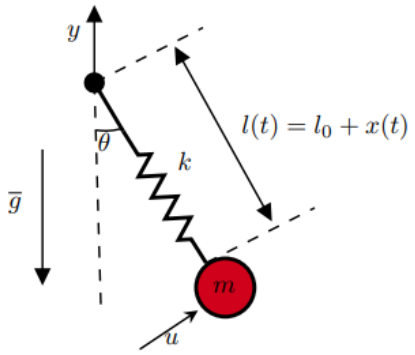
Student: Subtirica Gabriel

Grupa: 331AB

Modelul 2 – Pendulul elastic



Modelul matematic este descris de ecuaţiile:

$$\ddot{x}(t) = (\ell_0 + x(t))\dot{\theta}^2(t) - \frac{k}{m}x(t) + g\cos\theta(t) - \zeta\dot{x}(t) \quad (2a)$$

$$\ddot{\theta}(t) = -\frac{g}{\ell_0 + x(t)}\sin\theta(t) - \frac{2\dot{x}(t)}{\ell_0 + x(t)}\dot{\theta}(t) - \zeta\dot{\theta}(t) + \quad (2b)$$

$$+\frac{1}{m(\ell_0 + x(t))^2}u(t)$$

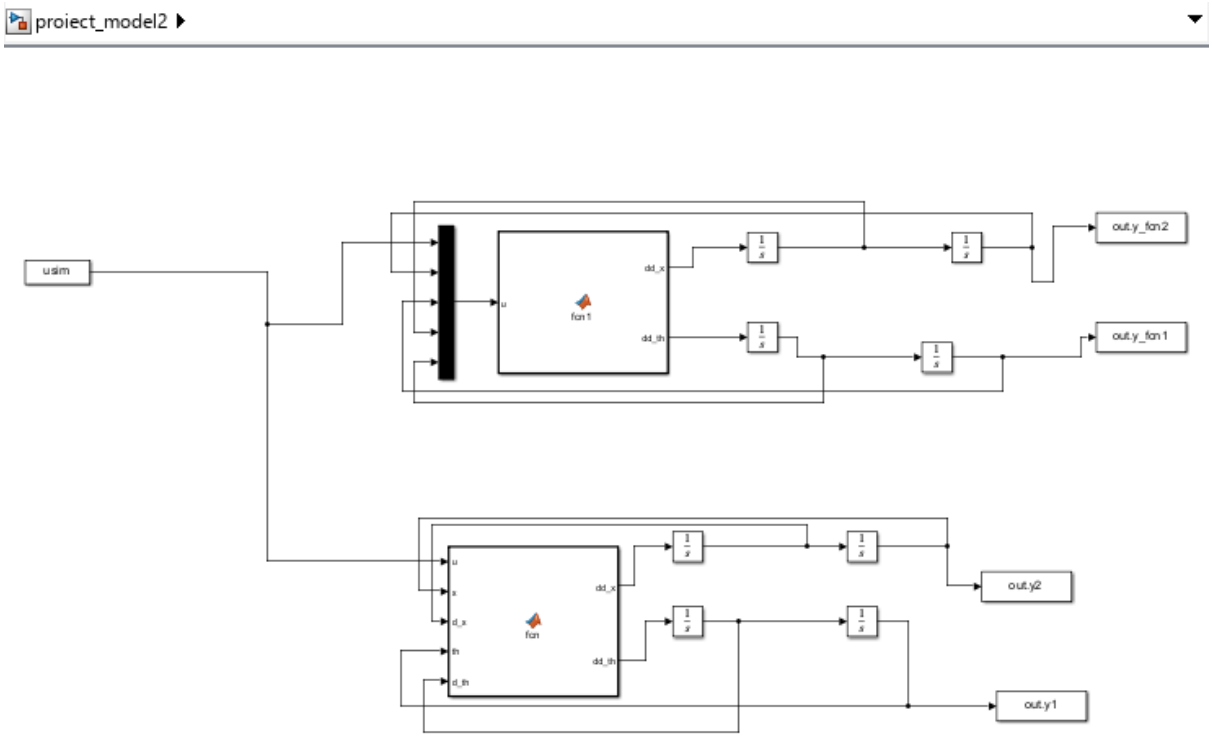
$$y_1(t) = \theta(t) \quad (2c)$$

$$y_2(t) = x(t) \quad (2d)$$

Cerinta 1: Incarcarea modelului prin Simulink

```
%% Cerinta 1

%Crearea si incarcarea modelului
model = 'proiect_model2';
load_system(model)
```

Modelul:



Modelul este alcatuit din doua Matlab Function:

Cod fcn:

proiect_model2 ▸ MATLAB Function

```
function [dd_x, dd_th] = fcn(u, x, d_x, th, d_th)

    %Alegerea valorilor pentru parametrii modelului in mod arbitrar(si pozitiv)
    l0 = 2.42;      %lungimea initiala
    m = 3.53;       %masa corpului
    k = 3.21;       %constanta elastica
    fact = 0.61;    %factorul de amortizare
    g = 9.81;       %acceleratia gravitationala

    %Formulele modelului
    dd_x = (l0 + x) * (d_th) * (d_th) - ((k * x) / m) + g * cos(th) - fact * d_x;
    dd_th = - (g * sin(th)) / (l0 + x) - (2 * d_x / (l0 + x)) * d_th - fact * d_th + 1 / (m * (l0 + x) * (l0 + x)) * u;

end
```

Cod fcn1:

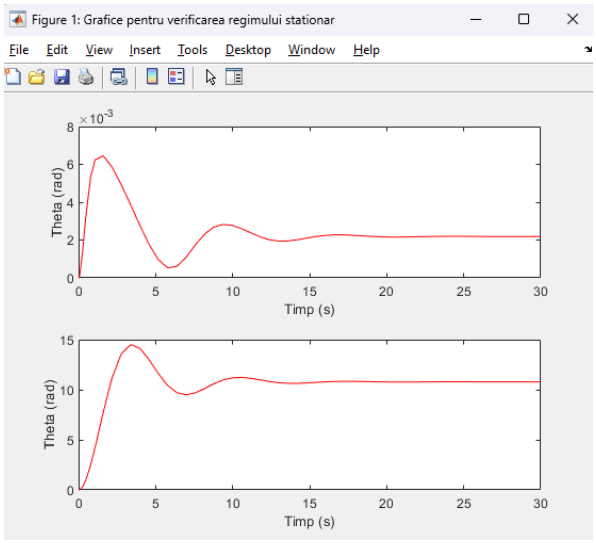
proiect_model2 ▸ MATLAB Function1

```
function [dd_x, dd_th] = fcn1(u)

    %Alegerea valorilor pentru parametrii modelului in mod arbitrar(si pozitiv)
    l0 = 2.42;      %lungimea initiala
    m = 3.53;       %masa corpului
    k = 3.21;       %constanta elastica
    fact = 0.61;    %factorul de amortizare
    g = 9.81;       %acceleratia gravitationala

    %Formulele modelului
    dd_x = (l0 + u(2)) * (u(5)) * (u(5)) - ((k *u(2)) / m) + g * cos(u(3)) - fact * u(4);
    dd_th = - (g * sin(u(3)) / (l0 + u(2)))- ((2 * u(4) * u(5)) / (l0 + u(2))) - fact * u(5) + (u(1) / (m * (l0 + u(2)) * (l0 + u(2))));

end
```

Cerinta 2:

```
%% Cerinta 2

% Setam timpul de simulare la 30 de secunde
timp = 30;
% Crearea ununui vector de timp cu 100 de puncte, de la 0 la timp
t = linspace(0, timp, 100);
% Crearea ununui vector de stare inițială
st = double(t >= 0);
% Crearea unei serii temporale din vectorul de stare inițială și vectorul de timp
usim = timeseries(st, t);
% Setarea parametrului `Stoptime` al modelului la timp
set_param(model, 'Stoptime', num2str(timp))
% Simulez modelul
out = sim(model);

%Salvez valorile de iesire ale modelului in variabilele: simout_mat1, simout_mat2, simout_fcn1, simout_fcn2;
simout_mat1 = out.y1;
simout_mat2 = out.y2;
simout_fcn1 = out.y_fcn1;
simout_fcn2 = out.y_fcn2;

%Generarea graficelor
figure('Name', 'Grafice pentru verificarea regimului stationar');
subplot(2,1,1);
plot(simout_mat1.time, simout_mat1.data, 'r');
xlabel('Timp (s)');
ylabel('Theta (rad)');
hold on
subplot(2,1,2);
plot(simout_mat2.time,simout_mat2.data, 'r');
xlabel('Timp (s)');
ylabel('Theta (rad)');

%In urma plotarii graficelor se poate observa faptul ca cele 2 simulari au
%atins un regim stationar, deoarece valorile raspunsului se stabilizeaza(si
%nu se mai modifica)
```

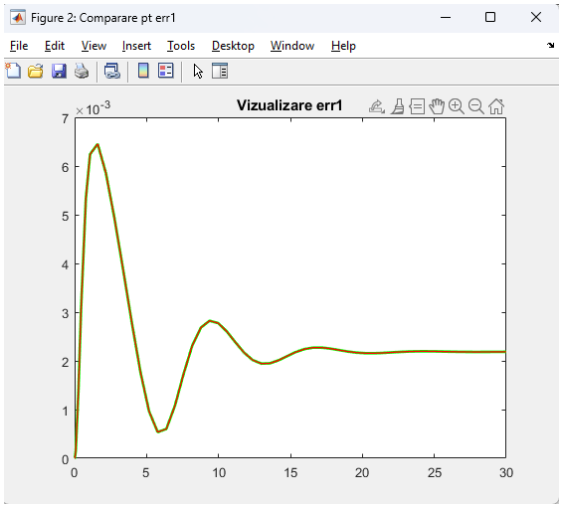
Se observa in Figura 1 obtinerea regimului stationar:

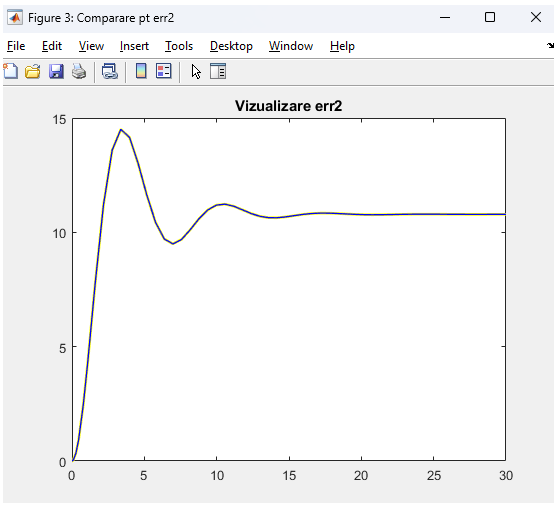


Cerinta 3:

```
%% Cerinta 3

%Simularea modelului
out = sim("proiect_model2");

%err1
figure('Name', 'Comparare pt err1');
plot(simout_mat1.time, simout_mat1.data, 'g', 'LineWidth', 2);
hold on;
plot(simout_fcn1.time,simout_fcn1.data, 'r', 'LineWidth', 1);
title('Vizualizare err1');

%err2
figure ('Name', 'Comparare pt err2');
plot(simout_mat2.time, simout_mat2.data, 'y', 'LineWidth', 2);
hold on;
plot(simout_fcn2.time,simout_fcn2.data, 'b', 'LineWidth', 1);
title('Vizualizare err2');

simout_mat1 = simout_mat1.data;
simout_mat2 = simout_mat2.data;

simout_fcn1 = simout_fcn1.data;
simout_fcn2 = simout_fcn2.data;
```

In cele doua figuri se observa similaritatea graficelor in ambele cazuri:





Cerinta 4:

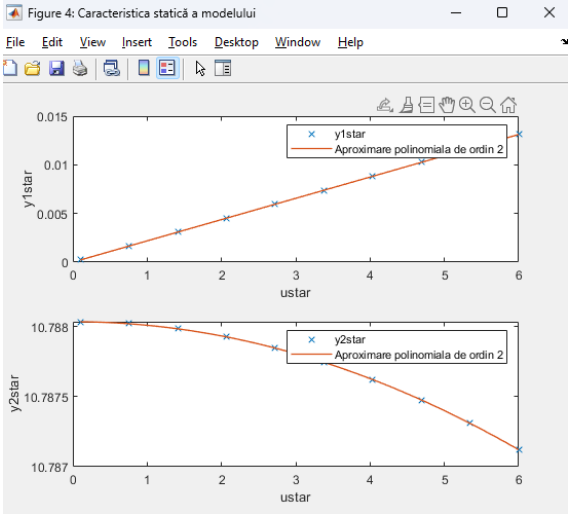
```
%% Cerinta 4

err1 = norm(simout_mat1 - simout_fcn1);
err2 = norm(simout_mat2 - simout_fcn2);
%Afisez erorile
disp('err1');
disp(err1);
disp('err2');
disp(err2);
```

```
err1
   3.3540e-18

err2
     0
```

Se poate observa ca nu exista discrepante(cel putin nu extraordinar de vizibile), deoarece erorile sunt foarte mici.

La err2 probabil ca nu trebuia sa fie exact 0, dar oricum trebuie sa fie o valoare apropiata de 0.

Cerinta 5:



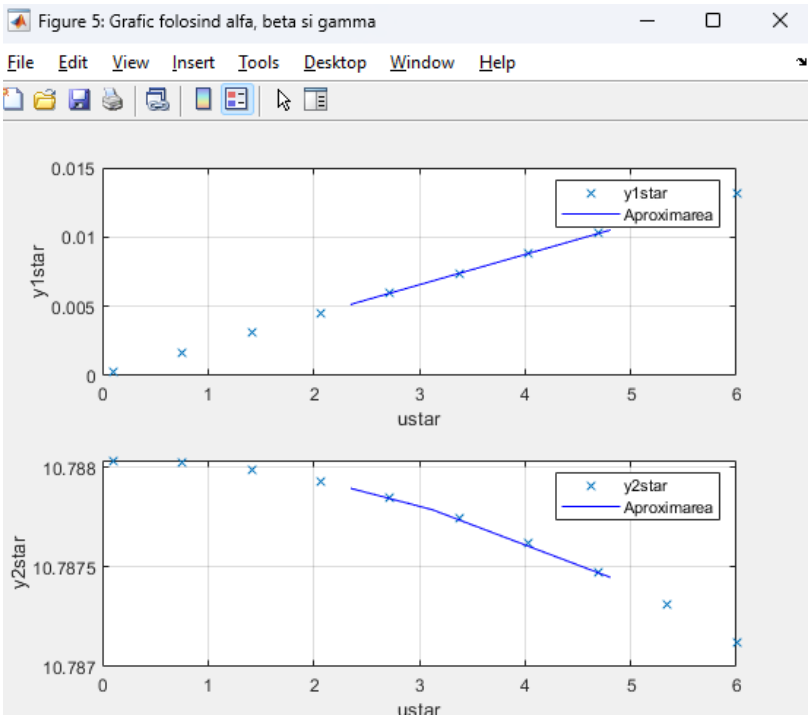
```
%% Cerinta 5

% Creez un vector de 10 valori ale semnalului de intrare `x` cuprinse intre 0.1 si 6.
x = linspace(0.1, 6, 10);

%Creez vectorii ustar, y1star, si y2star pentru a stoca valorile raspunsurilor sistemului.
ustar = zeros(10, 1);
y1star = zeros(10, 1);
y2star = zeros(10, 1);

simout_mat1 = out.y1;
simout_mat2 = out.y2;

for i = 1 : 10
    in = x(i).*st;
    usim = timeseries(in, t);
    out = sim(model);
    ustar(i) = x(i);

    % Stochez valoarea răspunsurilor y1 si y2 în vectorul y1star.
    y1star(i) = out.y1.data(end);
    y2star(i) = out.y2.data(end);
end
```

```
% Aproximarea polinomiala de ordin 2
%folosire polyfit pt a determina coeficientii
p1 = polyfit(ustar, y1star, 2);
p2 = polyfit(ustar, y2star, 2);
%genmerarea unui vector de valori pt exproximarea polinomiala
ustar1 = linspace(ustar(1), ustar(end), 100);
y1starr = polyval(p1, ustar1);
y2starr = polyval(p2, ustar1);

figure('Name', 'Caracteristica statică a modelului');
subplot(2,1,1);
%pune un x in fiecare punct
plot(ustar, y1star, 'x');
hold on
plot(ustar1, y1starr, 'LineWidth', 1);
xlabel('ustar'), ylabel('y1star');
legend('y1star', 'Aproximare polinomiala de ordin 2');

subplot(2,1,2);
plot(ustar, y2star, 'x');
hold on
plot(ustar1, y2starr, 'LineWidth', 1);
xlabel('ustar'), ylabel('y2star');
legend('y2star', 'Aproximare polinomiala de ordin 2');
```

Cerinta 6:

```
%% Cerinta 6

%Definirea valorilor pentru alpha, beta, gamma
alpha = 4.81;
beta = 3.12;
gamma = 2.35;

%manipularea valorilor
ustar_nou = [alpha, beta, gamma];
y1star_nou = polyval(p1, ustar_nou);
y2star_nou = polyval(p2, ustar_nou);

figure('Name', 'Grafic folosind alfa, beta si gamma');
subplot(2,1,1);
plot(ustar, y1star, 'x');
hold on
plot(ustar_nou, y1star_nou, 'b');
grid on
xlabel('ustar'), ylabel("y1star")
legend('y1star','Aproximarea')

subplot(2,1,2);
plot(ustar, y2star, 'x');
hold on
plot(ustar_nou, y2star_nou, 'b');
grid on
xlabel('ustar'), ylabel("y2star")
legend('y2star','Aproximarea')
```

Graficul obtinut:



ustar_nou | y1star_nou | y2star_nou

1x3 double

	1	2	3	4
1	4.8100	3.1200	2.3500	

ustar_nou | y1star_nou | y2star_nou

1x3 double

	1	2	3	4
1	0.0105	0.0068	0.0051	

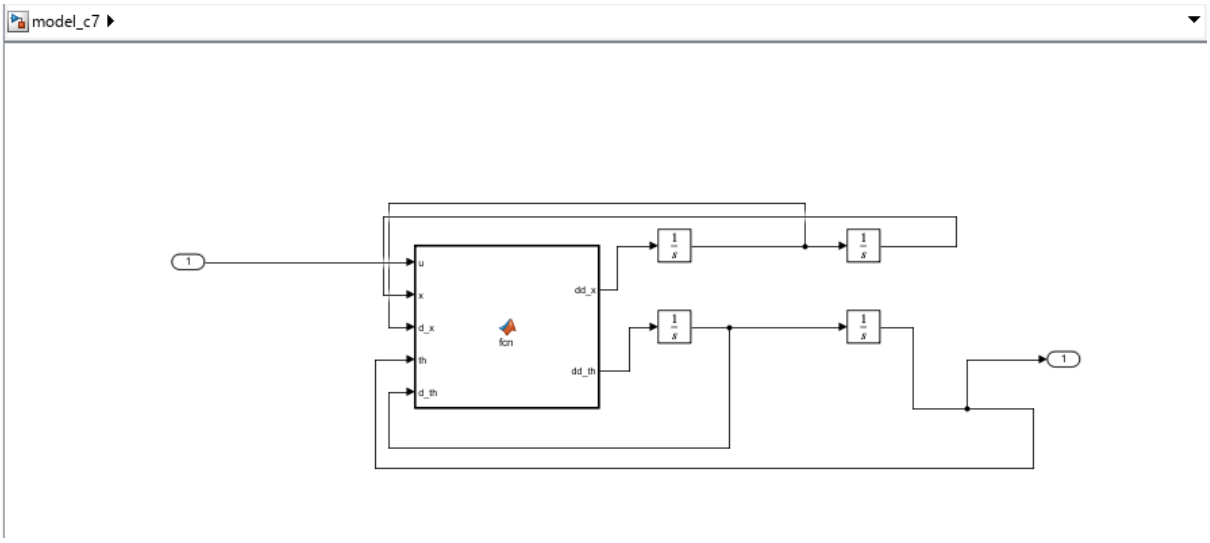
ustar_nou | y1star_nou | y2star_nou

1x3 double

	1	2	3	4
1	10.7874	10.7878	10.7879	

Cerinta 7:

```
%% Cerinta 7

%Am creat un nou model simulink(model_c7.slx) pe baza caruia am utilizat
%modelul de la prima cerinta, lunad in calcul doar iesirea y1(am inlocuit
%intrarea usim si iesirea y1 cu input, respectiv output)
```



Cerinta 8:

```
%% Cerinta 8

u0 = zeros(1, 1);
[xstar, ustar, ystar, ~] = trim("model_c7", [], u0, [], [], 1, []);
%Afisarea valorii punctului static de functionare
fprintf("Eroare = %.2f\n", norm(abs(ustar - u0)));

%Se observa faptul ca nu sunt diferente intre cele 2 valori, asadar
%valoarea erorii este 0
```

```
Eroare = 0.00
```

Cerinta 9:

```
%% Cerinta 9

%utilizarea functiei linmod() pentru a determina o aproximare liniara a
%raspunsului modelului la un semnal de intrare constant
[A_lin, B_lin, C_lin, D_lin] = linmod("model_c7", xstar, ustar)
```

```
A_lin =

         0         0    1.0000         0
         0   -0.7100         0   -0.7086
   -0.6032   -0.0000   -0.7100   -0.0000
         0    1.0000         0         0


B_lin =

   1.0e-03 *

         0
         0
    0.8345
         0


C_lin =

     1     0     0     0


D_lin =

     0
```

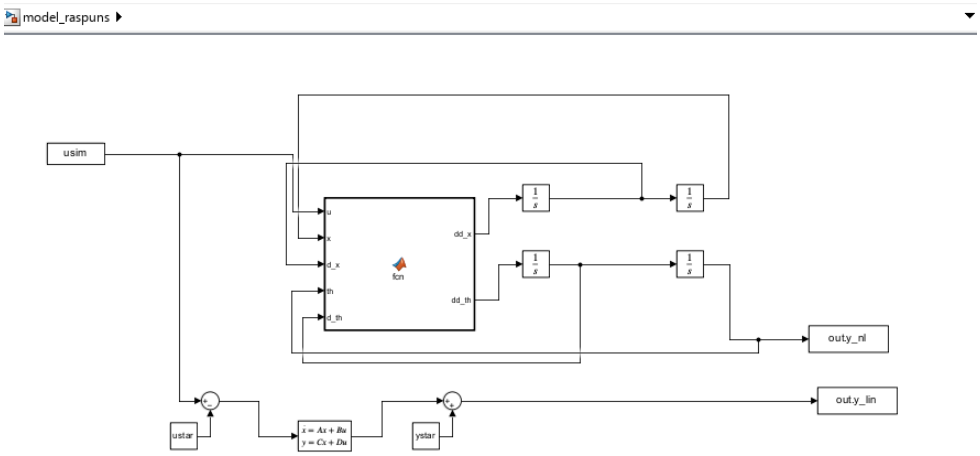
Cerinta 10:

```
%% Cerinta 10

vp = eig(A_lin)
%sistemul este stabil, deoarece toate valorile spectrului matricei A_lin
%sunt mai mici decat 0;
```

```
vp =

  -0.3550 + 0.6908i
  -0.3550 - 0.6908i
  -0.3550 + 0.7633i
  -0.3550 - 0.7633i
```

Cerinta 11:

```
%% Cerinta 11

model_nou = 'model_raspuns';
load_system = ('model_raspuns');
%crearea semnalului de intrare
usim = timeseries(st, t);
set_param(model_nou, 'StopTime', num2str(timp))
%simularea raspunsului modelului
out = sim(model_nou);
%obtinerea raspunsului liniar
y_lin = out.y_lin;
```



Cerinta 12:

```
%% Cerinta 12

%obtinerea raspunsului neliniar
y_nl = out.y_nl;
%redimensionarea raspunsurilor
y_lin_flat = reshape(y_lin.data, [1, 60]);
y_nl_flat = reshape(y_nl.data, [1,60]);
%calcularea normei/a erorii
err = norm(y_nl_flat - y_lin_flat, 'inf');
disp('Eroarea dintre rasp neliniar si cel liniar');
disp(err);
```

```
Eroarea dintre rasp neliniar si cel liniar:
    0.0044
```

Cerinta 13:

```
%% Cerinta 13


% Alegerea perioadei de esantionare
Te = 0.05;

% Discretizarea modelului liniarizat continuu
%sys_disc = c2d(sys_cont, Te, 'zoh');

% Afisarea modelului discretizat
disp('Modelul discretizat:');
disp(sys_disc);
```

Am incercat sa fac ceva cu modelul „ZOH”, dar nu am reusit