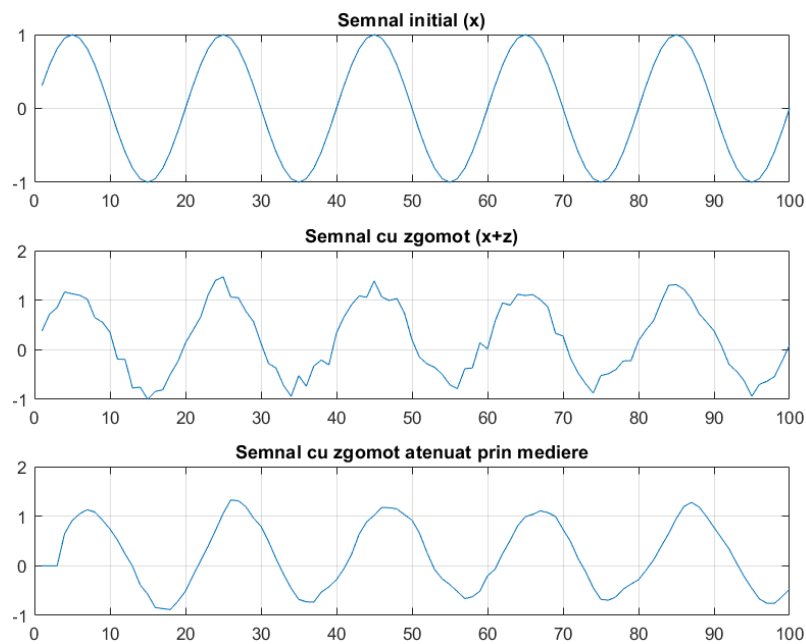


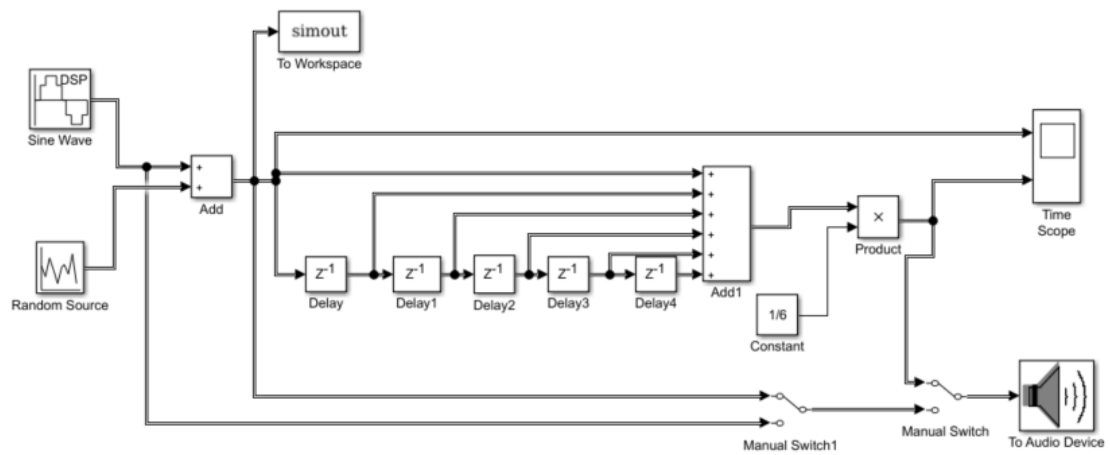
Figura 4. Rezultat exercițiu 3.1

## 5.2 Exemplu model Simulink

Vom adăuga pe rând blocurile de mai jos și le vom conecta conform schemei.

- Sine Wave (DSP)
- Random Source
- Add
- Product
- Constant
- Delay ( $z^{-1}$ )
- Manual Switch
- Audio Device Writer

- *Scope*



Pentru blocurile de tip *Delay* păstrăm setările implicite, iar pentru *Sine Wave* și și *Random Source*, configurăm parametrii:

Block Parameters: Delay

Delay

Delay input signal by a specified number of samples.

Main

State Attributes

Data

	Source	Value	Upper Limit
Delay length:	Dialog	1	
Initial condition:	Dialog	0	

Algorithm

Input processing: Columns as channels (frame based)

☐ Use circular buffer for state

Control

☐ Show enable port

External reset: None

Sample time (-1 for inherited):


-1


OK

Cancel

Help

Apply

Block Parameters: Sine Wave



Sine Wave (mask) (link)

Output samples of a sinusoid. To generate more than one sinusoid simultaneously, enter a vector of values for the Amplitude, Frequency, and Phase offset parameters.

MainData Types

Amplitude: 1

Frequency (Hz): 1000

Phase offset (rad): 0

Sample mode: Discrete


Output complexity: Real

Computation method: Trigonometric fcn

Sample time: 1/16000

Samples per frame: 1024

Resetting states when re-enabled: Restart at time zero

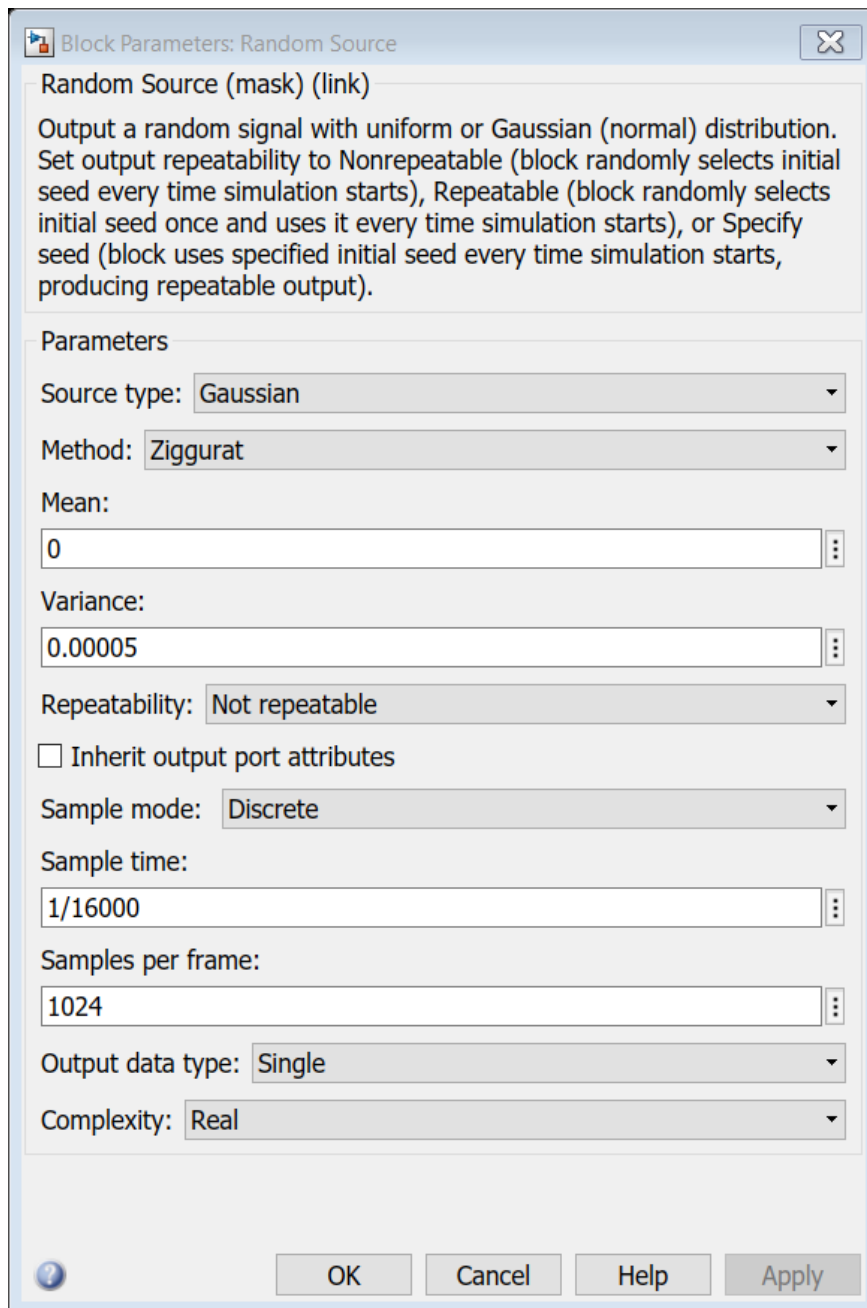


OK

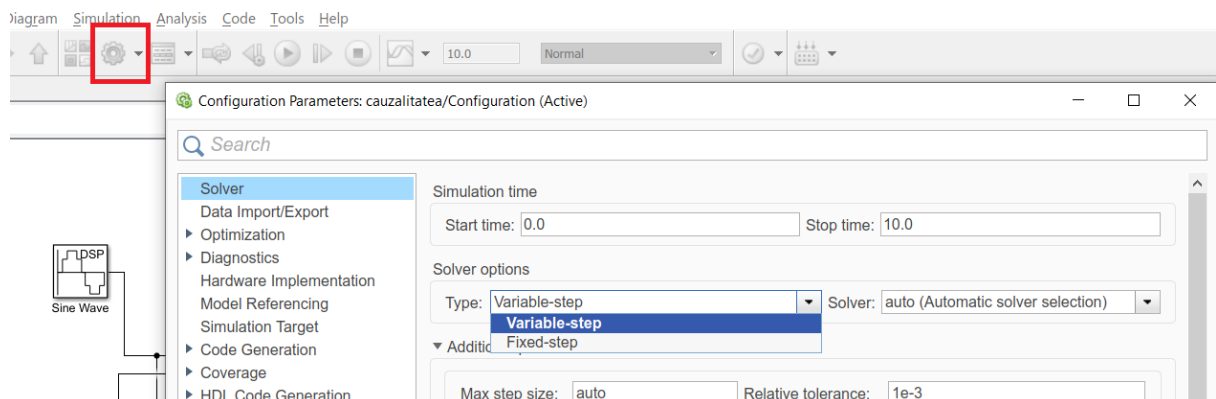
Cancel

Help

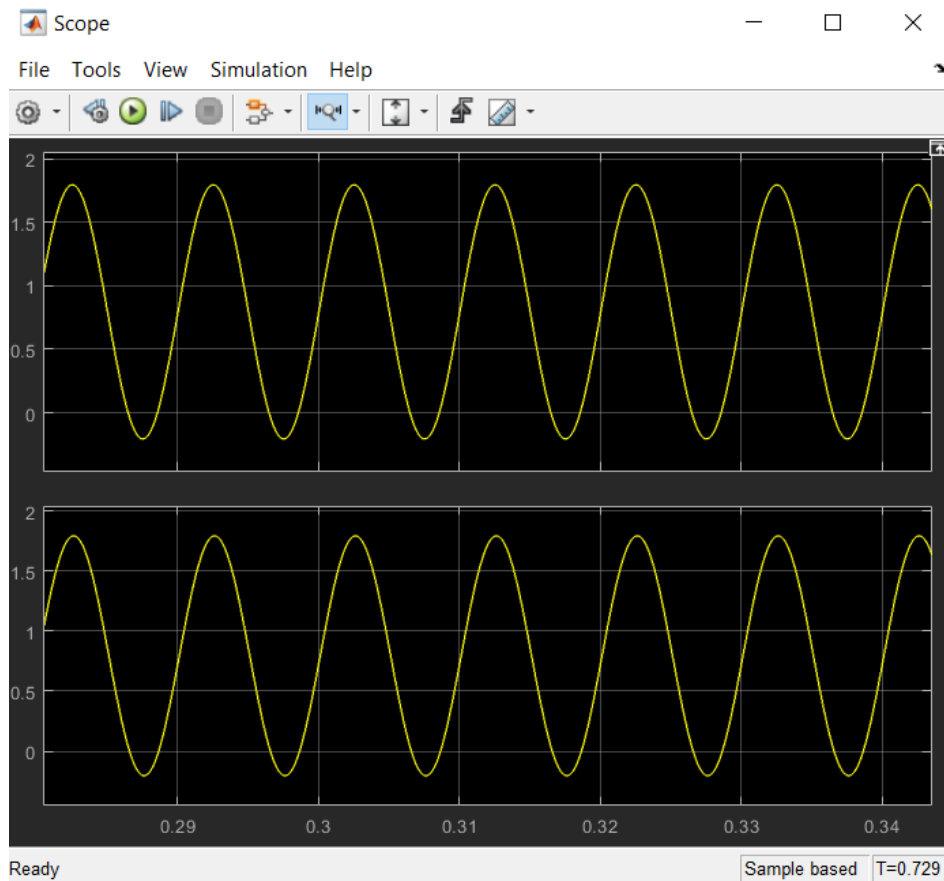
Apply



Din fereastra de configurare a parametrilor globali alegem pentru Solver tipul **Variable-step** și păstrăm restul opțiunilor implicite.



După pornirea simulării, în difuzor se poate auzi un ton generat de valorile anterioare și valoarea curentă a blocurilor Sine Wave și Random Source, iar pe Scope se va trasa graficul undei.



## 6. Liniaritatea

Un sistem este liniar dacă răspunsul său la o combinație liniară de semnale (vezi relația 4) este combinația liniară a răspunsurilor sistemului la fiecare semnal în parte (vezi (5)).

$$x[n] = \sum_{(i)} a_i \cdot x_i[n] \quad (4)$$

$$y[n] = S\{x[n]\} = S\left\{\sum_{(i)} a_i \cdot x_i[n]\right\} = \sum_{(i)} a_i S\{x_i[n]\} = \sum_{(i)} a_i y_i[n] \quad (5)$$



## 6.1 Exercițiu Matlab

Vom demonstra că sistemul numeric reprezentat prin funcția de transfer  $H(z)$  din (6) este liniar.

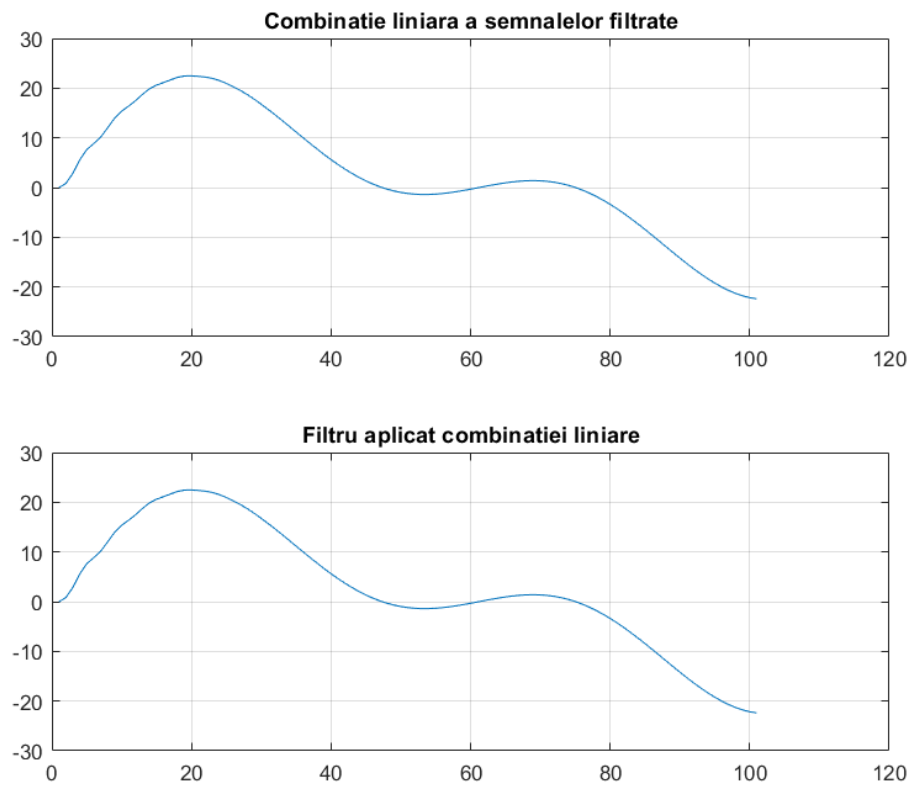
$$H(z) = \frac{2.2403 + 2.4908z^{-1} + 2.2403z^{-2}}{1 - 0.4z^{-1} + 0.75z^{-2}} \quad (6)$$

### Indicații

- se consideră sistemul caracterizat prin funcția de transfer din (6)
- se inițializează variabilele num și den cu valorile corespunzătoare coeficienților din relația  $H(z)$
- se generează două semnale sinus
- se calculează semnalul obținut prin combinația liniară a celor două semnale, pentru  $a_1 = 2$  și  $a_2 = 3$
- se calculează răspunsul sistemului la combinația liniară a semnalelor de intrare
- se calculează răspunsul sistemului pentru cele două semnale sinus (se folosește funcția filter) separat și se calculează combinația liniară a răspunsului pentru aceleași valori ale lui  $a_i$
- se calculează și se afișează opțional diferența dintre răspunsuri

### Rezolvare

```
n=0:1:10;  
x1=sin(pi/3*n);  
x2=sin(pi/6*n);  
num=[ 2.2403 2.4908 2.2403];  
den=[1 -0.4 0.75];  
ic=[0 0];  
  
y1=filter(num,den,x1,ic);  
y2=filter(num,den,x2,ic);  
  
y=2*y1+3*y2;% aplic combinatia liniara iesirii filtrului  
x3=2*x1+3*x2;  
  
y3=filter(num,den,x3,ic);% aplic filtrul combinatiei liniare  
% a semnalelor initiale  
  
subplot(2,1,1);  
plot(y), grid on, title ('Comb. lin. a semnalelor filtrate');  
subplot(2,1,2);  
plot(y3), grid on, title ('Filtru aplicat combinatiei lin.');
```



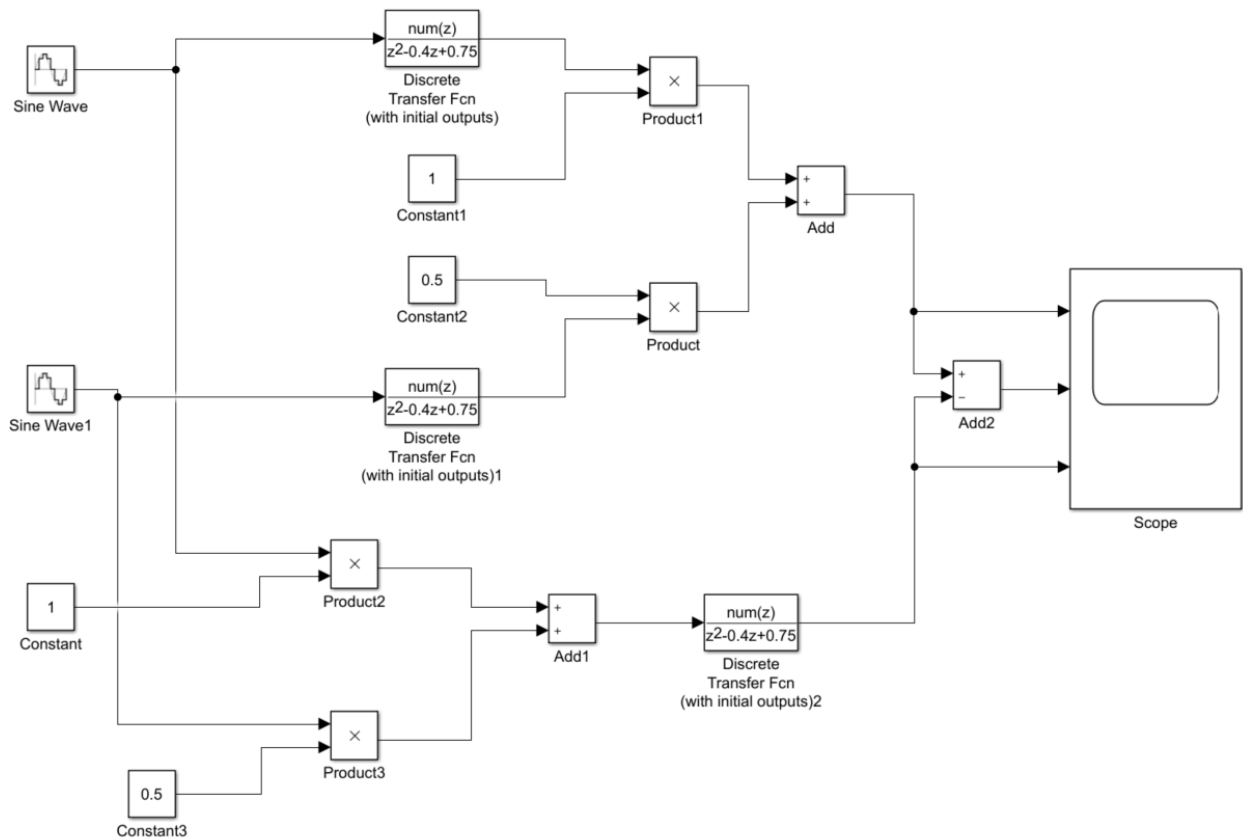
*Figura 7. Rezultatul exercițiului 6.1*

Observăm că pentru un sistem liniar, cele două grafice sunt identice. Se poate trasa graficul diferenței dintre  $y$  și  $y_3$ , care ar trebui să aibă valoarea 0 constant.

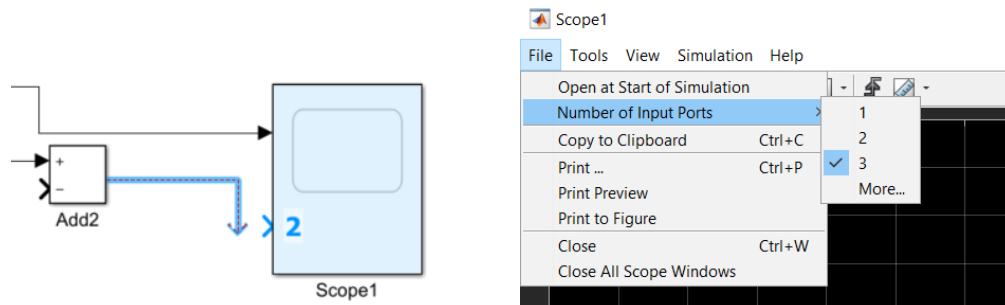
## 6.2 Exemplu model Simulink

Vom adăuga pe rând blocurile de mai jos și le vom conecta conform schemei.

- *Sine Wave* (**nu** Sine Wave Function)
- *Add*
- *Product*
- *Constant*
- *Discrete Transfer Fcn (with initial outputs)* – categoria Simulink Extras > Additional Discrete
- *Scope*

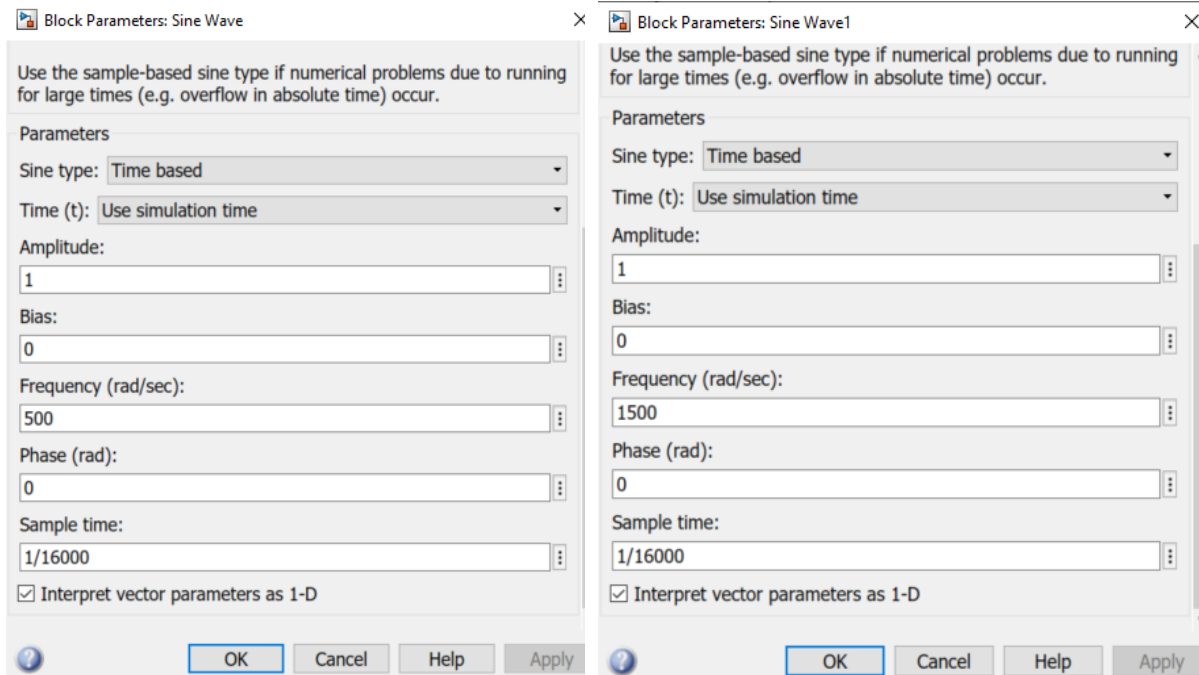


Adăugarea de intrări în *Scope* se face prin apropierea unei conexiuni noi sau din meniul acestuia.

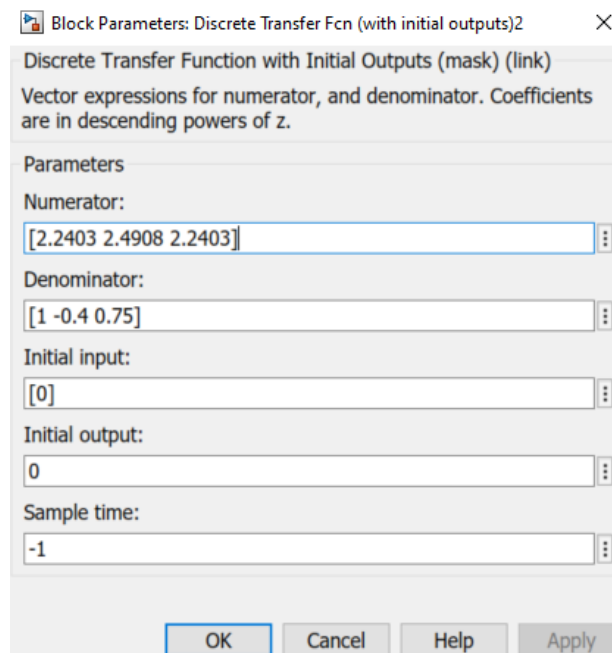


**Observație:** Blocul Add2 va fi setat să calculeze diferența celor două semnale de intrare.

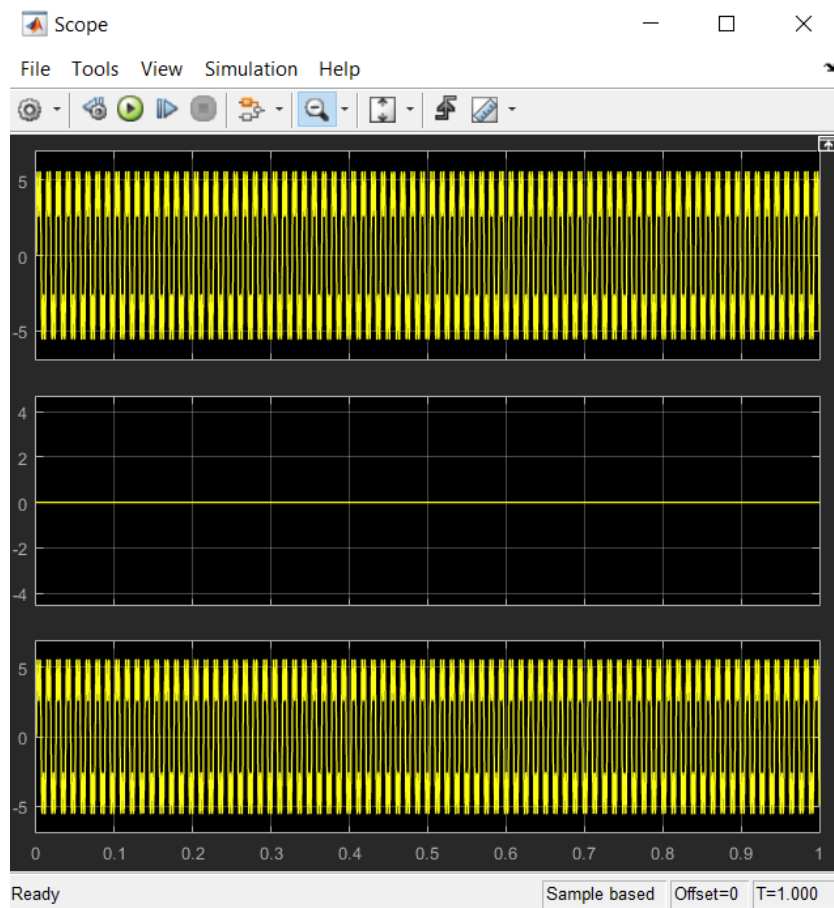
Pentru blocurile *Sine Wave* și *Sine Wave1* se vor seta următorii parametrii:



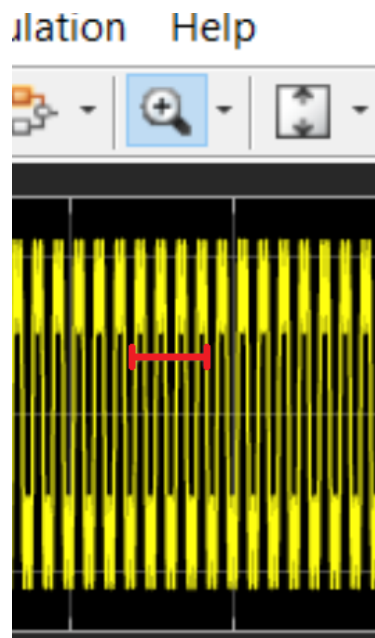
Pentru toate blocurile Discrete Transfer Fcn se aleg aceleași valori ca în exercițiul anterior:

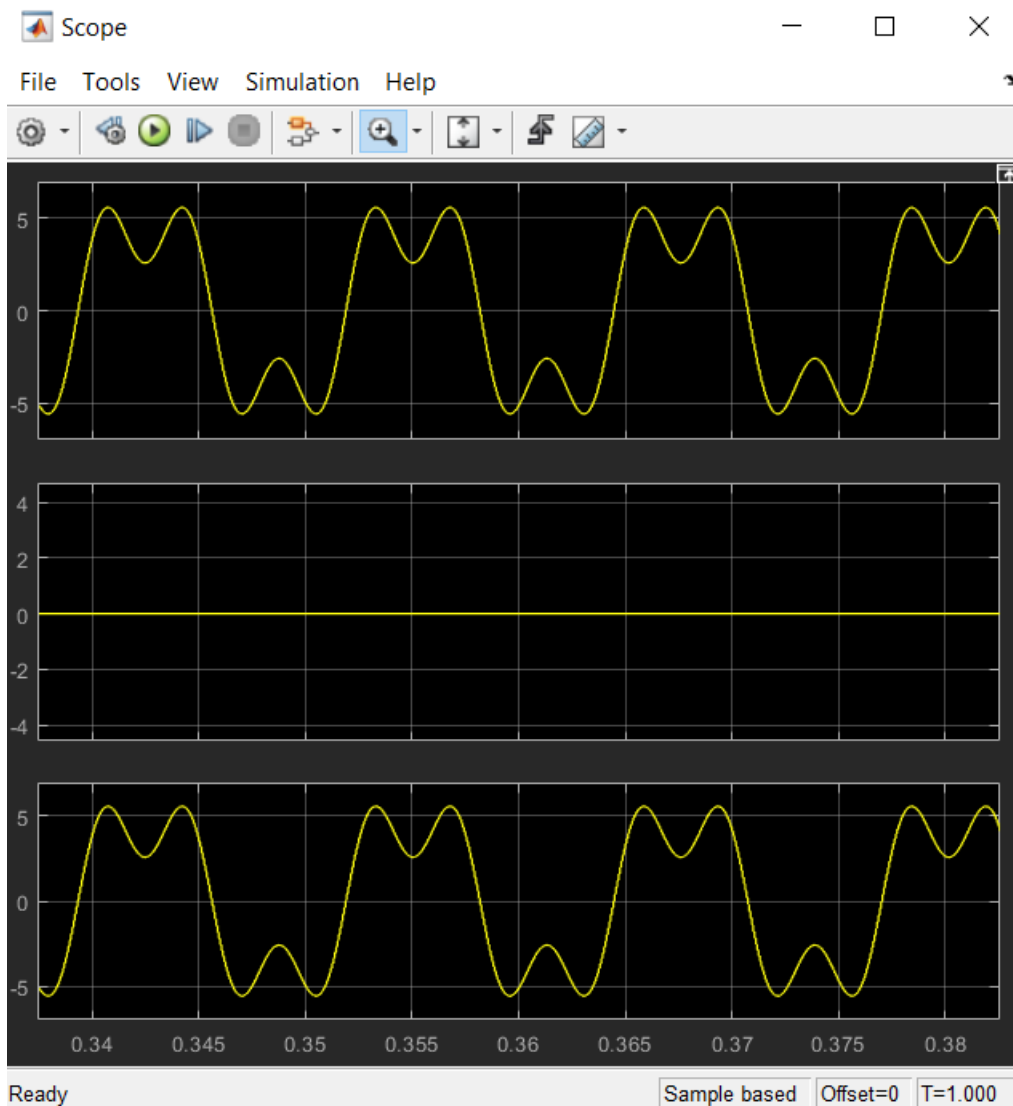


Se alege o durată maximă (de ex. 1 sec) și se pornește simularea. Făcând dublu click pe Scope se pot observa semnalele generate și, în graficul din mijloc, diferența acestora care ar trebui să fie nulă datorită liniarității sistemului modelat.



Putem folosi instrumentul *zoom in* pe pentru o vizualizare mai apropiată a sinusoidei.





## 7. Invarianța în timp

Un sistem este invariant în timp dacă răspunsul la o intrare dată nu depinde de momentul apariției acestei intrări (vezi relația (7)).

$$\begin{aligned} y[n] &= S\{x[n]\} \\ y[n-k] &= S\{x[n-k]\} \end{aligned} \quad (7)$$

### 7.1 Exercițiu Matlab

Să se pună în evidență în Matlab invarianța unui sistem.

#### Instrucțiuni

- se consideră sistemul caracterizat prin funcția de transfer din (6 bis)
- se inițializează variabilele num și den cu valorile corespunzătoare coeficienților din relația  $H(z)$

- se generează un semnalul de intrare sinusoidal  $x_1$
- se generează un semnal  $x_2$  reprezentând semnalul intrare, dar întârziat cu 10 eșantioane
- se calculează răspunsul sistemului, pe rând, la cele două semnale  $x_1$  și  $x_2$  în variabilele  $y_1$  și  $y_2$
- se compară rezultatele

$$H(z) = \frac{2.2403 + 2.4908z^{-1} + 2.2403z^{-2}}{1 - 0.4z^{-1} + 0.75z^{-2}}$$

(6 bis)

### Rezolvare

```
n=0:40;
x1=sin(pi/3*n);
num=[ 2.2403 2.4908 2.2403];
den=[1 -0.4 0.75];
ic=[0 0];

y1=filter(num,den,x1,ic); %aplic filtrul pt y1
x2=[zeros(1,10) x1]; % x2 = x1 intarziat cu 10 unitati
y2=filter(num,den,x2,ic); %aplic filtrul pt x2

dif=y1-y2(11:length(y2)); %calculez diferenta celor 2 semnale
subplot(3,1,1);
plot(y1), grid on, title ('Semnal y1 (neintarziat)');

subplot(3,1,2);
plot(y2), grid on, title ('Semnal y2 (intarziat)');

subplot(3,1,3); %diferenta va fi nula pt un sistem invar. in timp
plot(dif), grid on, title ('Diferenta y2 - y1');
```

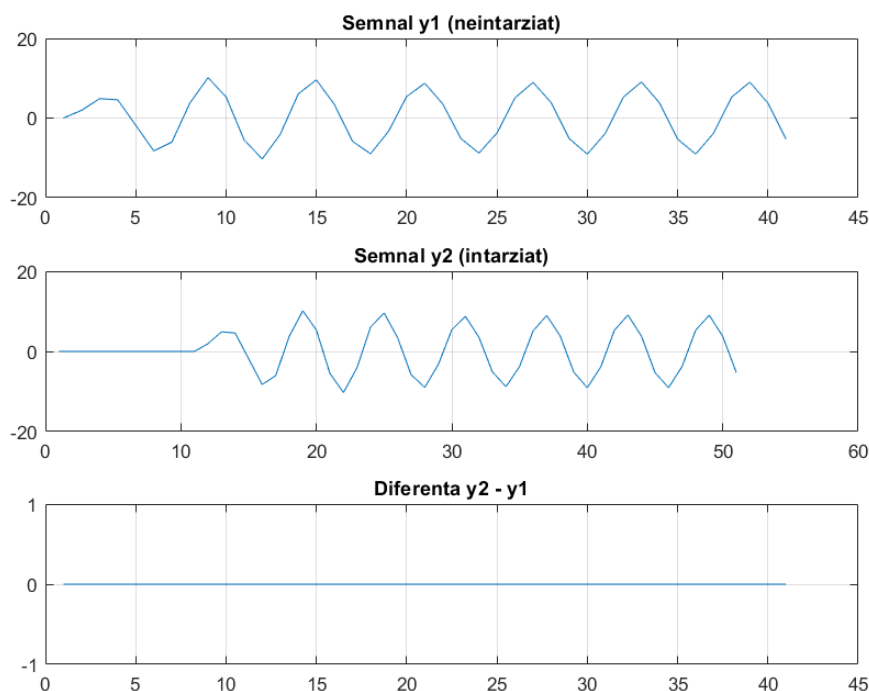
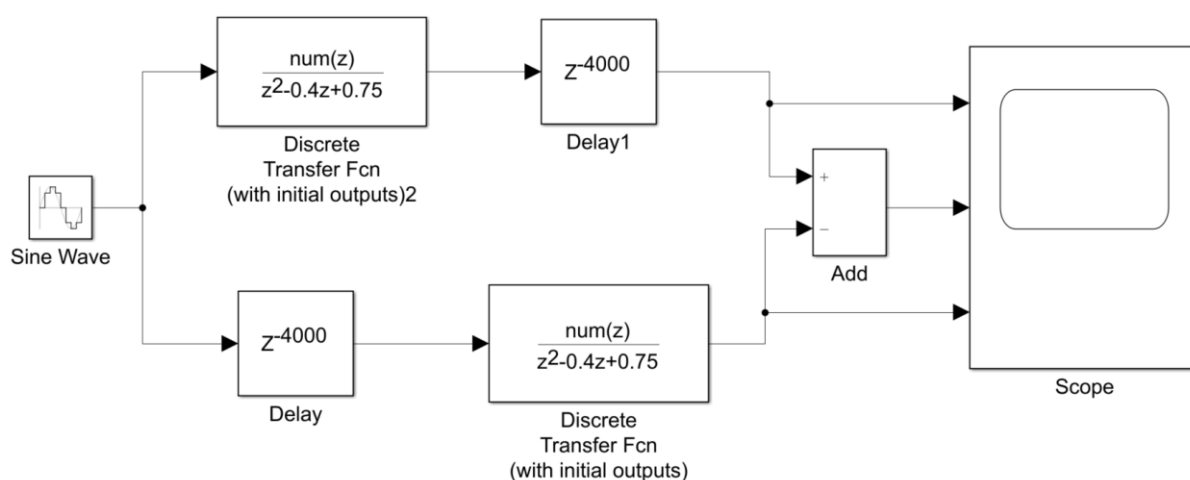


Figura 8. Rezultat exercițiu 7.1

## 7.2 Exemplu model Simulink

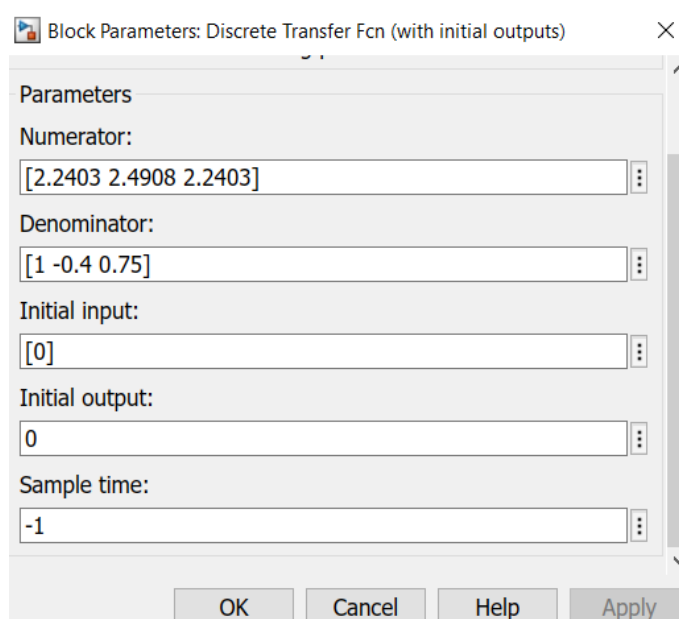
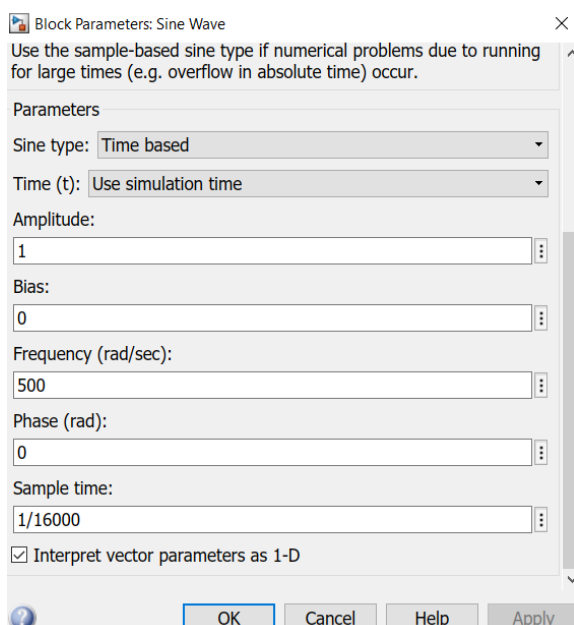
Vom adăuga pe rând blocurile de mai jos și le vom conecta conform schemei.

- *Sine Wave* (**nu** Sine Wave Function)
- *Delay* ( $z^{-1}$ )
- *Discrete Transfer Fcn (with initial outputs)* – categoria Simulink Extras > Additional Discrete
- *Add*
- *Scope*

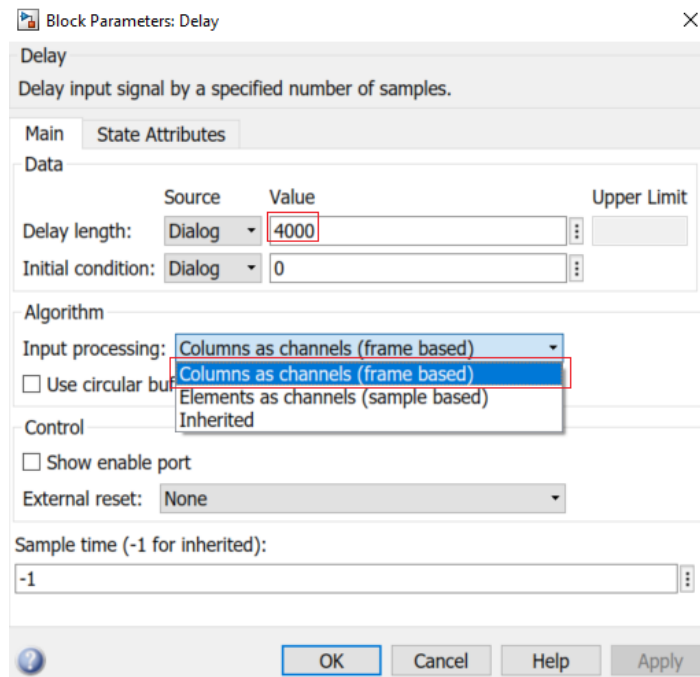


Parametrul *List of signes al blocului sumator* va fi “+ –”, pentru a calcula diferența celor două semnale de intrare.

Valorile pentru *Sine Wave* și *Discrete Transfer Fcn* se setează analog exercițiilor anterioare:







Similar punctului 5.2, diferența nulă a celor două semnale sinusoidale demonstrează invarianța în timp a sistemului simulat: fie că aplicăm o transformare la un moment  $t_0$  sau  $t_0 + \Delta t$  unui semnal de intrare, rezultatul va fi același.

