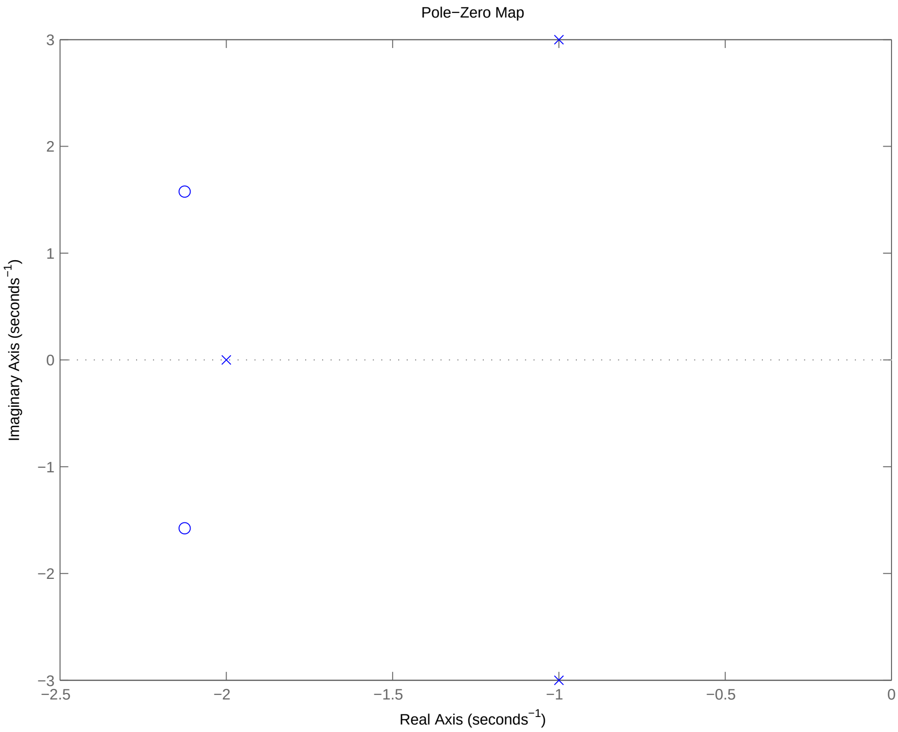
Linear systems in Matlab

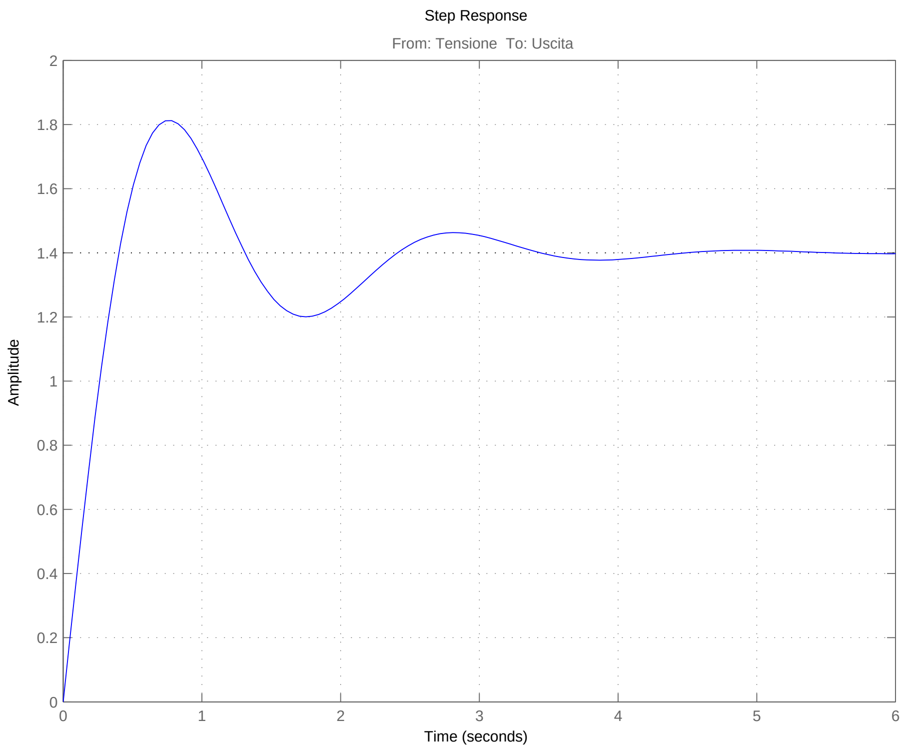
- Numerical example in Matlab: `LTI_Systems_in_Matlab.m`

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   LTI_Systems_in_Matlab.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all; clc
echo on
% Let us consider the time-continuous (TC) linear system (LS)
% characterized by the following matrices MA, MB and MC
% MA=[ 2     3     3;
%     -4    -5    -3;
%     -2     1    -1];
% MB=[  3; -2;  1];
% MC=[  2   1   0];
echo off
MA=[ 2     3     3;
    -4    -5    -3;
    -2     1    -1];
MB=[  3; -2;  1];
MC=[  2   1   0];
echo on
pause; clc;             % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A time-continuous linear system can be defined in Matlab using the
% command "ss" (state space). Using the command
SYS=ss(MA,MB,MC,0)
----- Matlab output ----------------------------------------------------------------
SYS =

  a =
        x1  x2  x3
   x1    2   3   3
   x2   -4  -5  -3
   x3   -2   1  -1

  b =
        u1
   x1    3
   x2   -2
   x3    1

  c =
        x1  x2  x3
   y1    2   1   0

  d =
       u1
   y1   0
--------------------------------------------------------------------------
% the variable SYS is given the meaning of dynamic linear system
% characterized by the three matrices MA, MB and MC (the last matrix is MD=0).
pause; clc;             % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% It is possible to assign a name to all the input, state and output
%  variables using the command "set"
set(SYS,'InputName','Voltage')
set(SYS,'StateName',[   'Current ';  'Velocity ';      'Position'])
set(SYS,'OutputName','Output')
SYS
```

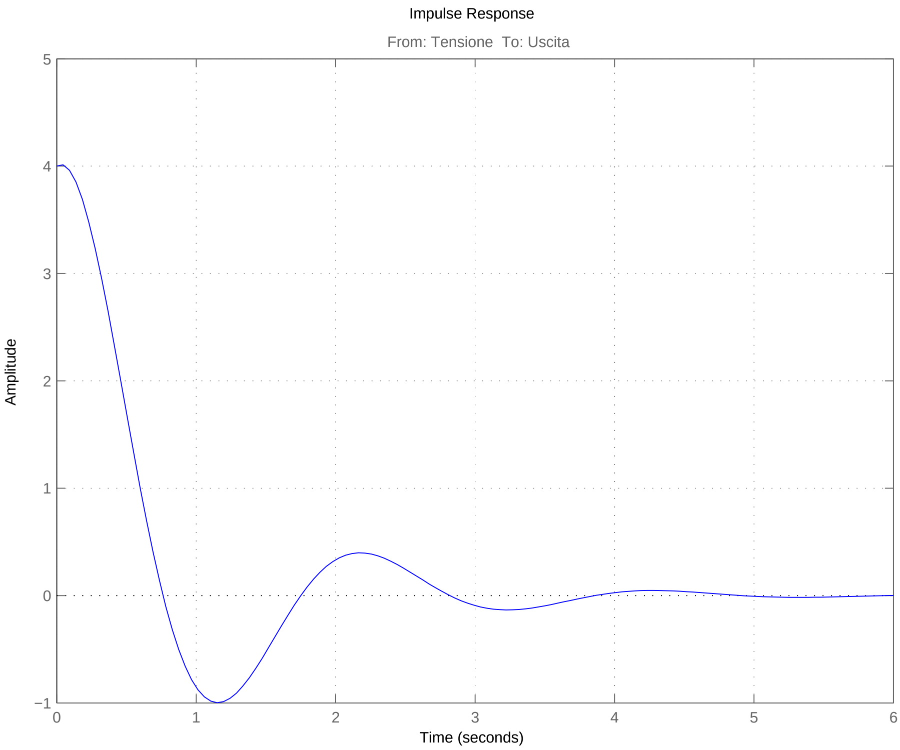
```
----- Matlab output -------------------------------------------------------------------
SYS =

  a =
              Current   Velocity   Position
   Current          2          3          3
   Velocity        -4         -5         -3
   Position        -2          1         -1

  b =
             Voltage
   Current          3
   Velocity        -2
   Position         1

  c =
           Current   Velocity   Position
   Output        2          1          0

  d =
          Voltage
   Output       0
-----------------------------------------------------------------------
pause; clc;               % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The transfer function of system G(s) can be obtained using the command "tf"
GS=tf(SYS)
----- Matlab output -------------------------------------------------------------------
GS =

  From input "Voltage" to output "Output":
     4 s^2 + 17 s + 28
  ----------------------
  s^3 + 4 s^2 + 14 s + 20
-----------------------------------------------------------------------
pause; clc;               % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The poles of the system $G(s)$
pole(GS)
----- Matlab output -------------------------------------------------------------------
ans =

  -1.0000 + 3.0000i
  -1.0000 - 3.0000i
  -2.0000
-----------------------------------------------------------------------
% are equal to the poles of the system SYS
pole(SYS)
----- Matlab output -------------------------------------------------------------------
ans =

  -1.0000 + 3.0000i
  -1.0000 - 3.0000i
  -2.0000
-----------------------------------------------------------------------
% and equal to the eigenvalues of matrix MA
roots(poly(MA))
----- Matlab output -------------------------------------------------------------------
ans =

  -1.0000 + 3.0000i
  -1.0000 - 3.0000i
  -2.0000
-----------------------------------------------------------------------
pause; clc;               % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The position of poles and zeros of system SYS can be plotted
% using the following command
figure(1); clf
pzmap(SYS)
----- Matlab output -------------------------------------------------------------------
```



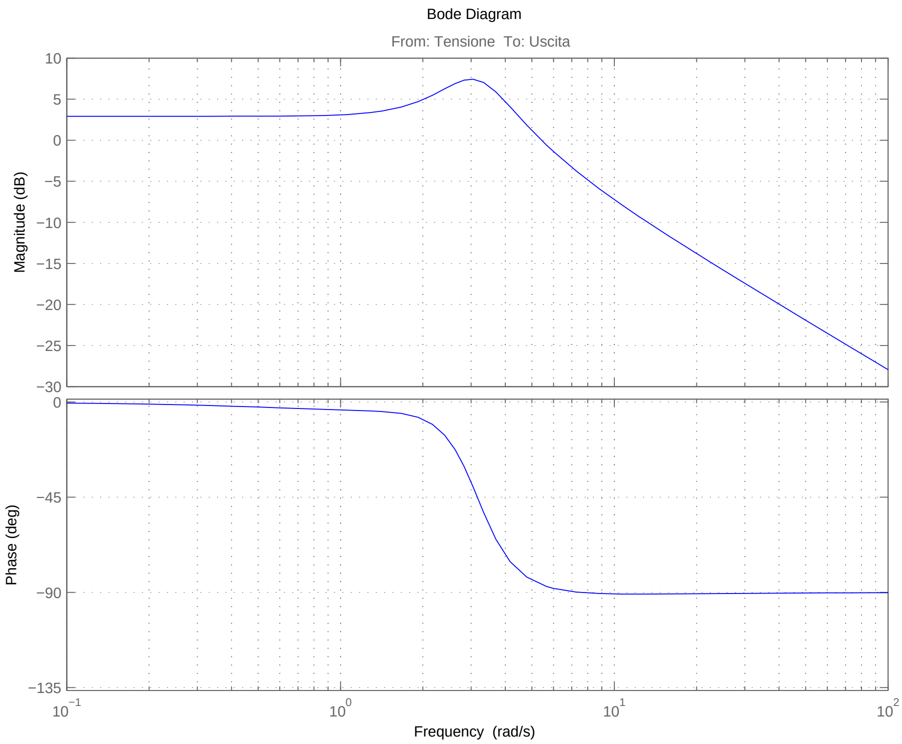

```
---------------------------------------------------------------------------
pause;                     % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The step response is obtained using the following command
figure(1); clf
step(SYS); grid on
----- Matlab output ---------------------------------------------------------------
```



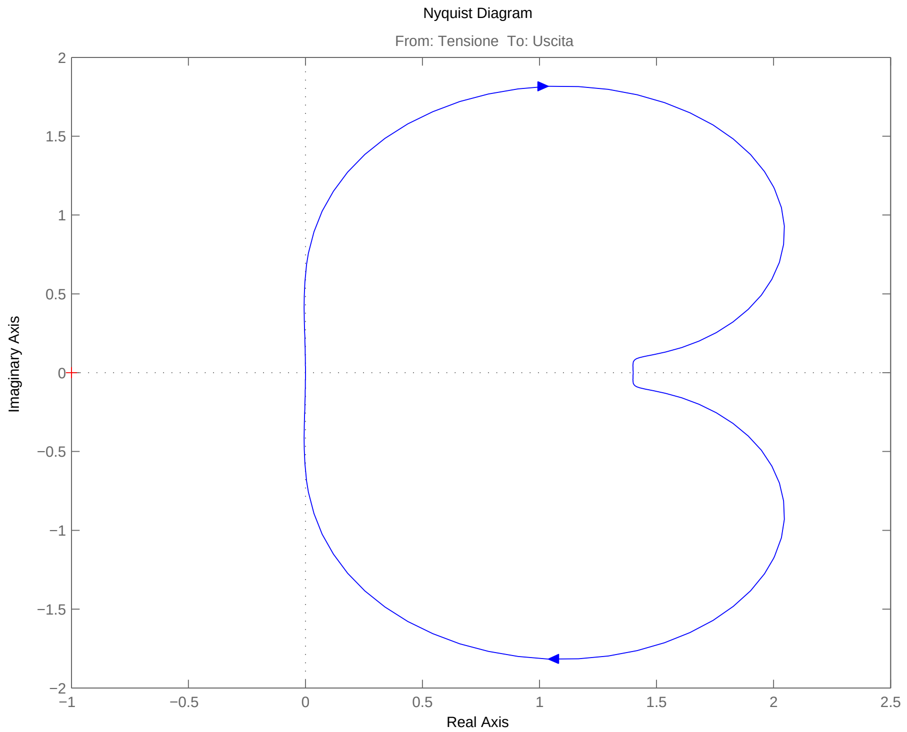

```
---------------------------------------------------------------------------
pause;                     % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The impulse response is obtained using the command
     figure(1); clf
impulse(SYS); grid on
----- Matlab output ---------------------------------------------------------------
```




```
-----------------------------------------------------------------------
pause;                       % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The Bode diagrams are obtained using the command:
figure(1); clf
bode(SYS); grid on
----- Matlab output ----------------------------------------------------
```



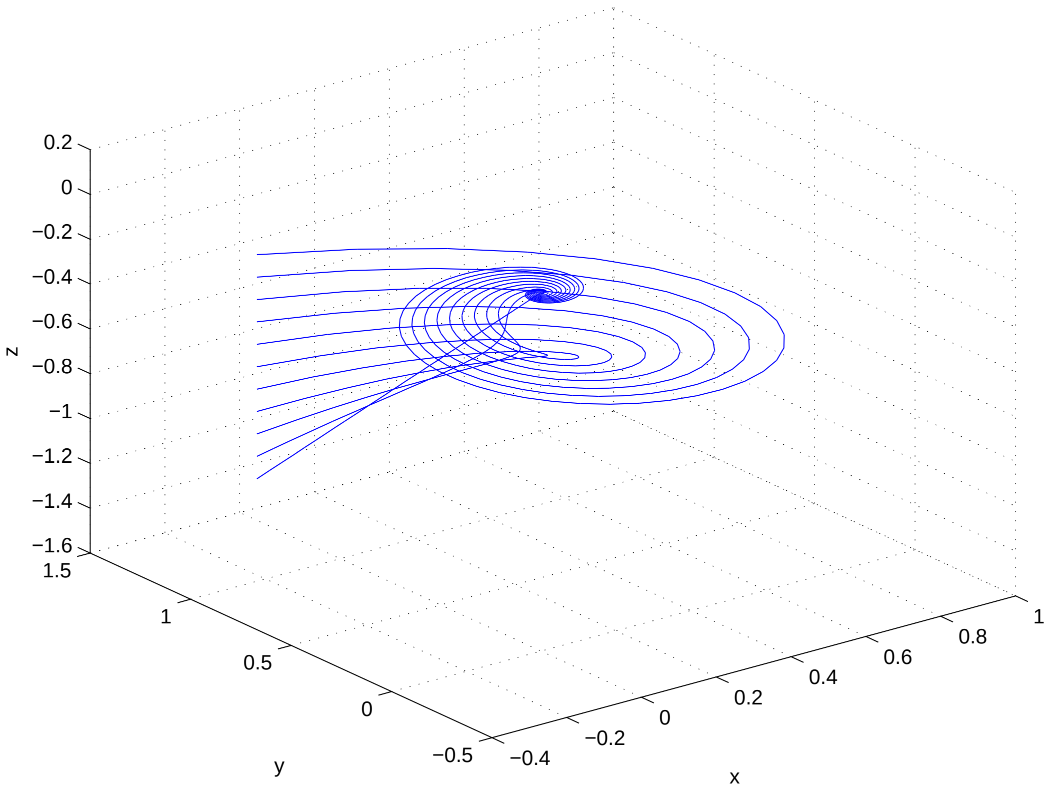

```
-----------------------------------------------------------------------
pause;                       % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The Nyquist diagrams are obtained using the command:
figure(1); clf
nyquist(SYS)
----- Matlab output ----------------------------------------------------
```




```
--------------------------------------------------------------------------
pause;                    % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
set(figure(1),'Visible','off')
clc
% The eigenvalues and the eigenvectors of matrix MA can be obtained
% using the following command
[V,D] = eig(MA)
% The first two eigenvectors (that is the first two columns of matrix V)
% are complex coniugate because they are the eigenvectors corresponding
% to the two complex conjugate eigenvalues. V is the transformation
% matrix which diagonalize the matrix $MA$
inv(V)*MA*V
% Deleting  the small computation errors one obtains:
approx(inv(V)*MA*V)
----- Matlab output ---------------------------------------------------------------------
% The eigenvalues and the eigenvectors of matrix MA can be ontained
% using the following command
[V,D] = eig(MA)

V =

   0.5774              0.5774             -0.0000
  -0.5774 - 0.0000i  -0.5774 + 0.0000i  -0.7071
  -0.0000 + 0.5774i  -0.0000 - 0.5774i   0.7071


D =

  -1.0000 + 3.0000i        0                  0
        0            -1.0000 - 3.0000i        0
        0                  0            -2.0000

% The first two eigenvectors (the first two columns of matrix V)
% are complex conjugate because they are the eigenvectors corresponding
% to the complex conjugate eigenvalues. V is the transformation
% matrix which diagonalize matrix $MA$
inv(V)*MA*V

ans =

  -1.0000 + 3.0000i  -0.0000 + 0.0000i  -0.0000 - 0.0000i
  -0.0000 - 0.0000i  -1.0000 - 3.0000i  -0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 - 0.0000i  -2.0000 + 0.0000i
```

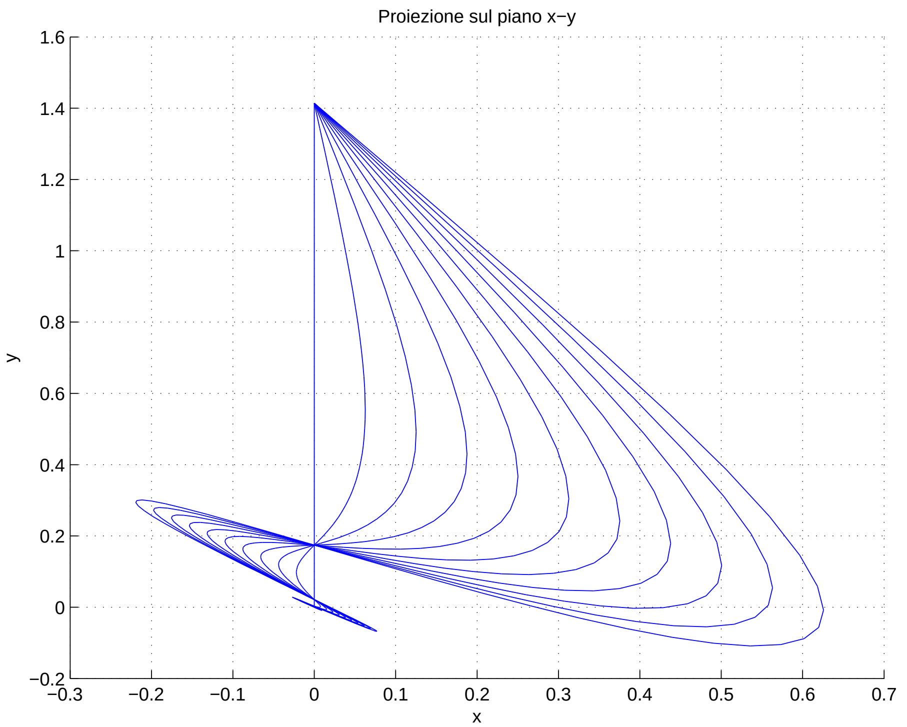
```
% Deleting the small computation errors:
approx(inv(V)*MA*V)

ans =

  -1.0000 + 3.0000i        0                  0
        0            -1.0000 - 3.0000i        0
        0                  0             -2.0000
-------------------------------------------------------------------------
pause; clc;              % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Matrix MA can be transformed in the real Jordan canonical form using
% real transformation matrix: the complex conjugate eigenvalues
% are substituted by the real and the imaginary parts of one of
% the two complex coniugate eigenvectors
TM=[real(V(:,1)) imag(V(:,1)) V(:,3)]
inv(TM)*MA*TM
% Deleting the small computation errors:
approx(inv(TM)*MA*TM)
----- Matlab output ------------------------------------------------------------------
% Matrix MA can be transformed in the real Jordan canonical form, using
% a real transformation matrix: the complex conjugate eigenvectors
% are substituted by the real and imaginary parts of one of the two
% complex conjugate eigenvectors
TM=[real(V(:,1)) imag(V(:,1)) V(:,3)]

TM =

    0.5774         0   -0.0000
   -0.5774   -0.0000   -0.7071
   -0.0000    0.5774    0.7071

inv(TM)*MA*TM

ans =

   -1.0000    3.0000   -0.0000
   -3.0000   -1.0000    0.0000
    0.0000    0.0000   -2.0000

% Deleting the small computation errors:
approx(inv(TM)*MA*TM)

ans =

    -1     3     0
    -3    -1     0
     0     0    -2
-------------------------------------------------------------------------
pause; clc;              % Premi un tasto per continuare
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The same Jordan real canonical form can also be obtained
% using the following command
[CSYS,TM]=canon(SYS)
----- Matlab output ------------------------------------------------------------------
% The same Jordan real canonical form can also be obtained
% using the following command
[CSYS,TM]=canon(SYS)

CSYS =

  a =
                Current   Velocity   Position
   Current           -1          3          0
   Velocity          -3         -1          0
   Position           0          0         -2

  b =
              Voltage
   Current          2
   Velocity        -3
   Position    -1.414
```

```
  c =
              Current   Velocity   Position
   Output           0         -1    -0.7071

  d =
            Voltage
   Output         0

Continuous-time state-space model.


TM =

    1.0000    1.0000    1.0000
   -1.0000    0.0000    0.0000
   -1.4142   -1.4142   -0.0000
-------------------------------------------------------------------------
pause; clc;              % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The free evolution of the system  is obtained using the command "initial":
% X0 initial state; X and Y state and output vectors; T time vector
figure(1); clf
X00=-2*V(:,3);
dx0=[0:0.1:1];
for dx=dx0
    X0=X00+[0;0;dx];
   [Y,T,X] = initial(SYS,X0);
   plot3(X(:,1),X(:,2),X(:,3)); hold on
   echo off
end
grid; xlabel('x'); ylabel('y'); zlabel('z')
echo on
----- Matlab output ------------------------------------------------------------
```

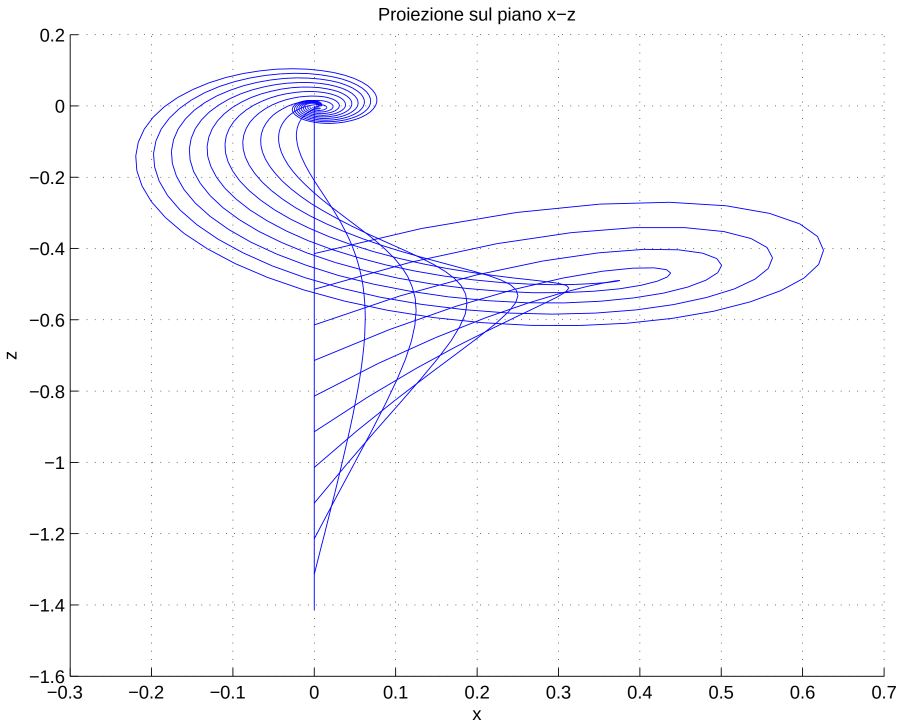



```
-------------------------------------------------------------------------
pause;                   % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Projection on the x-y plane
figure(1)
view(0,90)
title('Projection on the x-y plane')
```

```
----- Matlab output ---------------------------------------------------------------
```

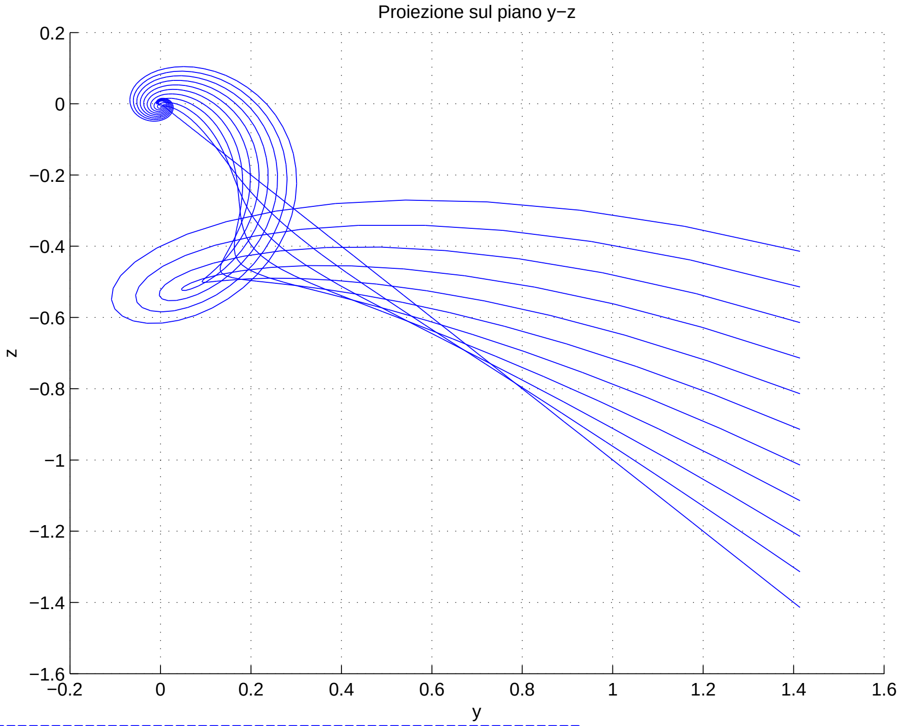



```
-------------------------------------------------------------------
pause;                         % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Projection on the x-z plane
figure(1)
view(0,0)
title('Projection on the x-z plane')
----- Matlab output ---------------------------------------------------------------
```



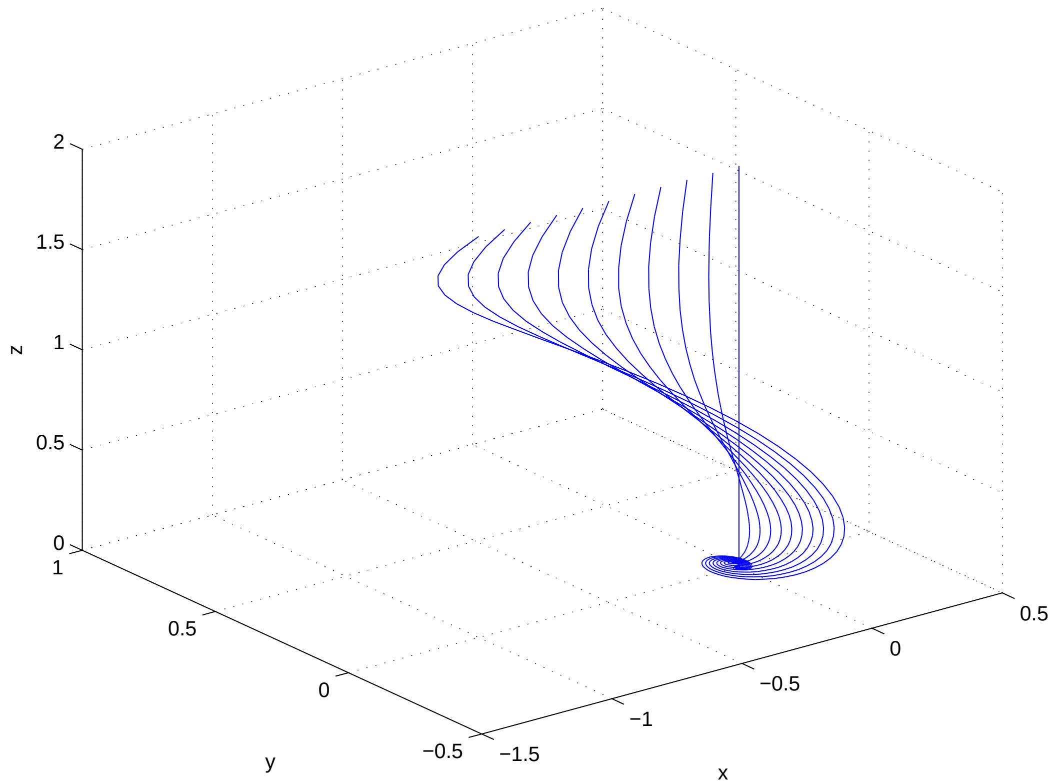

```
-------------------------------------------------------------------
pause;                         % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Projection on the y-z plane
figure(1)
view(90,0)
title('Projection on the y-z plane')
```

----- Matlab output --

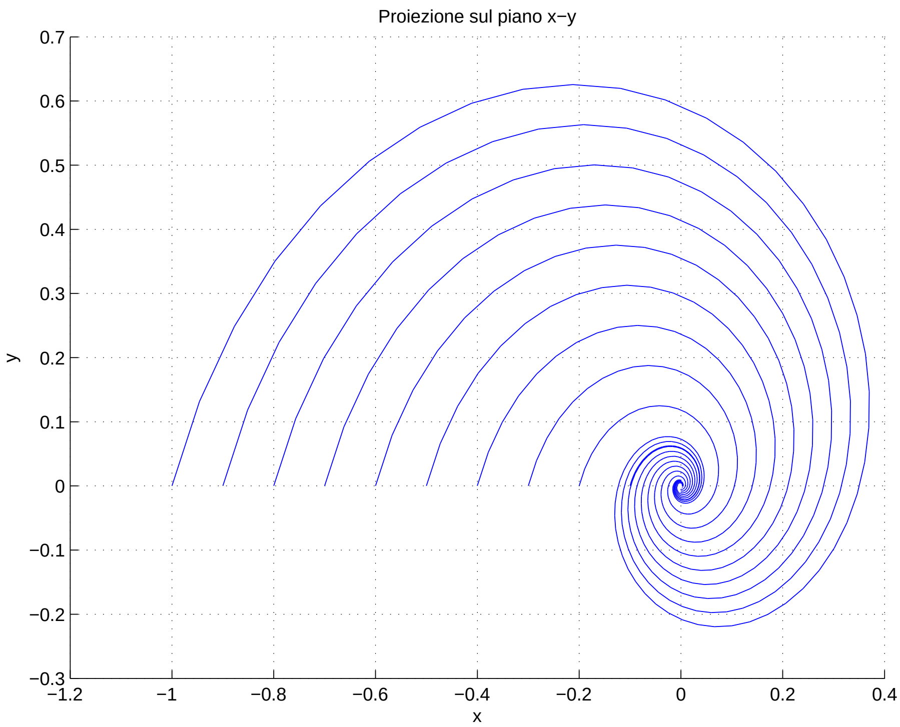



```
---------------------------------------------------------------------------
pause; clc;                % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The same trajectory in the Jordan real canonical base
% transforms as follows:
figure(1); clf
for dx=dx0
    X0=X00+[0;0;dx];
   [Y,T,X] = initial(CSYS,-TM*X0);
   plot3(X(:,1),X(:,2),X(:,3)); hold on
   echo off
end
grid; xlabel('x'); ylabel('y'); zlabel('z')
echo on
```

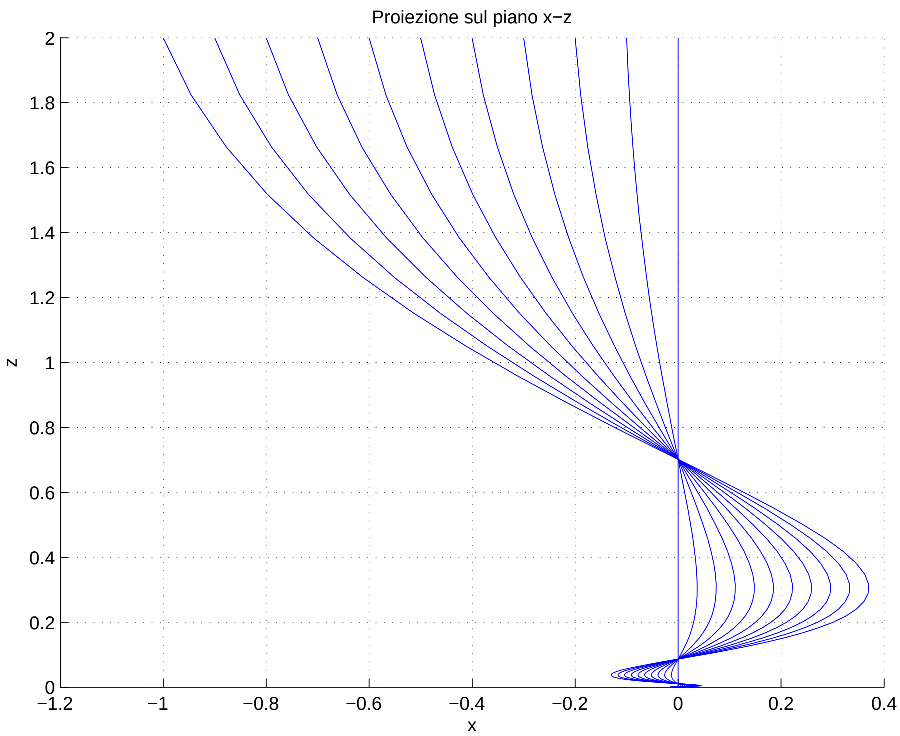
----- Matlab output --



```
pause;                      % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Projection on the x-y plane
figure(1)
view(0,90)
title('Projection on the x-y plane')
----- Matlab output ---------------------------------------------------------------
```

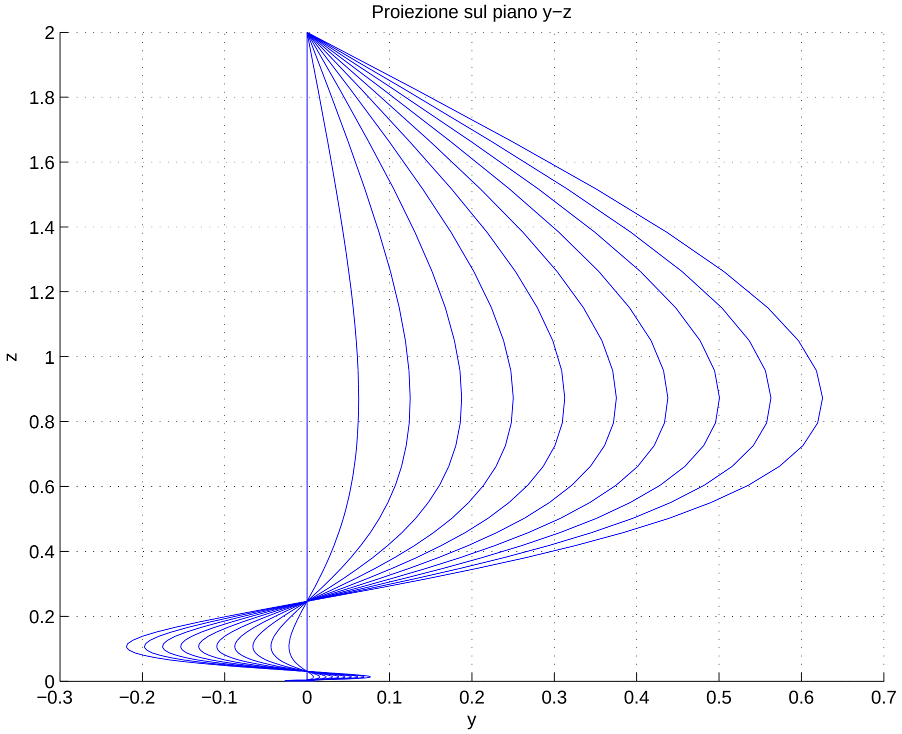



```
-------------------------------------------------------------------
pause;                      % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Projection on the x-z plane
figure(1)
view(0,0)
title('Projection on the x-z plane')
----- Matlab output ---------------------------------------------------------------
```




```
-------------------------------------------------------------------
```

```
pause;                   % Press any key to continue
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Projection on the y-z plane
figure(1)
view(90,0)
title('Projection on the y-z plane')
----- Matlab output ------------------------------------------------------
```




```
-------------------------------------------------------------------------
echo off
```