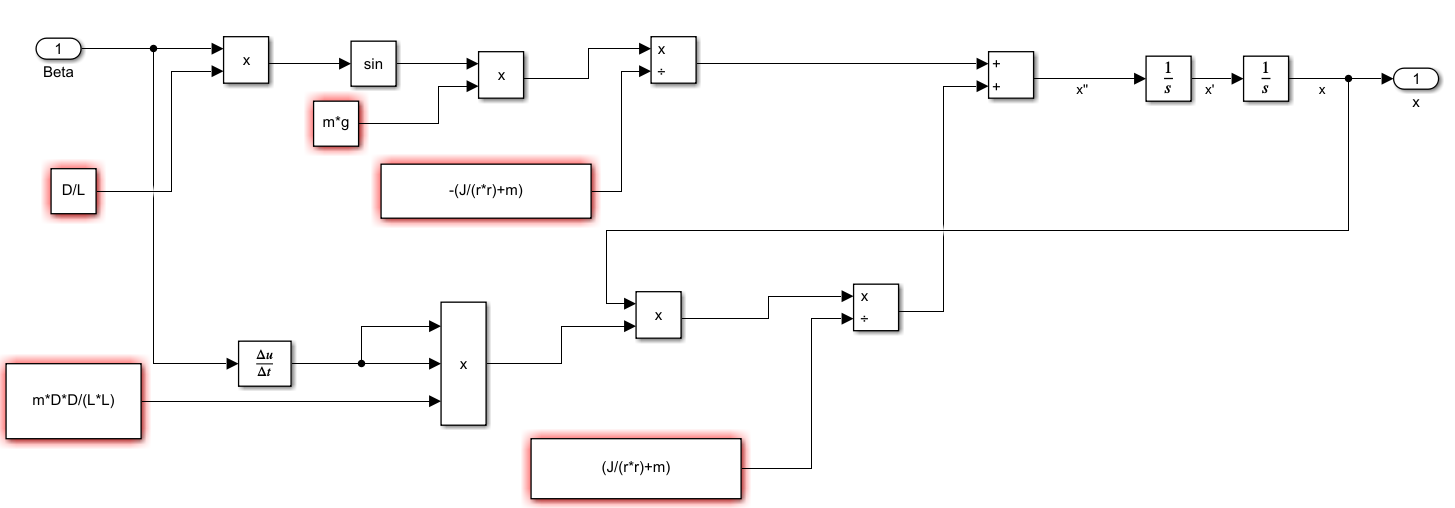
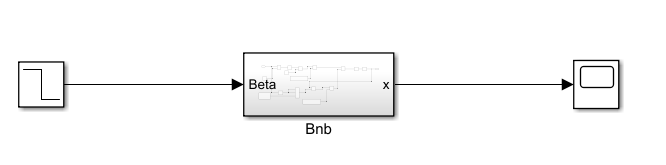
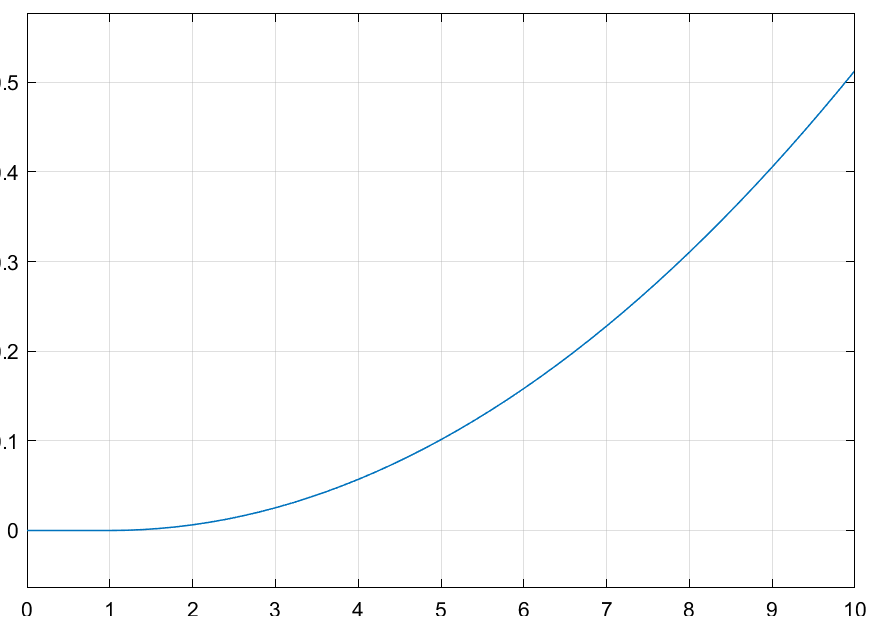
Lab1

BnB

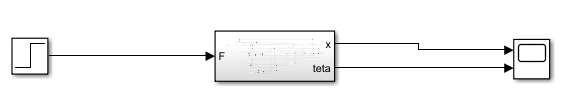


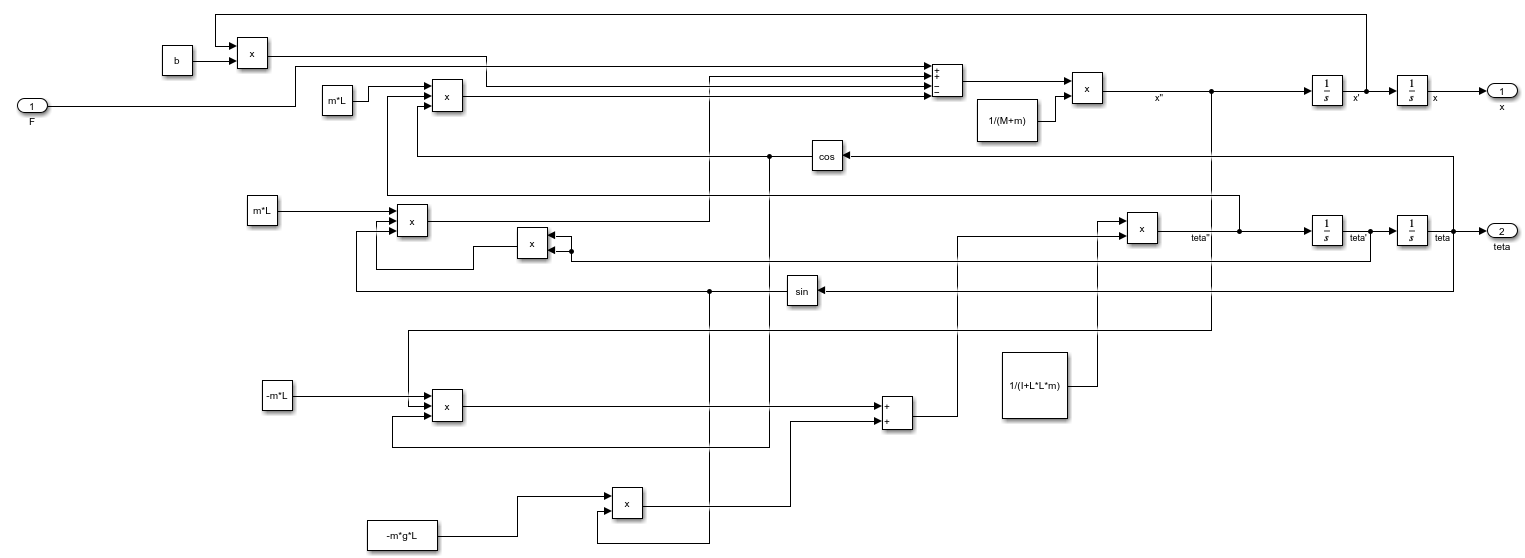
Rezultate obtinute:



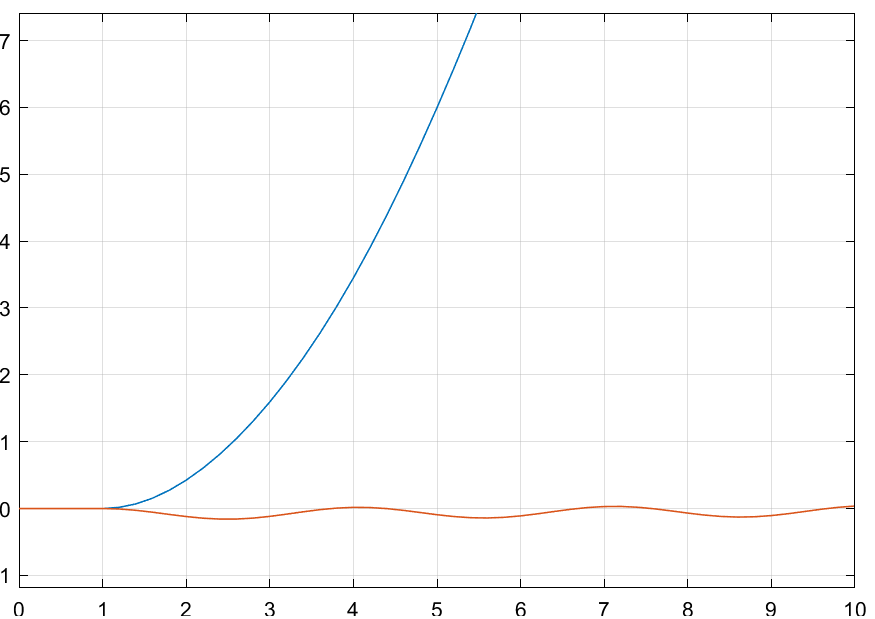
Sistemul este neliniar si instabil.

Pentul Invers:





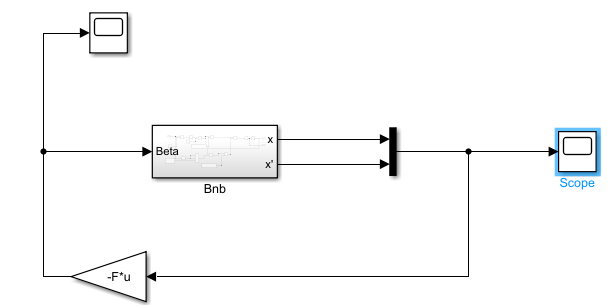
Rezultate obtinute:



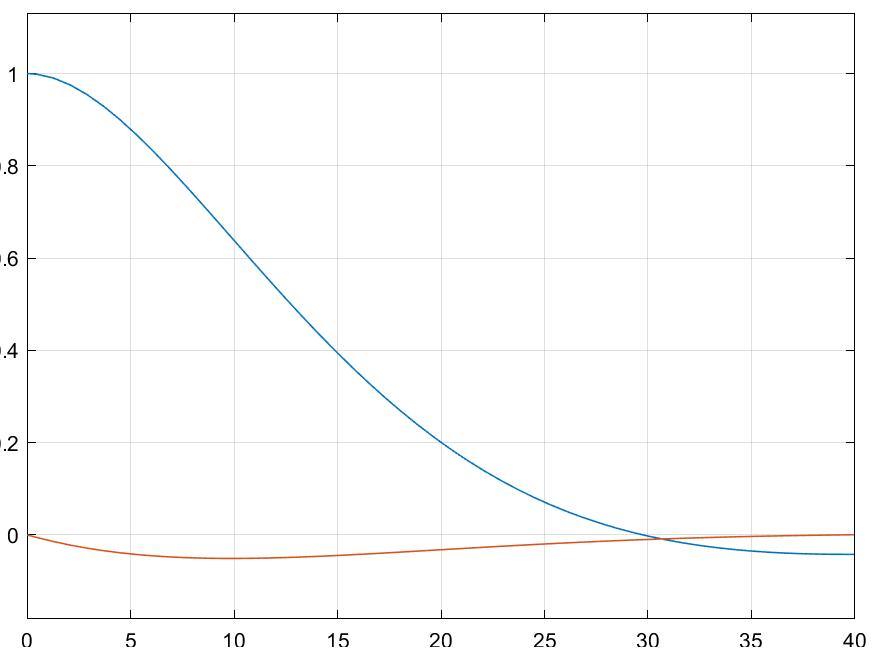
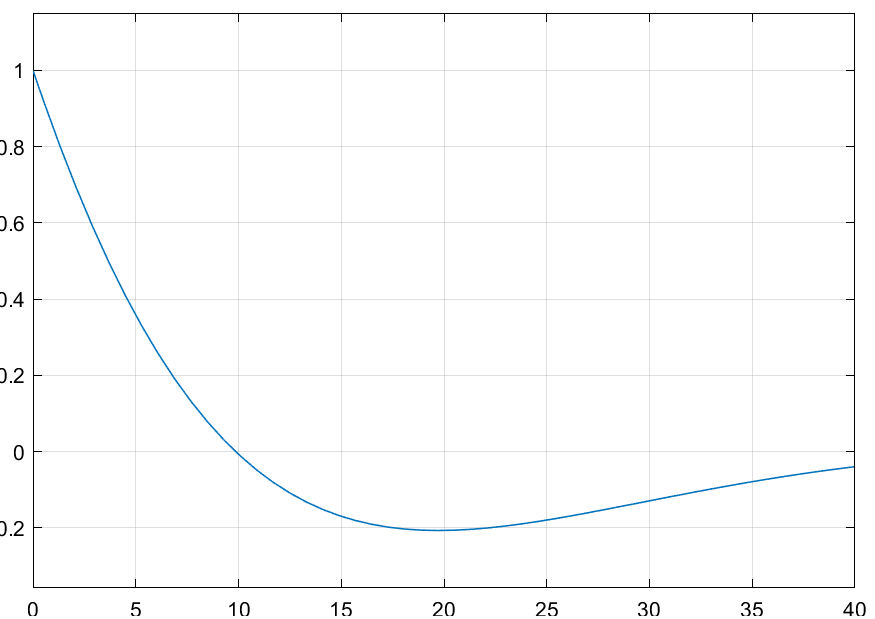
Sistemul este instabil si neliniar.

Lab2

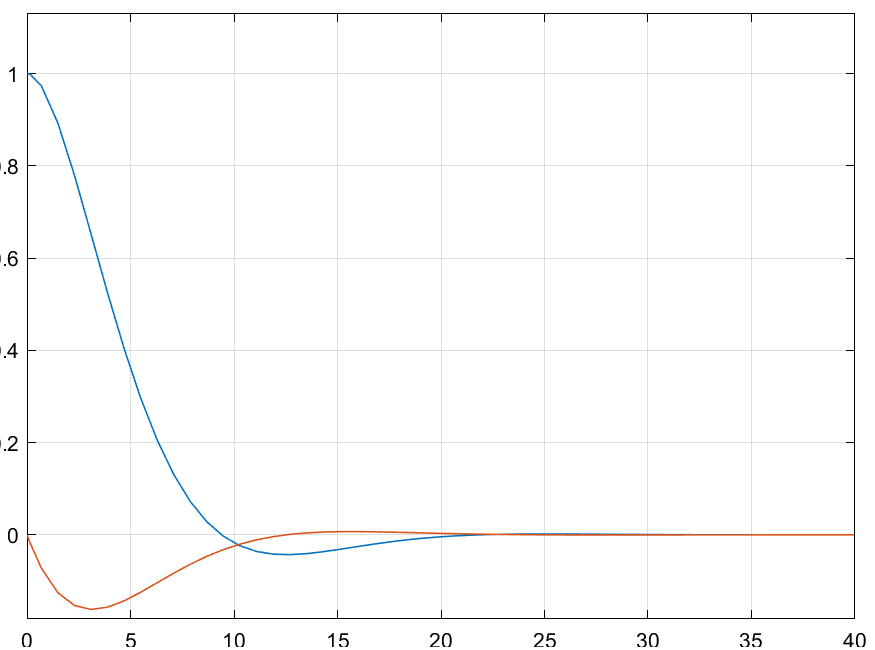
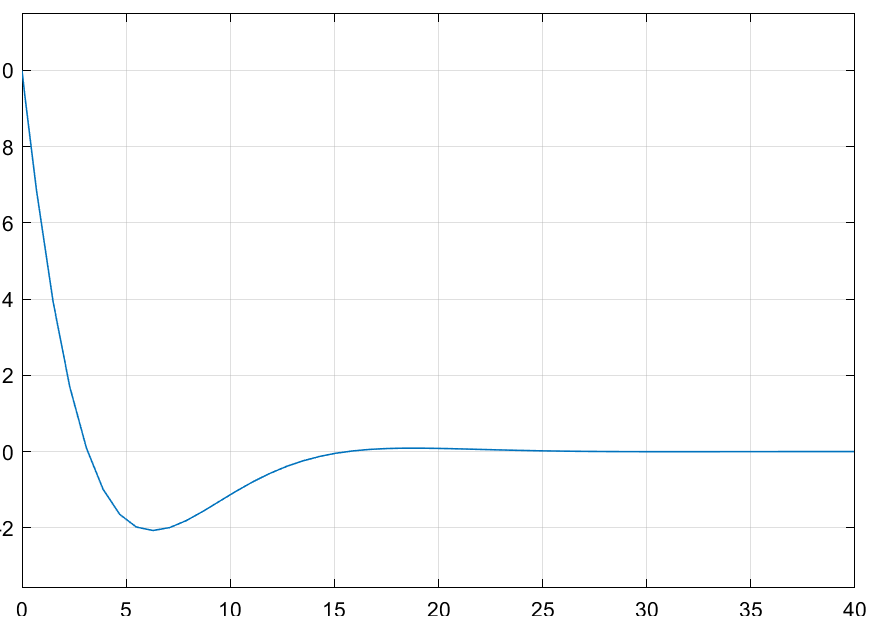
BnBOptimal



Rezultate Obtinute:

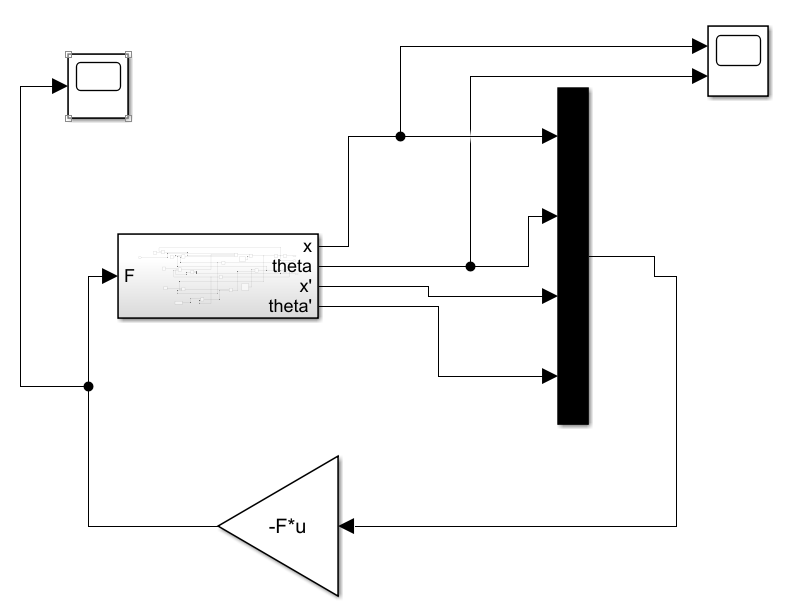
Q = [1 0; 0 1] Comanda:

Q = [100 0; 0 1] Comanda:

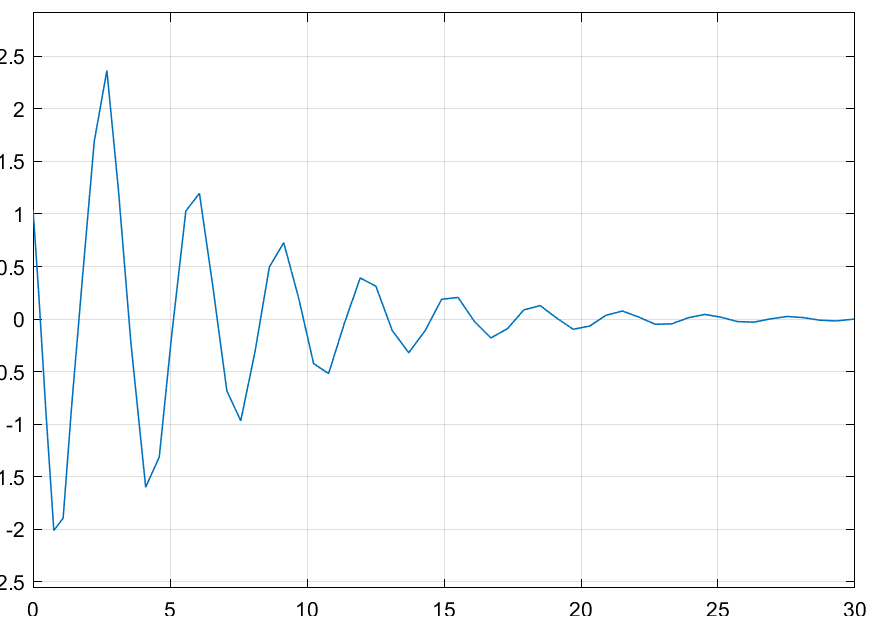
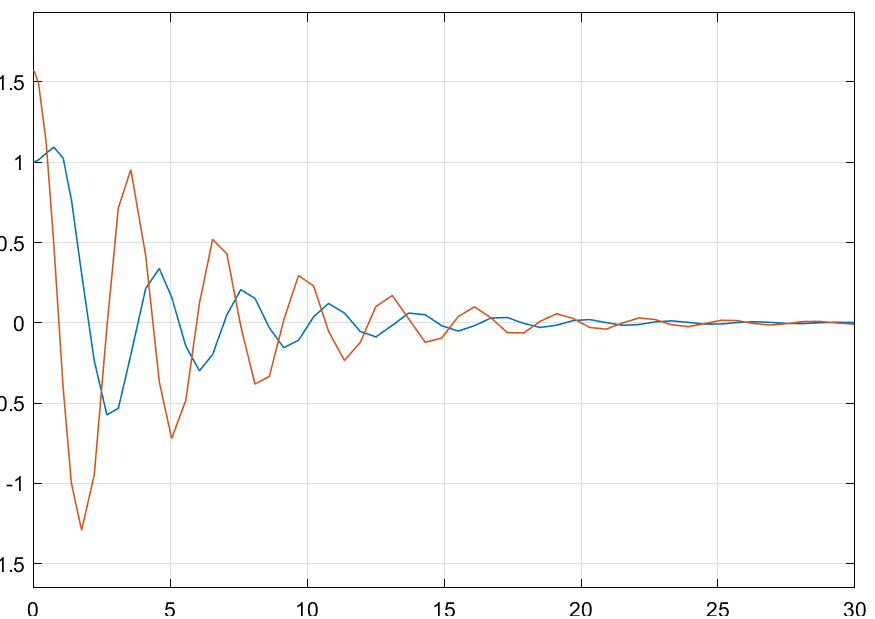


Performanta de 3 ori mai puna pt o comanda de 10 ori mai mare.

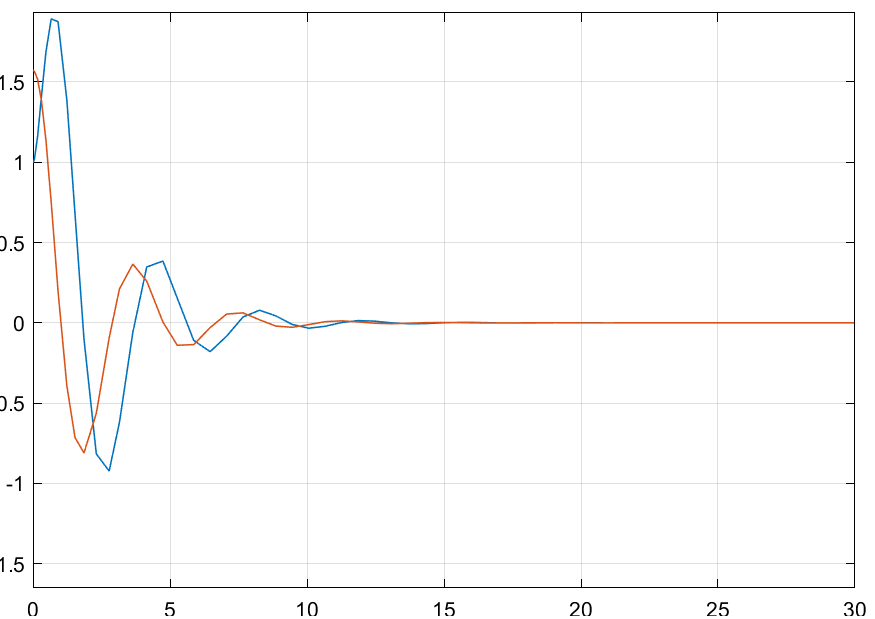
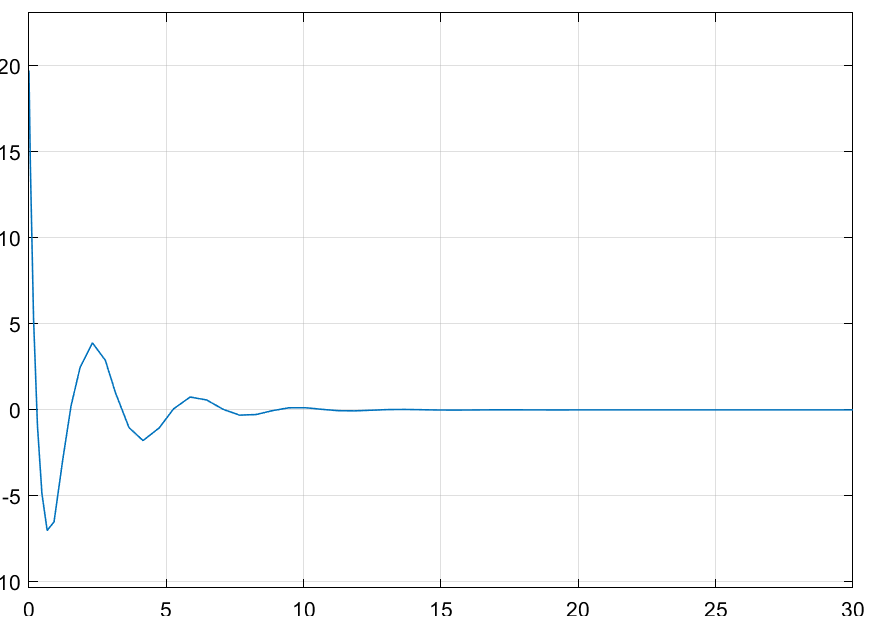
Pendul Invers Optimal:



Rezultate:

Q=eye(4) Comanda:

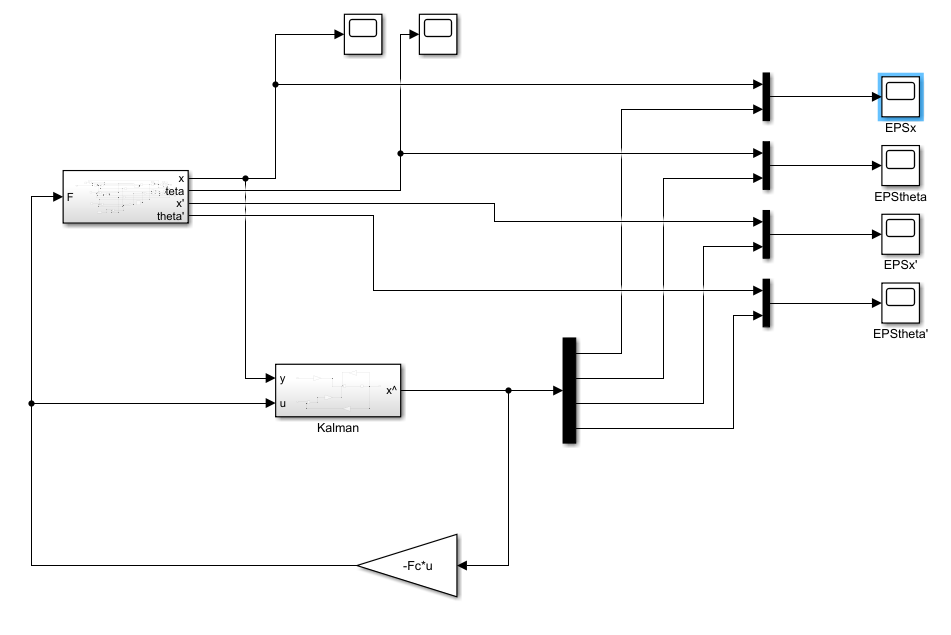
Q=[100 0 0 0; 0 1 0 0; 0 0 1 0; 0 0 0 100] Comanda:

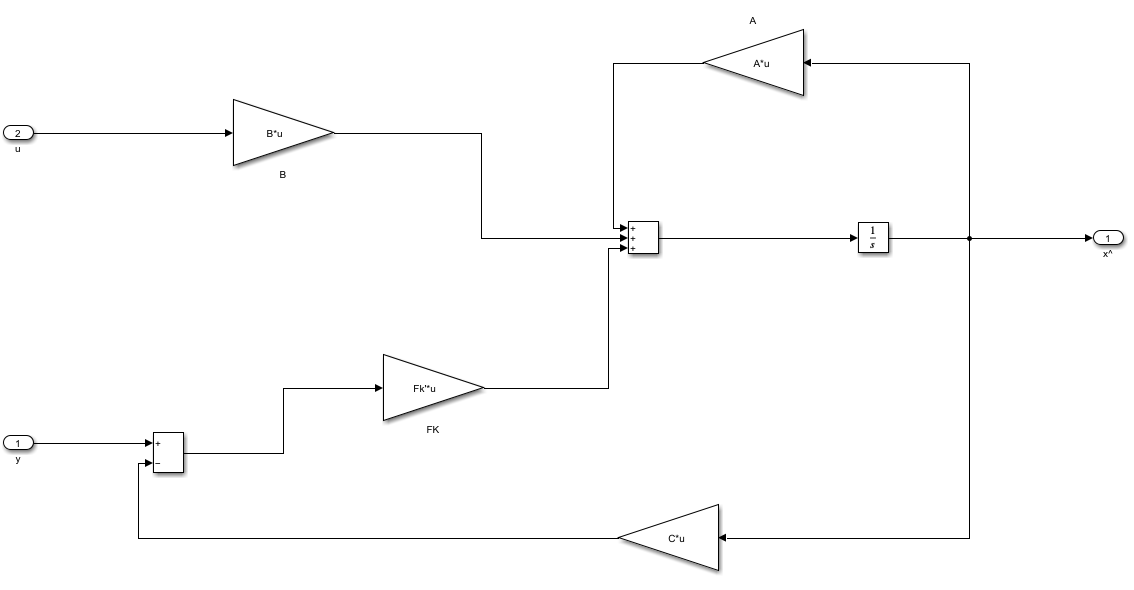


La fel ca in cazul BnB odata cu cresterea costului cresc si performantele.

Lab3

Schemele simulink conform laboratorului:

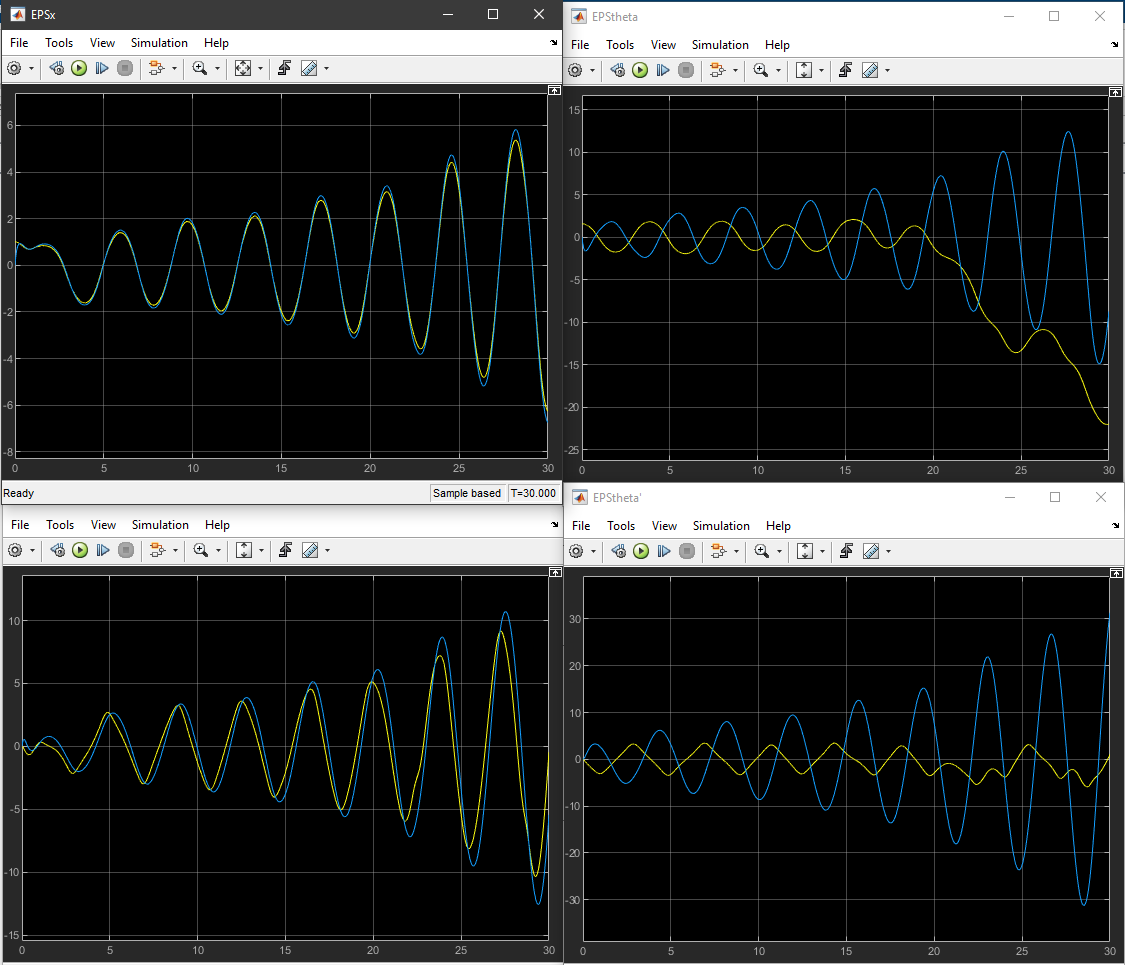




Pentru Qc = [1 0 0 0; 0 1 0 0; 0 0 1 0; 0 0 0 1]

Qk = [100 0 0 0; 0 100 0 0; 0 0 100 0; 0 0 0 100]

Am obtinut urmatoarele rezultate:



Lab4

Polinoamele se aduc in forma canonica (gradA > gradB).

B si C sunt stabile.

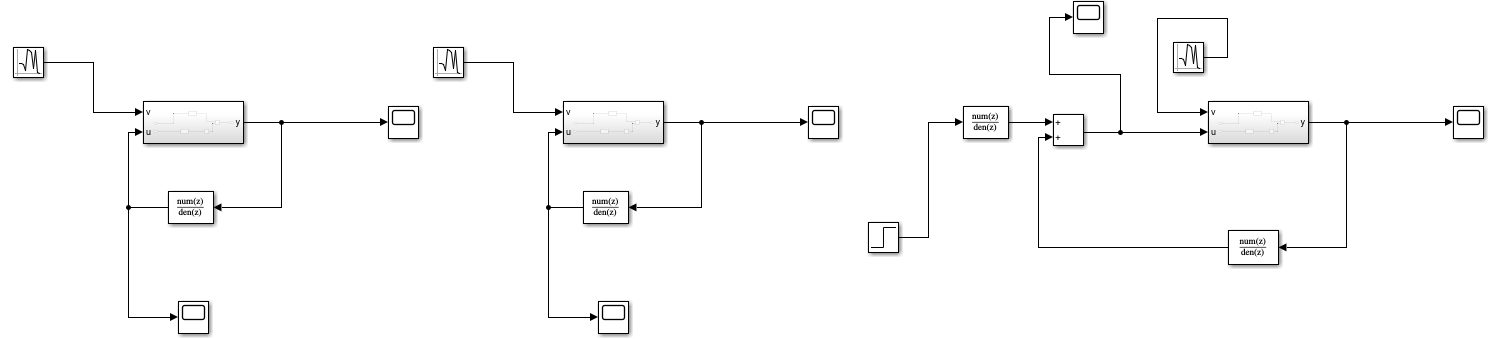
Grad F = d;

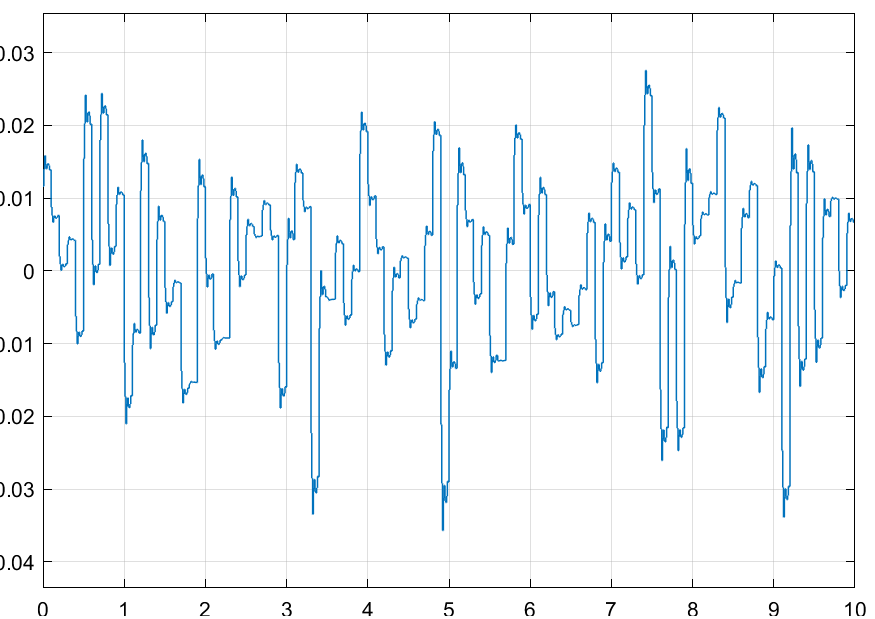
Grad G = n-1;

F = catul impartirii lui C la A;

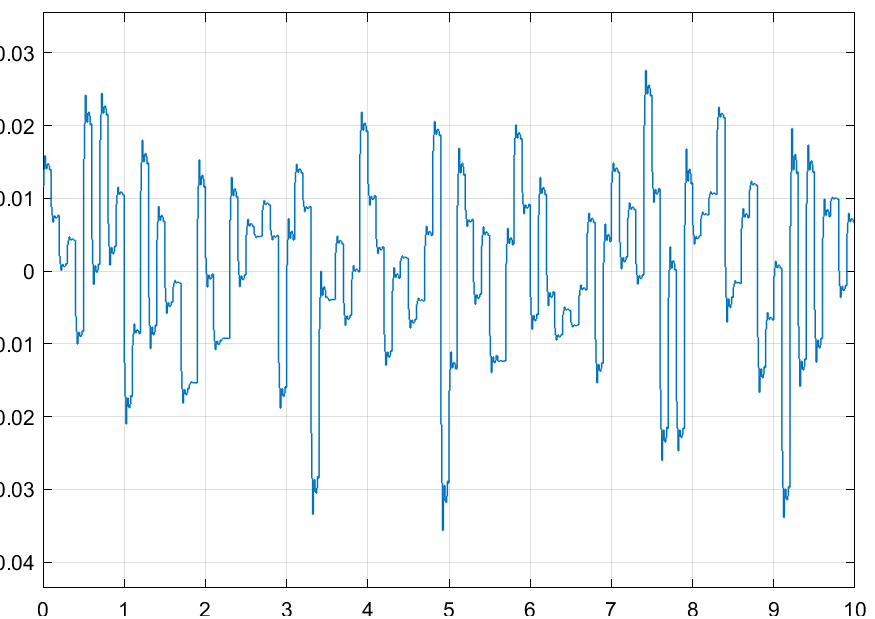
G = restul impartirii lui C la A;

Criteriul J1 Criteriul J2 Criteriul J3

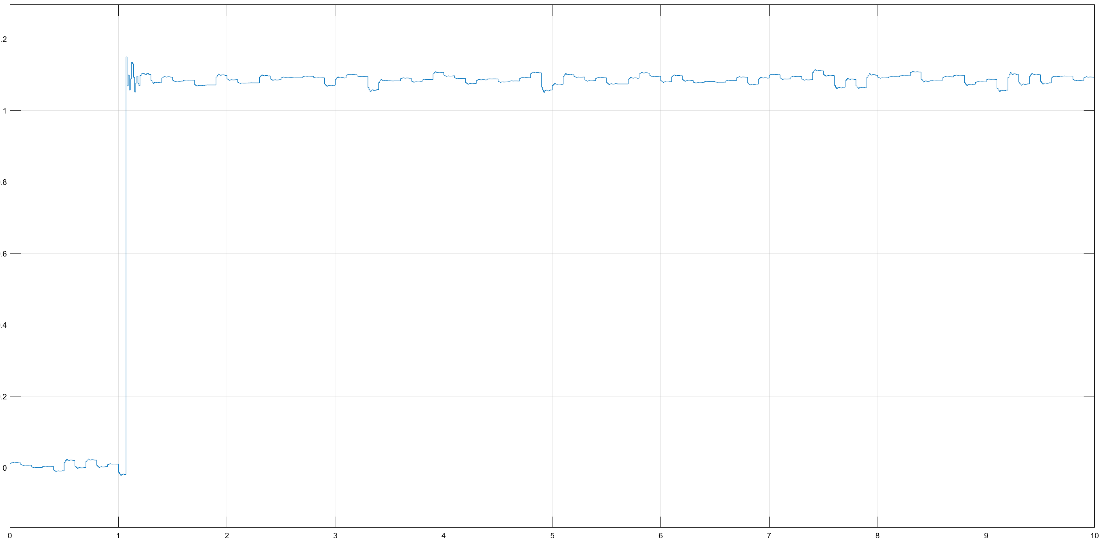




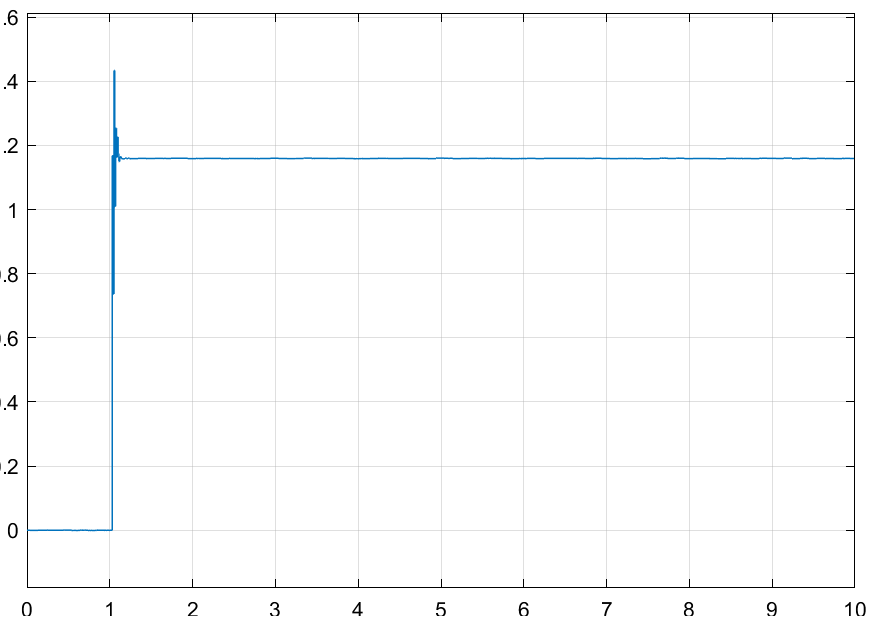
J1: minimizarea dispersiei iesirii in raport cu perturbatia



J2: penalizarea comenzii



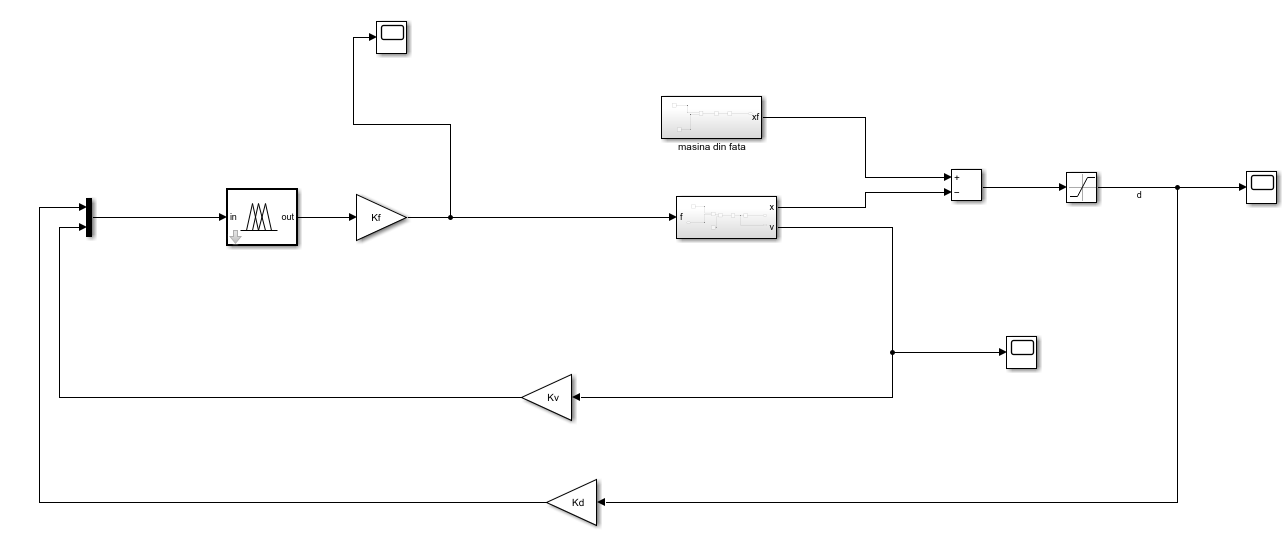
J3: asigura performantele independente in urmarire si reglare

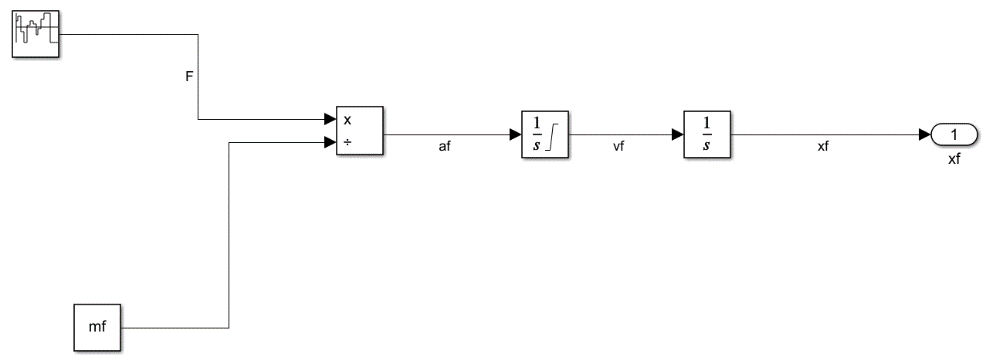


Comanda J3:

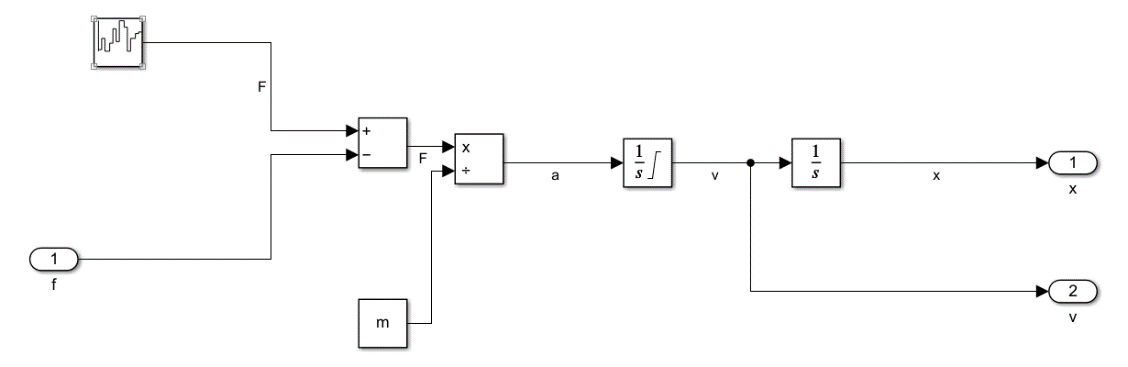
Lab6

Slx Fuzzy:





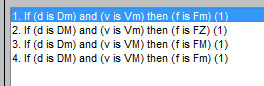
Masina din fata:



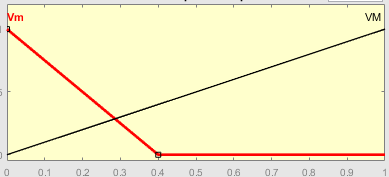
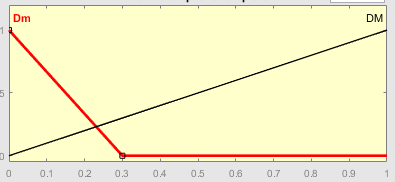
Masina din spate:

Regulatorul Fuzzy:

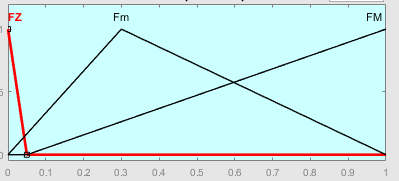
Reguli aplicate:



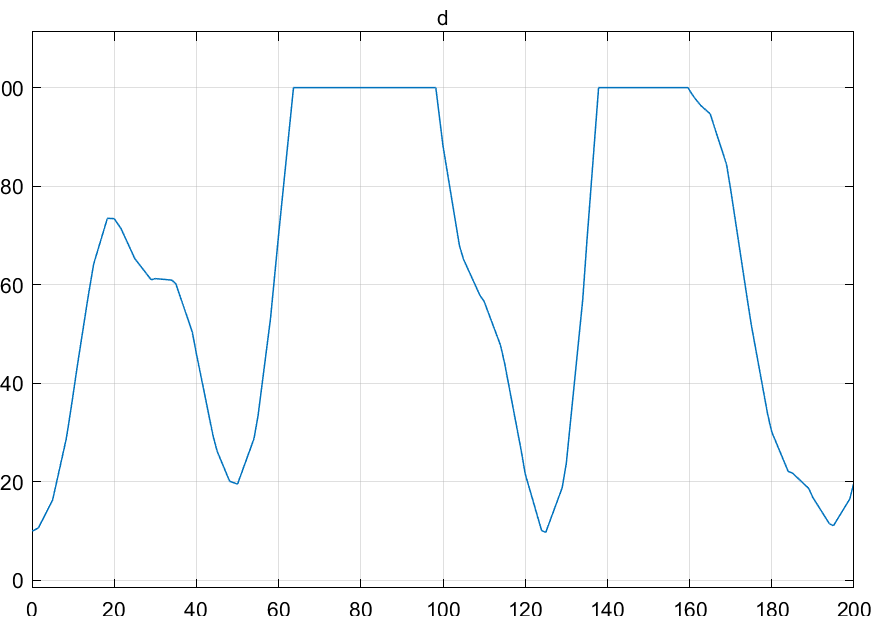
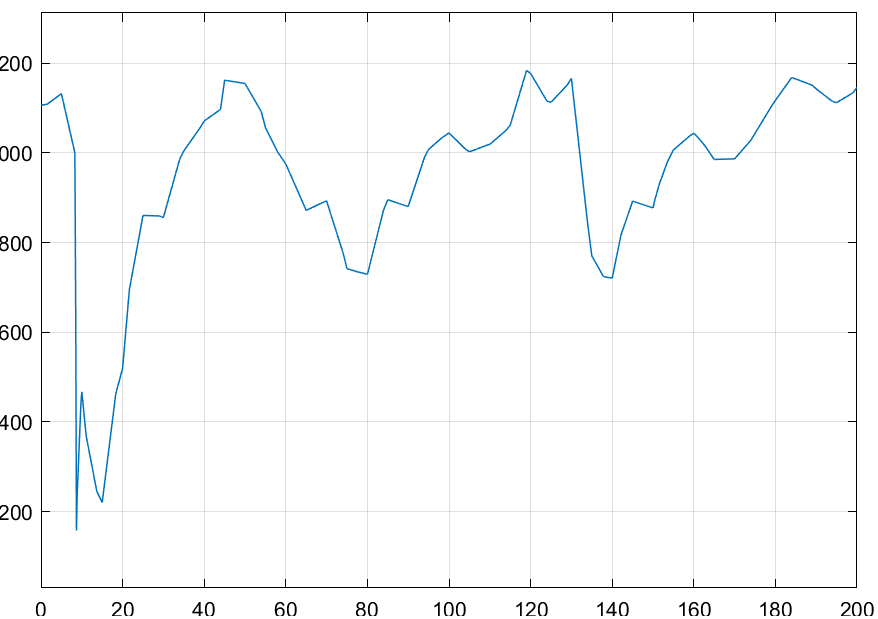
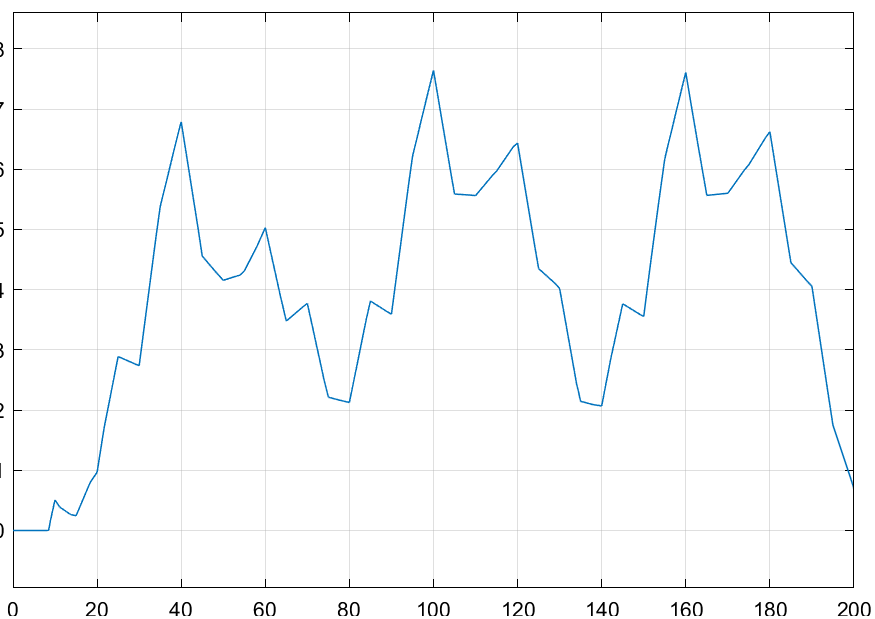
Intrarile:



Iesirea:



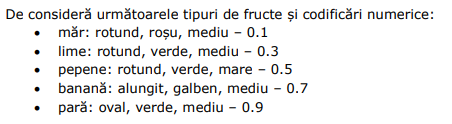
Rezultate obtinute:

Distanta intre masini: Forta de franare: Viteza masinii din spate:

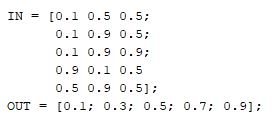
Lab7

Antrenam un neuron pentru a clasifica fructe in functie de forma, culoare si marime.

Avem urmatoarele codificari:



Setul de invatare:



Rezultatele obtinute sunt destul de diferite de la o rulare la alta.

Exemple rulari pt precizia de oprire 0.01:



Exemple rulari pt precizia de oprire 0.1:

Exemplu personal:

Antrenam un neuron pentru a clasifica masini in functie de cost, culoare si marca.

Avem urmatoarele codificari:

Cost:

0.1– pret ieftin

0.5 – pret mediu

0.9 – pret scump

Culoare:

0.1– rodu

0.3 – galben

0.6 – verde

0.9 – albastru

Cost:

0.1– Audi

0.5 – VW

0.9 – Dacia

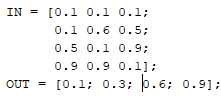
Se considera urmatoarele tipuri de masini si codificari numerice(ce reprezinta persoanele care au acele masini):

Alin: ieftin, rosu, audi – 0.1;

Dan: ieftin, verde, VW – 0.3;

Bogdan: mediu, rosu, Dacia – 0.6;

Rares: scump, albastru, Audi – 0.9;

Setul de invatare este:

Rezultate obtinute:

Lab8

Am rulat scriptul prezentat si am obtinut solutia in functie de algoritmul AG:



Pentru o analiza mai ampla a algoritmului AG am schimbat valroi.

Krmax,Timax,Tdmax = 100 si restul setarilor default:



Dimensiunea populatiei: 50 si restul setarilor default:



Numarul de generatii (iteratii): 100 si restul setarilor default:



Concluzie: Se observa ca in toate cazurile prezentate mai sus suprareglajul este unul de sub 5% si timpul tranzitoruiu este foarte mic.