ΕΡΓΑΣΙΑ ΕΞΑΜΗΝΟΥ ΕΙΔΙΚΑ ΘΕΜΑΤΑ ΣΑΕ

ON/MO: ΝΙΚΟΛΑΟΣ ΚΥΡΙΤΣΗΣ

AEM: 57322

Οι σταθερές μεταβλητές σύμφωνα με τα στοιχεία μου :

m = 5 : $b_5 = 13$, $b_4 = 9$, $b_3 = 10$, $b_2 = 15$, $b_1 = 18$.

n = 8 : $a_8 = 10$, $a_7 = 20$, $a_6 = 17$, $a_5 = 9$, $a_4 = 19$, $a_3 = 18$, $a_2 = 7$, $a_1 = 18$.

Οι συναρτήσεις f,g

$$f_1(x) = x_1 x_3 x_5 + \log(x_1)$$
$$g_1(x) = x_4 x_3 \cos(x_2) + x_1 x_2 \log(x_1)$$

$$f_2(x) = x_1 x_3 \cos(x_4) + x_2 \log(x_1)$$
$$g_2(x) = x_4 x_5 \log(x_1) + x_2 x_4 \cos(x_2)$$

$$f_3(x) = x_2 x_3 x_4 + x_1 \cos(x_2)$$
$$g_3(x) = x_1 x_5 \mathrm{x}_3 + x_4 \log(x_1)$$

$$f_4(x) = \cos(x_1) \log(x_1) + x_2 x_3 x_4$$
$$g_4(x) = \cos(x_5) \log(x_1) + x_2 x_1 \mathrm{x}_3$$

$$f_5(x) = x_5 x_2 \cos(x_3) + x_4 x_1 \log(x_1)$$
$$g_5(x) = x_2 x_4 \mathrm{x}_5 + \mathrm{x}_3 \cos(\mathrm{x}_2)$$

1)<u>Γραμμικοποιήστε το σύστημα γύρω από ένα σημείο ισορροπίας</u>

Για να γραμμικοποιήσω το σύστημα γύρω από ένα σημείο ισορροπίας , πρέπει να χρησιμοποιήσω το ανάπτυγμα Taylor.

Επίσης θα θεωρήσω το σημείο ισορροπίας x_0 και u_0 όπου :

$$x_0 = [x_{01}, x_{02}, x_{03}, x_{04}, x_{05}] = \left[1, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}\right]$$

$$u_0 = [u_{01}, u_{02}, u_{03}, u_{04}, u_{05}] = [u_{05}] = 0$$

Το σύστημα είναι μη γραμμικό λόγω της ύπαρξης του cos(x) και log(x) στη μεταβλητή κατάσταση $\dot{x}_5$.

Η $\dot{x}$ είναι της μορφής $\dot{x} = F(x, u)$ άρα το ανάπτυγμα Taylor θα είναι :

$$\dot{x} = F(x_0) + \frac{dF}{dx}\Big|_{\substack{x=x_0 \\ u=u_0}}(x - x_0) + \frac{dF}{du}\Big|_{\substack{x=x_0 \\ u=u_0}}(u - u_0)$$

όπου $F(x_0) = 0$.

$$A_{nxn} = \frac{dF}{dx}\Big|_{\substack{x=x_0 \\ u=u_0}} = \begin{bmatrix} \frac{dF_1}{dx_1} & \cdots & \frac{dF_1}{dx_5} \\ \vdots & \ddots & \vdots \\ \frac{dF_5}{dx_1} & \cdots & \frac{dF_5}{dx_5} \end{bmatrix}$$

Αναλυτικά :

$\frac{dF_1}{dx_1} = 0, \frac{dF_1}{dx_2} = 1 , \frac{dF_1}{dx_3} = 0 , \frac{dF_1}{dx_4} = 0 , \frac{dF_1}{dx_5} = 0$

$\frac{dF_2}{dx_1} = 0, \frac{dF_2}{dx_2} = 0 , \frac{dF_2}{dx_3} = 1 , \frac{dF_2}{dx_4} = 0 , \frac{dF_2}{dx_5} = 0$

$\frac{dF_3}{dx_1} = 0, \frac{dF_3}{dx_2} = 0 , \frac{dF_3}{dx_3} = 0 , \frac{dF_3}{dx_4} = 1 , \frac{dF_3}{dx_5} = 0$

$\frac{dF_4}{dx_1} = 0, \frac{dF_4}{dx_2} = 0 , \frac{dF_4}{dx_3} = 0 , \frac{dF_4}{dx_4} = 0 , \frac{dF_4}{dx_5} = 1$

$\frac{dF_5}{dx_1} = 18\left(x_3 x_5 + \frac{x_2}{x_1}\right) + 7\left(x_3\cos(x_4) + \frac{x_2}{x_1}\right) + 18\cos(x_2) + 19\left(-\sin(x_1)\log(x_1) + \frac{\cos(x_1)}{x_1}\right) + 9x_4 + 9x_4\log(x_1)$

Αν βάλω τις τιμές x_0 και u_0 θα έχω :

$$\boldsymbol{\frac{dF_5}{dx_1} = 93.82}$$

Αντίστοιχα και για τις υπόλοιπες παραγωγίσεις :

$\frac{dF_5}{dx_2} = 18\log(x_1) + 7\log(x_1) + 18(x_3x_4 - x_1\sin(x_2)) + 19x_3x_4 + 9x_5\cos(x_3)$

$$\boldsymbol{\frac{dF_5}{dx_2} = 63.01}$$

$\frac{dF_5}{dx_3} = 18x_1x_5 + 7x_1\cos(x_4) + 18x_2x_4 + 19x_2x_4 - 9\sin(x_3)x_5x_2$

$$\boldsymbol{\frac{dF_5}{dx_3} = 97.36}$$

$$\frac{dF_5}{dx_4} = -7x_1x_3\sin(x_4) + 18x_2x_3 + 19x_2x_3 + 9x_1\log(x_1)$$

$$\frac{dF_5}{dx_4} = 80.3$$

$\frac{dF_5}{dx_5} = 18x_1x_3 + 9x_2\cos(x_3)$

$$\boldsymbol{\frac{dF_5}{dx_5} = 28.27}$$

Άρα ο πίνακας θα γίνει :

$$A_{nxn} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 97.82 & 63.01 & 97.36 & 80.3 & 28.27 \end{bmatrix}$$

Για το $\frac{dF}{du}$ υπάρχει μόνο 1 στοιχείο :

$$\begin{aligned} \frac{dF}{du}\Big|_{\substack{x=x_0 \\ u=u_0}} &= \big(b_1g_1(x) + b_2g_2(x) + b_3g_3(x) + b_4g_4(x) \\ &\quad + b_5g_5(x)\big)^2 \\ &= (18x_4x_3\cos(x_2) + 18x_1x_2\log(x_1) \\ &\quad + 15x_4x_5\log(x_1) + 15x_2x_4\cos(x_2) + 10x_1x_5x_3 \\ &\quad + 10x_4\log(x_1) + 9\cos(x_5)\log(x_1) + 9x_1x_2x_3 \\ &\quad + 13x_2x_4x_5 + 15x_3\cos(x_2))^2 = \mathbf{9460.52} \end{aligned}$$

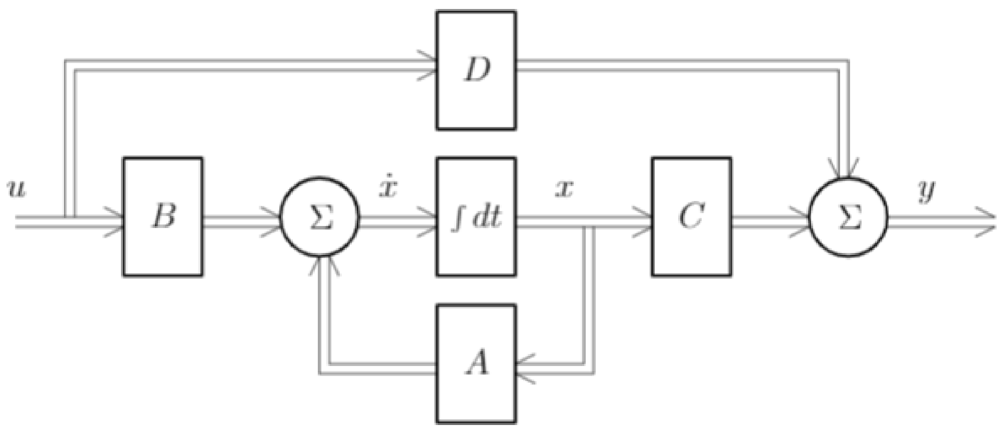
Αν κάνω την ίδια διαδικασία για y = x$_1$ θα έχω $\dot{\boldsymbol{y}} = \mathbf{1}$.

Άρα το γραμμικοποιημένο σύστημα θα έχει τη μορφή

$$\dot{x} = 0 + \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 97.82 & 63.01 & 97.36 & 80.3 & 28.27 \end{bmatrix} (x - x_0) + 9460.52(u - u_0)$$

⇨ $\dot{x} = A\delta x$

⇨ $y = 1\delta x$



Γραφική παράσταση των εξισώσεων κατάστασης

2)

A)<u>Υπολόγισε συνάρτηση μεταφοράς γραμμικού συστήματος</u>

$$\dot{x} = Ax + Bu$$

$$y = Cx$$

Ο μετασχηματισμός Laplace της κατάστασης εξισώσεων είναι

$$sX(s) = AX(s) + BU(s)$$

- ⇨ $(s - A)X(s) = AX(s) + BU(s)$
- ⇨ $X(s) = \frac{B}{s-A}U(s)$
- ⇨ Αντικαθιστώ το X(s) στην κατάσταση εξισώσεων Y :
- ⇨ $Y(s) = CX(s)$
- ⇨ $Y(s) = \frac{BC}{s-A}U(s)$
- ⇨ Άρα η συνάρτηση μεταφοράς είναι
- ⇨ $H(s) = \frac{Y(s)}{U(s)} = \frac{\frac{BC}{s-A}U(s)}{U(s)} = \frac{BC}{s-A}$
- ⇨ $O\ BC\ πίνακας\ βγάζει\ αποτέλεσμα\ 9460.52$
- ⇨ $[sI - A]^{-1} = \left[\begin{bmatrix} s & 0 & 0 & 0 & 0 \\ 0 & s & 0 & 0 & 0 \\ 0 & 0 & s & 0 & 0 \\ 0 & 0 & 0 & s & 0 \\ 0 & 0 & 0 & 0 & s \end{bmatrix} - \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 97.82 & 63.01 & 97.36 & 80.3 & 28.27 \end{bmatrix}\right]^{-1} = \begin{bmatrix} s & -1 & 0 & 0 & 0 \\ 0 & s & -1 & 0 & 0 \\ 0 & 0 & s & -1 & 0 \\ 0 & 0 & 0 & s & -1 \\ 97.82 & 63.01 & 97.36 & 80.3 & 28.27 \end{bmatrix}$

Μέσω κώδικα βρίσκω την open – loop TF στο MATLAB:

```
clear;
clc;
```

```
syms s Kd Ki Kp

A = [0 1 0 0 0;0 0 1 0 0;0 0 0 1 0;0 0 0 0 1;97.82 63.01
97.36 80.3 28.27];
B = [0;0;0;0;9460.52];
C = [1 0 0 0 0];
D = 0;

[b,a] = ss2tf(A,B,C,D);
sys   = tf(b,a);

closed_loop_tf = feedback(sys,1);
```

$$H_{op}(s) = \frac{9461}{s^5 - 28.27s^4 - 80.3s^3 - 97.36s^2 - 63.01s - 97.82}$$

PID Ελεγκτής

Η συνάρτηση μεταφοράς του PID ελεγκτή έχει την μορφή :

$$G(s) = K_p + K_d s + \frac{K_i}{s}$$

Αφού έχουμε τη συνάρτηση ανοιχτού βρόχου $H_{op}(s)$, η συνάρτηση μεταφοράς κλειστού βρόχου είναι η εξής :

$$T_s = \frac{G(s)H_{op}(s)}{1 + G(s)H_{op}(s)} =>$$

$$T_s = \frac{9461(K_p + K_d s + \frac{K_i}{s})}{s^6 - 28.27s^5 - 80.3s^4 - 97.36s^3 + (9461K_d - 63.01)s^2 + (9461K_p - 97.82)s + 9461K_i}$$

Βελτιστοποίηση PID Ελεγκτή

Από την συνάρτηση μεταφοράς **κλειστού βρόχου** παίρνω τον παρονομαστή , που έχει τη μορφή :

$$H(s) = s^6 - 28.27s^5 - 80.3s^4 - 97.36s^3 + (9461K_d - 63.01)s^2 + \left(9461K_p - 97.82\right)s + 9461K_i$$

Παρατηρώ πως πρόκειται για ένα πολυώνυμο $6^{ου}$ βαθμού άρα θα έχει το πολύ 6 διαφορετικούς πόλους.Θα ορίσω 3 πόλους μέσω των οποίων θα μπορώ να βρω τα K_p , K_d , K_i και μετά θα βρω τους υπόλοιπους.

Επιθυμώ το πραγματικό μέρος των πόλων μου να είναι όσο πιο αρνητικό γίνεται,δηλαδή όσο πιο κοντά στο $-\infty$.

Για να προσπαθήσω να βρω αρνητικές ρίζες οι οποίες θα μου δώσουν τις τιμές του PID ελεγκτή θα υλοποιήσω έναν κώδικα στο Matlab στον οποίο θα ορίζω 3 αρχικές ρίζες (και οι 3 αρνητικές για να επιτύχω στο τέλος όλες οι ρίζες να είναι αρνητικές) και θα υπολογίζω τα Kp,Kd,Ki για αυτές.

Στη συνέχεια θα βρίσκω τις άλλες 3 ρίζες από τον παρονομαστή της συνάρτησης μεταφοράς κλειστού βρόχου , αφού πλέον θα έχω τα Kp,Kd,Ki.

$s_1 = -4$, $s_2 = -5$, $s_3 = -6$:

Οι τιμές του PID ελεγκτή είναι :

K_p = -29.10

K_d = 4.22

K_i = 54.4

Και οι ρίζες είναι :

x_1 = 5

x_2 = 4

x_3 = 6

x_4 = 29.89

x_5 = -8.312

x_6 = -8.312

Θα συμβουλευτώ τώρα το θεώρημα του Vieta.Σύμφωνα με αυτό το θεώρημα το άθροισμα των ριζών ενός πολυωνύμου είναι ίσο με $-\frac{a_{n-1}}{a_n}$,όπου α_n και α_{n-1} είναι οι συντελεστές του πολυωνύμου :

$$P(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$$

Άρα το άθροισμα των ριζών μας αναμένουμε να είναι ίσο με :

$$-\frac{a_{n-1}}{a_n} = -\left(-\frac{28.27}{1}\right) = 28.27$$

Οπότε έιναι πρακτικά αδύνατο να έχουμε ιδανικά 6 αρνητικούς πόλους.Επιθυμούμε τουλάχιστον όσοι πόλοι προκύπτουν να είναι κοντά στο 0.

Μετά από αυτή τη διαπίστωση θα αλλάξω τακτική και θα επιχειρήσω να σκανάρω τιμές στο εύρος [-1,1] για τους 3 αρχικούς πόλους,ώστε ναι μεν να είναι αρνητικοί ή ελάχιστα θετικοί για να μην ωθήσουν τον ή τους πόλους που είναι θετικοί σε ακόμα μεγαλύτερες θετικές τιμές.

Ο κώδικας Matlab:

```
close all;
clear;
clc;

a = 5000;
b = 0;
for s1 = -0.1:0.01:0.1 %σκανάρω τιμές για τον 1ο πόλο
    for s2 = -0.1:0.01:0.1 %σκανάρω τιμές για τον 2ο πόλο
        for s3 = -0.1:0.01:0 %σκανάρω τιμές για τον 3ο πόλο
        syms Ki Kd Kp;


        %βάσει των πόλων υπολογίζω τα Kp,Kd,Ki
        eq1 = (s1)^6 -28.27*(s1)^5 -80.3*(s1)^4 -
97.36*(s1)^3 +(9461*Kd-63.01)*(s1)^2 + (9461*Kp-
97.82)*(s1)+9461*Ki == 0;
   eq2 = (s2)^6 -28.27*(s2)^5 -80.3*(s2)^4 -97.36*(s2)^3
+(9461*Kd-63.01)*(s2)^2 + (9461*Kp-97.82)*(s2)+9461*Ki ==
0;
        eq3 = (s3)^6 -28.27*(s3)^5 -80.3*(s3)^4 -
97.36*(s3)^3 +(9461*Kd-63.01)*(s3)^2 + (9461*Kp-
97.82)*(s3)+9461*Ki == 0;
        sol = solve ([eq1,eq2,eq3] , [Ki Kd Kp]);
        Ki  = double(sol.Ki);
        Kd  = double(sol.Kd);
        Kp  = double(sol.Kp);
        %υπολογίζω τους υπόλοιπους 3 πόλους
        x = roots([1 -28.27 -80.3 -97.36 (9461*Kd-63.01)
(9461*Kp-97.82)  9461*Ki]);

        %κρατάω το πραγματικό μέρος των πόλων
        roots_real_part = real(x);
        %επιλέγω το μέγιστο
        max_real_part = max(roots_real_part);
        b = b+1; %counter των επαναλήψεων

       %εάν η μέγιστη τιμή είναι μικρότερη της προηγούμενης
       %κρατάω τις μέχρι εκείνη τη στιγμή καλύτερες λύσεις
         if max_real_part < a
             a = max_real_part;
```

```
                roots_best = roots_real_part;
                Kd_best = Kd;
                Kp_best = Kp;
                Ki_best = Ki;
            end
           end
       end
end
%τιμές του ελεγκτή και ρίζες
roots_best
Kd_best
Kp_best
Ki_best
```

Με τον κώδικα MATLAB βρήκα 5 αρνητικούς πόλους και 1 πολύ θετικό.

$x_1 = 30.965$

$x_2 = -1.3175$

$x_3 = -1.3175$

$x_4 = -0.03$

$x_5 = -0.02$

$x_6 = -0.01$

Και οι αντίστοιχες τιμές για τον PID ελεγκτή είναι

$K_d = 0.006$

$K_p = 0.01$

$Ki = -5.87 * 10^{-8}$

```
roots_best =

  30.964982501318783
  -1.317491250659395
  -1.317491250659395
  -0.030000000000014
  -0.019999999999981
  -0.010000000000005


Kd_best =

   0.006063481945883


Kp_best =

   0.010328469966251


Ki_best =

    -5.873338864813438e-08
```

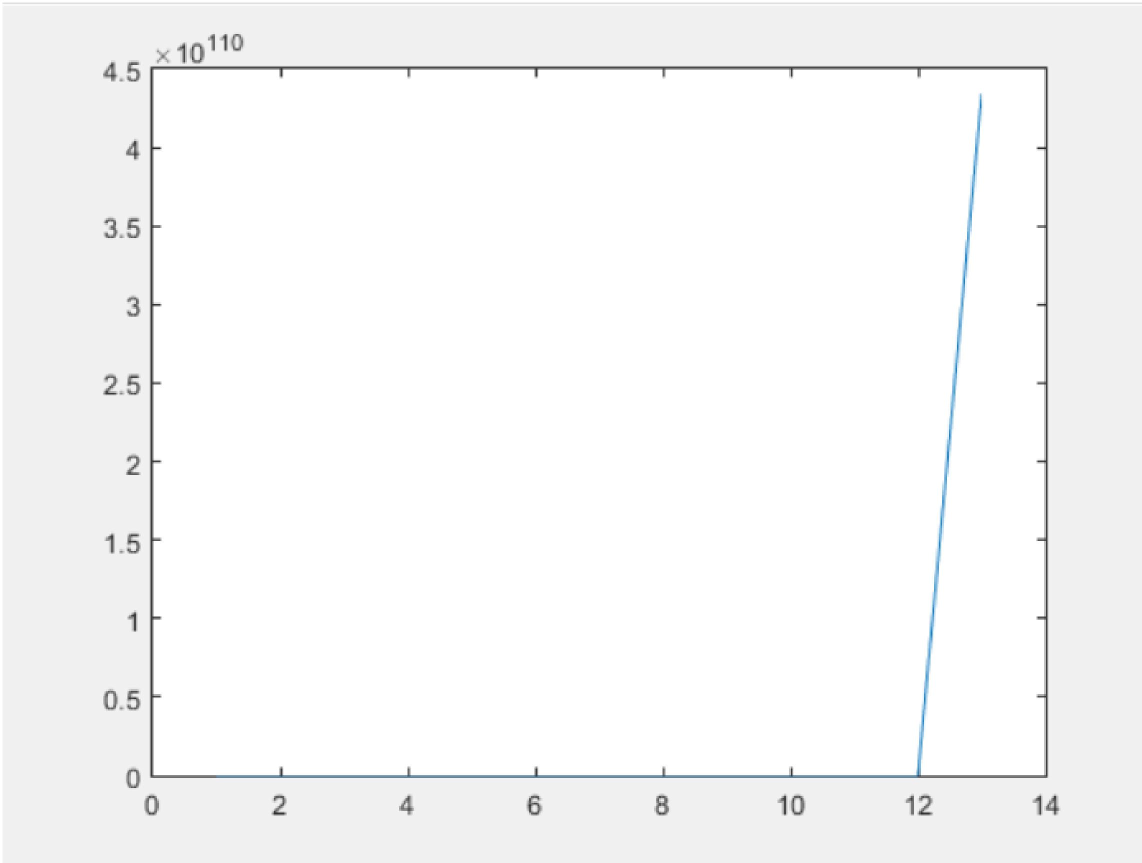
B)

Προσομοίωση μη γραμμικού συστήματος PID ελεγκτή

Προσομοιώνω το μη γραμμικό σύστημα με τον παρακάτω κώδικα Matlab :

```
close all;
clear;
clc;
format long;
%PID ελεγκτής
Kd = 0.006063481945883;
Kp = 0.010328469966251;
Ki = -5.873338864813438e-08;
%χρόνος προσομοίωσης
dt = 0.0001;
int_y = 0;
prev_y = 0;
%πίνακας σημείου ισορροπίας
x = [1 pi/2 pi/2 pi/2 pi/2];
for i = 1:5500
    y = x(1);
    int_y = y*dt+int_y;
    dy = (y-prev_y)/dt;
    u = Kd*dy + Kp + Ki*int_y;

    x1_dot = x(2);
    x2_dot = x(3);
    x3_dot = x(4);
    x4_dot = x(5);
    x5_dot = 18*x(1)*x(3)*x(5) +18*log(abs(x(1)+2)) + 7*x(1)*x(3)*cos(x(4))
+7*x(1)*x(2)*log(abs(x(1)+2)) +18*x(2)*x(3)*x(4) +18*x(1)*cos(x(2))
+19*cos(x(1))*log(abs(x(1)+2)) +19*x(4)*x(2)*x(3) +9*x(5)*x(2)*cos(x(3))
+9*x(4)*x(1)*log(abs(x(1)+2)) +(( 18*x(4)*x(3)*cos(x(2))
+18*x(1)*x(2)*log(abs(x(1)+2))+15*x(4)*x(5)*log(abs(x(1)+2))+15*x(2)*x(4)*cos(x
(2))+10*x(1)*x(5)*x(3)+10*x(4)*log(abs(x(1)+2))+9*cos(x(5))*log(abs(x(1)+2))+9*
x(1)*x(2)*x(3)+13*x(2)*x(4)*x(5)+13*x(3)*cos(x(2)))^2)*u;

    x(1) = x(1)+dt*x1_dot;
    x(2) = x(2)+dt*x2_dot;
    x(3) = x(3)+dt*x3_dot;
    x(4) = x(4)+dt*x4_dot;
    x(5) = x(5)+dt*x5_dot;
    prev_y = y;
    y_plot(i) = real(y);
end

plot(y_plot)
```

Η γραφική παράσταση της y(x)



Παρατηρώ πως ο ελεγκτής απειρίζει το μη γραμμικό σύστημα στις πρώτες 12 επαναλήψεις.

C) Τυχαίες μεταβολές PID ελεγκτή

Είδαμε ότι οι τωρινοί PID ελεγκτές δεν μπορούν να προσομοιώσουν το γραμμικό σύστημα.

Με αυτή τη μέθοδο θα δοκιμάσω να μεταβάλλω τυχαία τους PID ελεγκτές ώστε να δω αν μπορούν να προσομοιώσουν καλύτερα το μη γραμμικό σύστημα.

Για να αξιολογώ πόσο καλά ανταποκρίνεται το σύστημα στους PID ελεγκτές θα υπολογίζω κάθε φορά την τιμή μιας συνάρτησης σφάλματος,η οποία θα αντιστοιχεί **στο ολοκλήρωμα της απόλυτης τιμής της εξόδου y**.

Αυτή την συνάρτηση θα προσπαθώ να ελαχιστοποιώ σε κάθε επανάληψη με διαφορετικούς PID ελεγκτές.

<u>Υπερπαράμετροι</u>

Αρχικές τιμές ελεγκτή $K_{p,d,i} = [\,0 \quad 0 \quad 0\,]$

Εύρος τυχαίων μεταβολών [-1000,1000]

Επαναλήψεις *100,000*

Ανώτατο όριο συνάρτησης κόστους 10^9

Βήμα προσομοίωσης $dt = 10^{-3}$

Βήματα *40,000*

<u>Κώδικας Matlab</u>

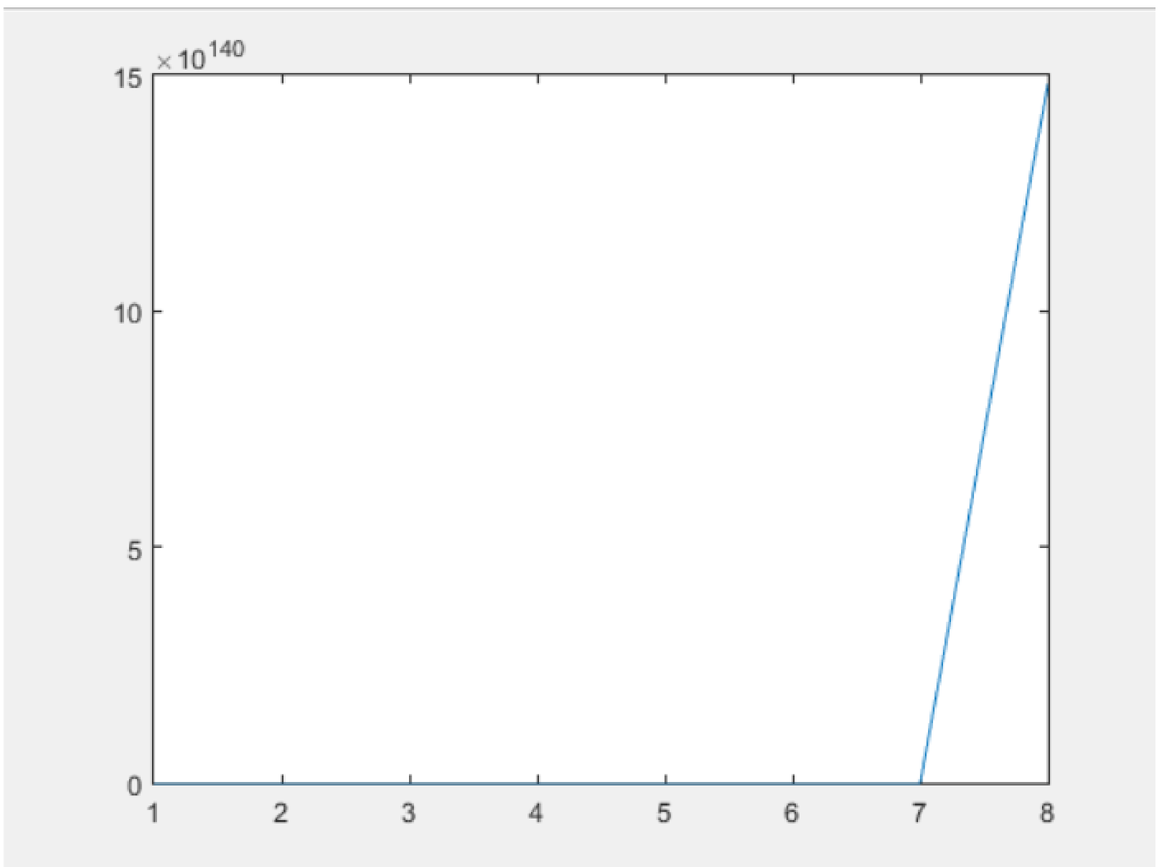
```
close all;
clear;
clc;
format long;

n = 100000;

%οριο για να κάνει break το loop της επανάληψης
tol = 1000000000; %10^9

%%%%%%%%%%%%%%-------ΑΡΧΙΚΟΠΟΙΗΣΕΙΣ ΠΑΡΑΜΕΤΡΩΝ ΕΛΕΓΚΤΗ-------
%%%%%%%%%%%%%%

%αρχικοποιώ τα Kp,Kd,Ki
Kd_initial = 0;
Kp_initial = 0;
Ki_initial = 0;

%αρχικοποιώ τα Kp_best,Kd_best,Ki_best
Kd_best = Kd_initial;
Kp_best = Kp_initial;
Ki_best = Ki_initial;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%ο πίνακας performance κρατάει τα performance κάθε επανάληψης και το
%best_perf το καλύτερο performance
performance = 0;
best_perf = 0;


%loop επαναλήψεων μέχρι να ωθήσω την έξοδο y στο 0 με τον καλύτερο
ελεγκτή
for k = 1:(n-1)
%αν βρίσκομαι στην πρώτη επανάληψη τότε χρησιμοποιώ τις αρχικές τιμές
που έχω ορίσει
Kd(1) = Kd_initial;
Kp(1) = Kp_initial;
Ki(1) = Ki_initial;


%βήμα της προσομοίωσης
dt = 0.001;


%αρχικοποιώ την μεταβλητή της έξοδου
y = 0;
%αρχικοποιώ τις μεταβλητές του ολοκληρώματος και της προηγούμενης τιμή
της εξόδου
int_y = 0;
prev_y = 0;
%πίνακας σημείου ισορροπίας
x = [1 0.05 0.01 0.17 0.08];
```

```
%ολοκλήρωμα του performance που χρησιμεύει ως συνάρτηση βελτιστοποίησης
του
%ελεγκτή
perf_int = 0;

 for i = 1:40000
    y = x(1);
    int_y = y*dt+int_y;
    dy = (y-prev_y)/dt;
    u = Kd(k)*dy + Kp(k) + Ki(k)*int_y;

    x1_dot = x(2);
    x2_dot = x(3);
    x3_dot = x(4);
    x4_dot = x(5);
    x5_dot = 18*x(1)*x(3)*x(5) +18*log(abs(x(1)+2)) +
7*x(1)*x(3)*cos(x(4)) +7*x(1)*x(2)*log(abs(x(1)+2)) +18*x(2)*x(3)*x(4)
+18*x(1)*cos(x(2)) +19*cos(x(1))*log(abs(x(1)+2)) +19*x(4)*x(2)*x(3)
+9*x(5)*x(2)*cos(x(3)) +9*x(4)*x(1)*log(abs(x(1)+2)) +((
18*x(4)*x(3)*cos(x(2))
+18*x(1)*x(2)*log(abs(x(1)+2))+15*x(4)*x(5)*log(abs(x(1)+2))+15*x(2)*x(
4)*cos(x(2))+10*x(1)*x(5)*x(3)+10*x(4)*log(abs(x(1)+2))+9*cos(x(5))*log
(abs(x(1)+2))+9*x(1)*x(2)*x(3)+13*x(2)*x(4)*x(5)+13*x(3)*cos(x(2)))^2)*
u;



    x(1) = x(1)+dt*x1_dot;
    x(2) = x(2)+dt*x2_dot;
    x(3) = x(3)+dt*x3_dot;
    x(4) = x(4)+dt*x4_dot;
    x(5) = x(5)+dt*x5_dot;
    prev_y = y;

    perf_int = abs(y) + perf_int;
    if(perf_int > tol)
        break;
    end

 end
    performance(k) = perf_int;

    %στην 1η επανάληψη αρχικοποιώ ως το best performance
    %το 1o performance
    if(k == 1)
        best_perf = performance(k);
    end







    %elegxos an to performance meiwnetai i oxi
if(performance(k) < best_perf)
```

```
    best_perf = performance(k);
    Kd_best = Kd(k);
    Kp_best = Kp(k);
    Ki_best = Ki(k);

    iteration_of_best_perf = k;
end

   %ορίζω τυχαίες μεταβολές για το K
   disp_Kp = 1000 - 2000*rand;
   disp_Kd = 1000 - 2000*rand;
   disp_Ki = 1000 - 2000*rand;

   %υπολογίζω τα K της επόμενης επανάληψης
   Kp(k+1) = Kp_best + disp_Kp;
   Kd(k+1) = Kd_best + disp_Kd;
   Ki(k+1) = Ki_best + disp_Ki;
end


%%%%%%%%%%%%%%%------------ΑΠΕΙΚΟΝΙΖΩ ΤΗΝ ΕΞΟΔΟ Υ ΓΙΑ ΤΟΝ ΚΑΛΥΤΕΡΟ
ΕΛΕΓΚΤΗ------------%%%%%%%%%%%%%%%

%πίνακας σημείου ισορροπίας
 x = [1 0.05 0.01 0.17 0.08];
%βήμα της προσομοίωσης
 dt = 0.001;

  for i = 1:60000
    y = x(1);
    int_y = y*dt+int_y;
    dy = (y-prev_y)/dt;
    u = Kd_best*dy + Kp_best + Ki_best*int_y;

    x1_dot = x(2);
    x2_dot = x(3);
    x3_dot = x(4);
    x4_dot = x(5);
    x5_dot = 18*x(1)*x(3)*x(5) +18*log(abs(x(1)+2)) +
7*x(1)*x(3)*cos(x(4)) +7*x(1)*x(2)*log(abs(x(1)+2)) +18*x(2)*x(3)*x(4)
+18*x(1)*cos(x(2)) +19*cos(x(1))*log(abs(x(1)+2)) +19*x(4)*x(2)*x(3)
+9*x(5)*x(2)*cos(x(3)) +9*x(4)*x(1)*log(abs(x(1)+2)) +((
18*x(4)*x(3)*cos(x(2))
+18*x(1)*x(2)*log(abs(x(1)+2))+15*x(4)*x(5)*log(abs(x(1)+2))+15*x(2)*x(
4)*cos(x(2))+10*x(1)*x(5)*x(3)+10*x(4)*log(abs(x(1)+2))+9*cos(x(5))*log
(abs(x(1)+2))+9*x(1)*x(2)*x(3)+13*x(2)*x(4)*x(5)+13*x(3)*cos(x(2)))^2)*
u;
    x(1) = x(1)+dt*x1_dot;
    x(2) = x(2)+dt*x2_dot;
    x(3) = x(3)+dt*x3_dot;
    x(4) = x(4)+dt*x4_dot;
    x(5) = x(5)+dt*x5_dot;
    prev_y = y;

    y_plot(i) = real(y);
```

```
 end


%apeikonizw tis veltistes times
best_perf
iteration_of_best_perf
Kp_best
Kd_best
Ki_best

plot(y_plot)
```

Αποτελέσματα




```
best_perf =

     1.000051923112337e+11


iteration_of_best_perf =

      225456
```

Τιμές PID ελεγκτή

```
Kp_best =

     2.540362902925693e+02


Kd_best =

    -1.456363293346622e+02


Ki_best =

     3.710072543158735e+02
```

Καταλήγω λοιπόν στο συμπέρασμα πως οι PID ελεγκτές δεν μπορούν να προσομοιώσουν καλά το σύστημα καθώς το σφάλμα παραμένει υψηλότατο.

3)

Ελεγκτής u = Kx για προσομοίωση με το μη γραμμικό σύστημα

Ο ελεγκτής αυτός είναι ένας ελεγκτής βέλτιστου ελέγχου.Στον βέλτιστο έλεγχο επιδιώκουμε η απόδοση να είναι εκτός από αποδεκτή και βέλτιστη,δηλαδή να ελαχιστοποιήσουμε το σφάλμα παντού.

Ο βέλτιστος νόμος ελέγχου (Linear Quadratic Regulator – LQR) προκύπτει με ανατροφοδότηση κατάστασης

$$u = -Kx(t)$$

Το κέρδος του ελεγκτή δίνεται από

$$K = -R^{-1}B^{T}P$$

όπου P είναι συμμετρική θετικά ημιορισμένη λύση της αλγεβρικής εξίσωσης Riccati

$$A^{T}P + PA - PBR^{-1}B^{T}P + Q = 0$$

Η συνάρτηση κόστους ορίζεται ως

$$J = \frac{1}{2}\int_{0}^{\propto}(x^{T}Qx + u^{T}Ru)dt$$

Η ενέργεια του συστήματος σχετίζεται με τον παράγοντα $x^{T}Qx$.Κατά τη μεταβατική κατάσταση,πρέπει η ενέργεια να πέφτει γρήγορα στο μηδέν.Η μέγιστη τιμή της σχετίζεται με την υπερακόντιση,ενώ ο χρόνος μείωσης της ενέργειας στο μηδέν σχετίζεται με τον χρόνο αποκατάστασης(settling time).Η ενέργεια ελέγχου σχετίζεται με τον παράγοντα $u^{T}Ru$.

Αρχικά,θα υπολογίσω τον ελεγκτή K στο γραμμικό σύστημα για να τον εφαρμόσω στη συνέχεια στο μη γραμμικό.

Υλοποίηση με MATLAB

```
close all;
clear;
clc;


%εισάγω τους πίνακες
A = [0 1 0 0 0;0 0 1 0 0;0 0 0 1 0;0 0 0 0 1;97.82 63.01 97.36
80.3 28.27];
B = [0;0;0;0;9460.52];
C = [1 0 0 0 0];
D = 0;

%θέτω τους πίνακες για την LQR μέθοδο
Q = eye(5);
R = [1];

%υπολογίζω τον ελεγκτή
[K,M,E] = lqr(A,B,Q,R)

%υπολογίζω την απόκριση
sys2 = ss(A-B*K,B,C,D);
%υπολογίζω το διάνυσμα του χρόνου
tt = 0:0.01:8;

%προσομοιώνω την απόκριση
[y2,t2,x2] = initial(sys2,[1;pi/2;pi/2;pi/2;pi/2]',tt);

%υπολογίζω την είσοδο ελέγχου
u = -K*x2';

%plotting
subplot(211)
plot(t2,x2(:,1),'r-.'); hold on
plot(t2,x2(:,2),'k--'); hold on
plot(t2,x2(:,3),'y--'); hold on
plot(t2,x2(:,4),'m--'); hold on
plot(t2,x2(:,5),'b--');
legend('x1','x2', 'x3','x4','x5');
legend('Location','southeast','Orientation','horizontal');
grid; ylabel('states')
subplot(212)
plot(t2,u);
```

```
grid;
ylabel('control');
xlabel('time [s]');
title('LQR design ex. 2')
```

<u>Άρα το κέρδος του ελεγκτή είναι</u>

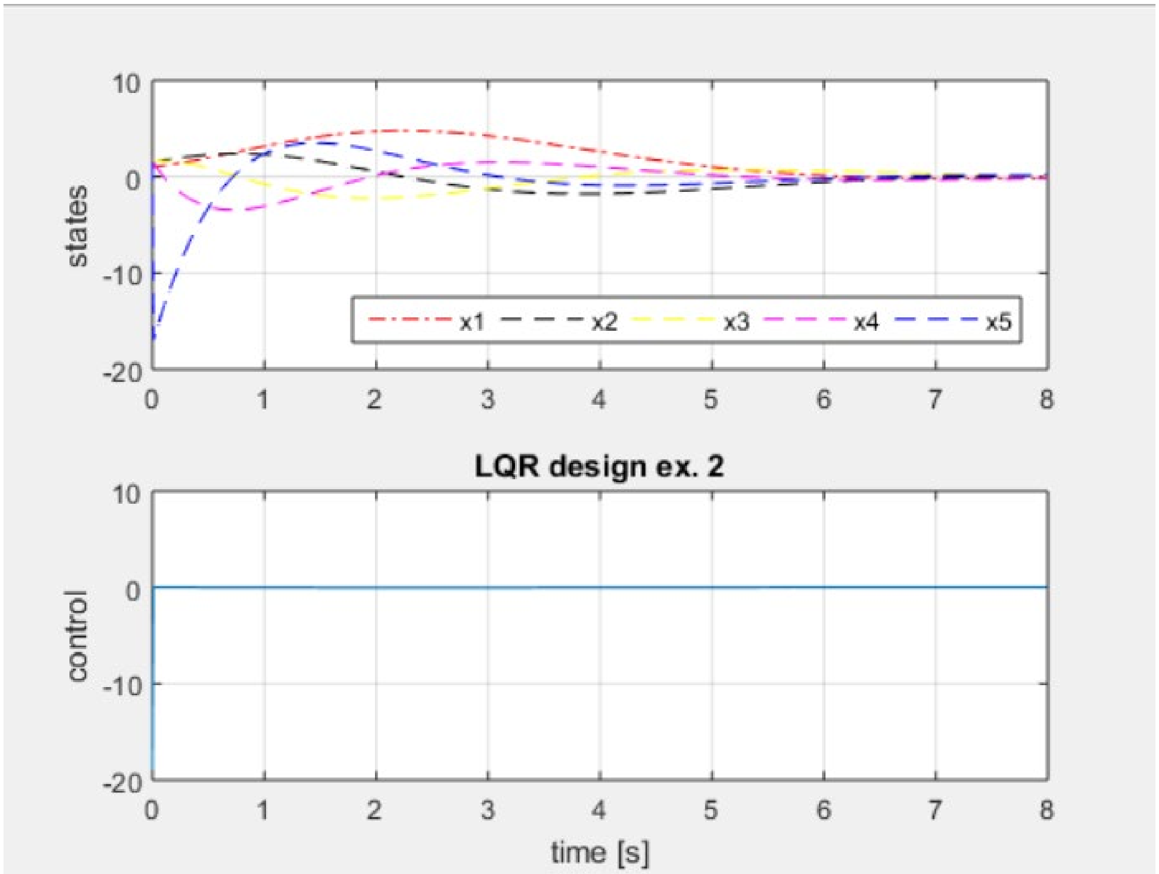
$$K = \begin{matrix} 1.010393266620135 & 3.084515805883382 \\ 4.246746186325314 & 3.086648109772531 \end{matrix}$$

<u>Οι ιδιοτιμές του συστήματος κλειστού βρόχου είναι</u>

$$E = \begin{matrix} -9.46 * 10^3 & -0.9511 + i0.3091 \\ -0.9511 + i0.309 & -0.5878 + i0.8090 \\ & -0.5878 + i0.8090 \end{matrix}$$

οι οποίες δείχνουν πως το σύστημα είναι ευσταθές.

Επίσης παραθέτω και γραφική παράσταση για τις αποκρίσεις των εισόδων και την εντολή ελέγχου πάντα για το γραμμικό σύστημα.



Προσομοίωση μη γραμμικού συστήματος με LQR ελεγκτή

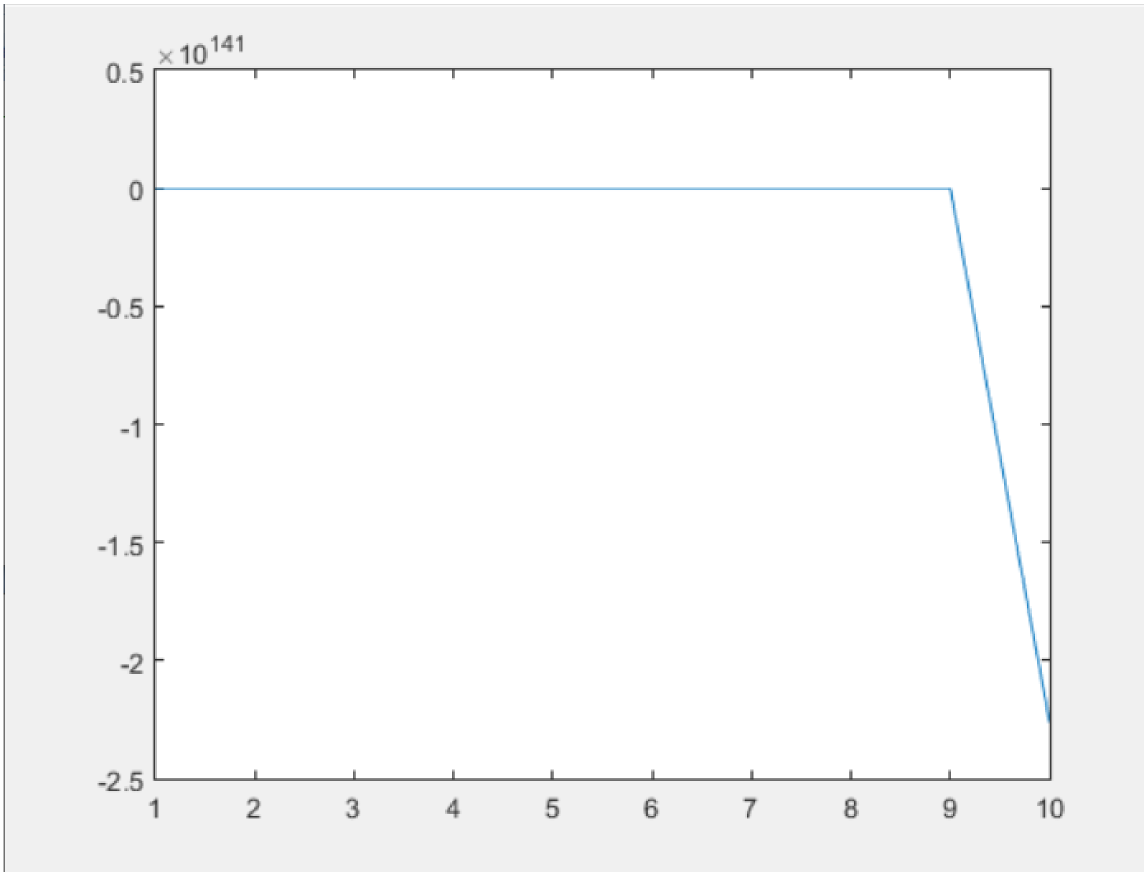
Για να απεικονίσω τη συμπεριφορά του συστήματος με τον LQR ελεγκτή που δουλεύει στο γραμμικό σύστημα,θα υλοποιήσω παρόμοιο κώδικα με το ερώτημα της προσομοίωσης μη γραμμικού συστήματος με PID ελεγκτή :

Βήμα προσομοίωσης dt = 0.001

Βήματα 5500

Κώδικας Matlab

```
close all;
clear;
clc;
format long;
%πίνακας τιμών u = -Kx ελεγκτή
K = [1.010393266620135 3.084515805883382 4.246746186325314
3.086648109772531 1.003318884077076];
%βήμα προσομοίωσης
dt = 0.0001;

%πίνακας τιμών σημείου ισορροπίας
x = [1 pi/2 pi/2 pi/2 pi/2];
for i = 1:5500
    y = x(1);
    u = -(K(1)*x(1) + K(2)*x(2) + K(3)*x(3) + K(4)*x(4) +
K(5)*x(5));

    x1_dot = x(2);
    x2_dot = x(3);
    x3_dot = x(4);
    x4_dot = x(5);
    x5_dot = 18*x(1)*x(3)*x(5) +18*log(abs(x(1)+2)) +
7*x(1)*x(3)*cos(x(4)) +7*x(1)*x(2)*log(abs(x(1)+2))
+18*x(2)*x(3)*x(4) +18*x(1)*cos(x(2))
+19*cos(x(1))*log(abs(x(1)+2)) +19*x(4)*x(2)*x(3)
+9*x(5)*x(2)*cos(x(3)) +9*x(4)*x(1)*log(abs(x(1)+2)) +((
18*x(4)*x(3)*cos(x(2))
+18*x(1)*x(2)*log(abs(x(1)+2))+15*x(4)*x(5)*log(abs(x(1)+2))+15*x
(2)*x(4)*cos(x(2))+10*x(1)*x(5)*x(3)+10*x(4)*log(abs(x(1)+2))+9*c
os(x(5))*log(abs(x(1)+2))+9*x(1)*x(2)*x(3)+13*x(2)*x(4)*x(5)+13*x
(3)*cos(x(2)))^2)*u;

    x(1) = x(1)+dt*x1_dot;
    x(2) = x(2)+dt*x2_dot;
    x(3) = x(3)+dt*x3_dot;
    x(4) = x(4)+dt*x4_dot;
    x(5) = x(5)+dt*x5_dot;

    y_plot(i) = real(y);
end

plot(y_plot)
```



Παρατηρώ ότι ούτε αυτός ο ελεγκτής δεν μπορεί να ωθήσει την έξοδο του συστήματος στο 0 , και μετά από 10 επαναλήψεις η έξοδος απειρίζεται.

Προσομοίωση μη γραμμικού συστήματος με διαφορετικούς LQR ελεγκτές και τυχαίες μεταβολές

Σε αυτό το κομμάτι θα προσπαθήσω να βρω τον καλύτερο LQR ελεγκτή και να μηδενίσω την έξοδο θέτοντας τον ελεγκτή σε μια αρχική τιμή και προσομοιώνοντας το σύστημα αρκετές φορές η τιμή της συνάρτησης κόστους να ελαττώνεται διαρκώς ώστε να βελτιστοποιείται ο ελεγκτής.

Υπερπαράμετροι

Αρχικές τιμές ελεγκτή $K = [\,0 \quad 0 \quad 0 \quad 0\,]$

Εύρος τυχαίων μεταβολών [-100,100]

Επαναλήψεις *50,000*

Ανώτατο όριο συνάρτησης κόστους 10^7

Βήμα προσομοίωσης $dt = 10^{-3}$

Βήματα *30,000*

Στον κώδικα που υλοποίησα χρησιμοποιώ ένα διαφορετικό σημείο ισορροπίας από αυτό που χρησιμοποίησα για να υπολογίσω το γραμμικό σύστημα ώστε να συγκλίνει πιο γρήγορα στο 0.

Υλοποίηση στο Matlab

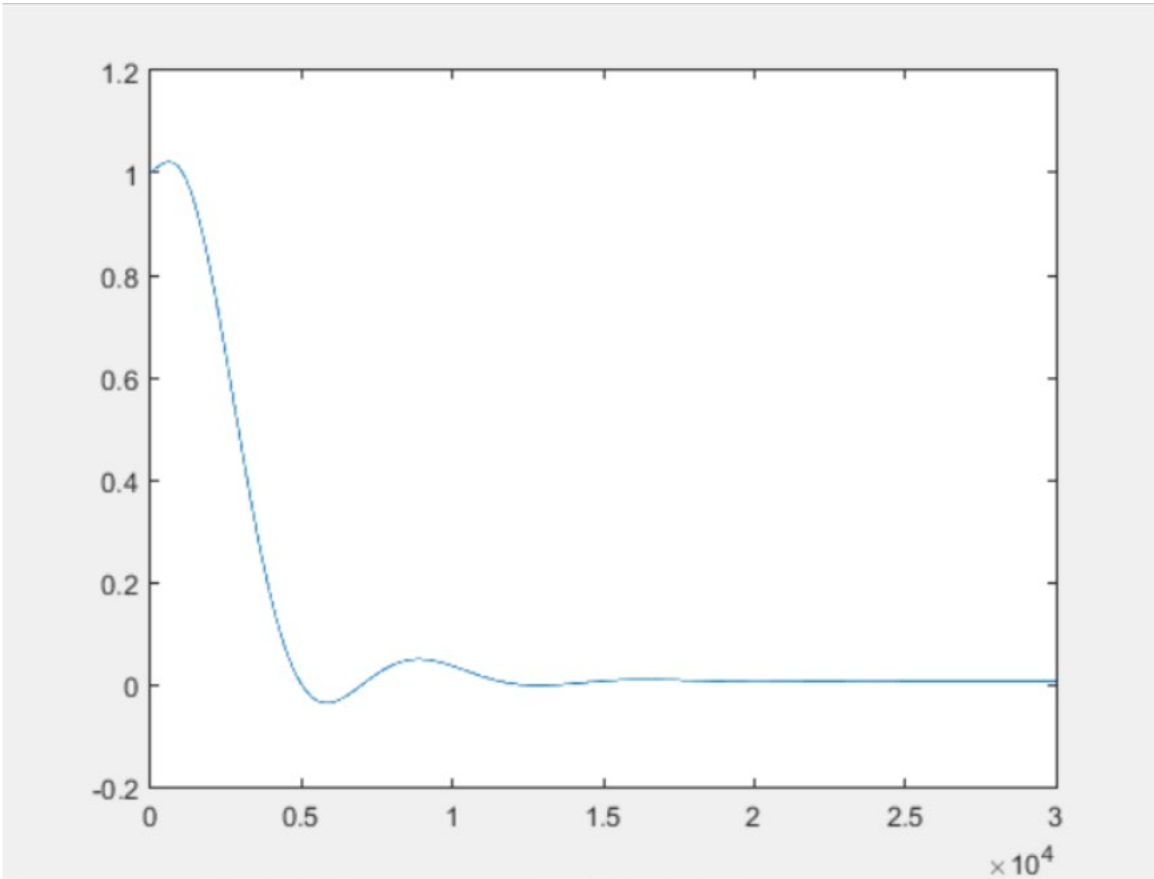
```
close all;
clear;
clc;
format long;

n = 50000;
%οριο για να κάνει break το loop της επανάληψης
tol = 10000000;

%%%%%%%%%%%%%%-------ΑΡΧΙΚΟΠΟΙΗΣΕΙΣ ΠΑΡΑΜΕΤΡΩΝ ΕΛΕΓΚΤΗ-------
%%%%%%%%%%%%%%

%αρχικοποιώ τα K
K1_initial = 0;
K2_initial = 0;
K3_initial = 0;
K4_initial = 0;
K5_initial = 0;
```

```
%αρχικοποιώ τα K_best
K1_best = K1_initial;
K2_best = K2_initial;
K3_best = K3_initial;
K4_best = K4_initial;
K5_best = K5_initial;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%

%ο πίνακας performance κρατάει τα performance κάθε επανάληψης και
το
%best_perf το καλύτερο performance

performance = 0;
best_perf = 0;

for k = 1:n

K_1(1) = K1_initial;
K_2(1) = K2_initial;
K_3(1) = K3_initial;
K_4(1) = K4_initial;
K_5(1) = K5_initial;

%βήμα της προσομοίωσης
dt = 0.001;

%πίνακας σημείου ισορροπίας
x = [1 0.05 0.01 0.17 0.08];

%ολοκλήρωμα του performance που χρησιμεύει ως συνάρτηση
βελτιστοποίησης του
%ελεγκτή
perf_int = 0;


 for i = 1:30000
    y = x(1);

    u = -(K_1(k)*x(1) + K_2(k)*x(2) + K_3(k)*x(3) + K_4(k)*x(4) +
K_5(k)*x(5));

    x1_dot = x(2);
    x2_dot = x(3);
    x3_dot = x(4);
    x4_dot = x(5);
    x5_dot = 18*x(1)*x(3)*x(5) +18*log(abs(x(1)+2)) +
7*x(1)*x(3)*cos(x(4)) +7*x(1)*x(2)*log(abs(x(1)+2))
+18*x(2)*x(3)*x(4) +18*x(1)*cos(x(2))
+19*cos(x(1))*log(abs(x(1)+2)) +19*x(4)*x(2)*x(3)
+9*x(5)*x(2)*cos(x(3)) +9*x(4)*x(1)*log(abs(x(1)+2)) +((
```

```
18*x(4)*x(3)*cos(x(2))
+18*x(1)*x(2)*log(abs(x(1)+2))+15*x(4)*x(5)*log(abs(x(1)+2))+15*x
(2)*x(4)*cos(x(2))+10*x(1)*x(5)*x(3)+10*x(4)*log(abs(x(1)+2))+9*c
os(x(5))*log(abs(x(1)+2))+9*x(1)*x(2)*x(3)+13*x(2)*x(4)*x(5)+13*x
(3)*cos(x(2)))^2)*u;

    x(1) = x(1)+dt*x1_dot;
    x(2) = x(2)+dt*x2_dot;
    x(3) = x(3)+dt*x3_dot;
    x(4) = x(4)+dt*x4_dot;
    x(5) = x(5)+dt*x5_dot;


    perf_int = abs(y) + perf_int;

    if(perf_int > tol)
        break;
    end

 end

     performance(k) = perf_int;

    %στην πρώτη επανάληψη αρχικοποιώ ως best performance
    %το πρώτο performance
    if(k == 1)
        best_perf = performance(k);
    end

%έλεγχος αν το performance μειώνεται ή όχι
if(performance(k) < best_perf)

    best_perf = performance(k);

    K1_best = K_1(k);
    K2_best = K_2(k);
    K3_best = K_3(k);
    K4_best = K_4(k);
    K5_best = K_5(k);


    iteration_of_best_perf = k;
end

    best_perf_plot(k) = best_perf;

   %ορίζω τυχαίες μεταβολές για το K
   disp_K1 = 100 - 200*rand;
   disp_K2 = 100 - 200*rand;
   disp_K3 = 100 - 200*rand;
   disp_K4 = 100 - 200*rand;
   disp_K5 = 100 - 200*rand;
```

```
    K_1(k+1) = K1_best + disp_K1;
    K_2(k+1) = K2_best + disp_K2;
    K_3(k+1) = K3_best + disp_K3;
    K_4(k+1) = K4_best + disp_K4;
    K_5(k+1) = K5_best + disp_K5;
end

%απεικονίζω το best performance και τις τιμές του K ελεγκτή
best_perf
iteration_of_best_perf
K1_best
K2_best
K3_best
K4_best
K5_best

%%%%%%%%%%%%%%%------------ΑΠΕΙΚΟΝΙΖΩ ΤΗΝ ΕΞΟΔΟ Υ ΓΙΑ ΤΟΝ ΚΑΛΥΤΕΡΟ
ΕΛΕΓΚΤΗ------------%%%%%%%%%%%%%%%

%πίνακας σημείου ισορροπίας
 x = [1 0.05 0.01 0.17 0.08];
%βήμα της προσομοίωσης
 dt = 0.001;

  for i = 1:30000
    y = x(1);

    u = -(K1_best*x(1) + K2_best*x(2) + K3_best*x(3) + K4_best*x(4) +
K5_best*x(5));

    x1_dot = x(2);
    x2_dot = x(3);
    x3_dot = x(4);
    x4_dot = x(5);
    x5_dot = 18*x(1)*x(3)*x(5) +18*log(abs(x(1)+2)) +
7*x(1)*x(3)*cos(x(4)) +7*x(1)*x(2)*log(abs(x(1)+2)) +18*x(2)*x(3)*x(4)
+18*x(1)*cos(x(2)) +19*cos(x(1))*log(abs(x(1)+2)) +19*x(4)*x(2)*x(3)
+9*x(5)*x(2)*cos(x(3)) +9*x(4)*x(1)*log(abs(x(1)+2)) +((
18*x(4)*x(3)*cos(x(2))
+18*x(1)*x(2)*log(abs(x(1)+2))+15*x(4)*x(5)*log(abs(x(1)+2))+15*x(2)*x(
4)*cos(x(2))+10*x(1)*x(5)*x(3)+10*x(4)*log(abs(x(1)+2))+9*cos(x(5))*log
(abs(x(1)+2))+9*x(1)*x(2)*x(3)+13*x(2)*x(4)*x(5)+13*x(3)*cos(x(2)))^2)*
u;

    x(1) = x(1)+dt*x1_dot;
    x(2) = x(2)+dt*x2_dot;
    x(3) = x(3)+dt*x3_dot;
    x(4) = x(4)+dt*x4_dot;
    x(5) = x(5)+dt*x5_dot;

    y_plot(i) = real(y);
  end
plot(y_plot)
```

<u>Αποτελέσματα</u>



Παρατηρώ πως πράγματι ο ελεγκτής σταθεροποιεί την έξοδο του μη γραμμικού συστήματος πολύ κοντά στο 0 (0.0062 για την ακρίβεια).

```
best_perf =

     3.431460742143077e+03


iteration_of_best_perf =

       35243
```

<u>Κ ελεγκτής</u>

```
K1_best =

  54.921759544737291


K2_best =

     1.474137899736749e+02


K3_best =

     1.518556939572198e+02


K4_best =

     1.297376106971986e+02


K5_best =

   8.103853420806223
```

4)

Σχεδίαση ελεγκτή $u = Kx + \frac{\theta_1 f_1(x)+ \theta_2 f_2(x)+\cdots+\theta_5 f_5(x)}{(u_1 g_1(x)+u_2 g_2(x)+\cdots+u_5 g_5(x))^2}$ και προσομοίωση με το μη γραμμικό σύστημα

Σε αυτό το ερώτημα θα ακολουθήσω παρόμοια διαδικασία με το προηγούμενο ερώτημα,μόνο που θα πρέπει να χρησιμοποιήσω περισσότερους παραμέτρους,κάτι που αναμένω να οδηγήσει την έξοδό μου στο 0 σε λιγότερες επαναλήψεις.

<u>Υπερπαράμετροι</u>

Αρχικές τιμές ελεγκτή $K = [\,0 \quad 0 \quad 0 \quad 0 \quad 0]$

Εύρος τυχαίων μεταβολών K [-1,1]

Αρχικές τιμές παραμέτρων $\theta = [-1 \quad -1 \quad -1 \quad -1 \quad -1]$

Εύρος τυχαίων μεταβολών θ [-1,1]

Αρχικές τιμές παραμέτρων $u = [1 \quad 1 \quad 1 \quad 1 \quad 1]$

Εύρος τυχαίων μεταβολών u [-1,1]

Επαναλήψεις *15,000*

Ανώτατο όριο συνάρτησης κόστους 10^7

Βήμα προσομοίωσης $dt = 10^{-2}$

Βήματα *20,000*

Κώδικας Matlab

```
close all;
clear;
clc;

format long;
n = 15000;
%οριο για να κάνει break το loop της επανάληψης
tol = 1000000;

%%%%%%%%%%%%%%-------ΑΡΧΙΚΟΠΟΙΗΣΕΙΣ ΠΑΡΑΜΕΤΡΩΝ ΕΛΕΓΚΤΗ-------
%%%%%%%%%%%%%%

%αρχικοποιώ τα K
K1_initial = 0;
K2_initial = 0;
K3_initial = 0;
K4_initial = 0;
K5_initial = 0;

%αρχικοποιώ τα theta
theta1_init = -1;
theta2_init = -1;
theta3_init = -1;
theta4_init = -1;
theta5_init = -1;

%αρχικοποιώ τα u
u1_init = 1;
u2_init = 1;
u3_init = 1;
u4_init = 1;
u5_init = 1;

%αρχικοποιώ τα K_best
K1_best = K1_initial;
K2_best = K2_initial;
K3_best = K3_initial;
K4_best = K4_initial;
K5_best = K5_initial;

%αρχικοποιώ τα theta_best
theta1_best = theta1_init;
theta2_best = theta2_init;
theta3_best = theta3_init;
theta4_best = theta4_init;
theta5_best = theta5_init;

%αρχικοποιώ u best
u1_best = u1_init;
u2_best = u2_init;
u3_best = u3_init;
u4_best = u4_init;
```

```
u5_best = u5_init;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%ο πίνακας performance κρατάει τα performance κάθε επανάληψης και το
%best_perf το καλύτερο performance
performance = 0;
best_perf = 0;


%loop επαναλήψεων μέχρι να ωθήσω την έξοδο y στο 0 με τον καλύτερο
ελεγκτή
for k = 1:(n-1)

   %αν βρίσκομαι στην πρώτη επανάληψη τότε χρησιμοποιώ τις αρχικές
τιμές που έχω ορίσει
    K_1(1) = K1_initial;
    K_2(1) = K2_initial;
    K_3(1) = K3_initial;
    K_4(1) = K4_initial;
    K_5(1) = K5_initial;

    theta_1(1) = theta1_init;
    theta_2(1) = theta2_init;
    theta_3(1) = theta3_init;
    theta_4(1) = theta4_init;
    theta_5(1) = theta5_init;

    u_1(1) = u1_init;
    u_2(1) = u2_init;
    u_3(1) = u3_init;
    u_4(1) = u4_init;
    u_5(1) = u5_init;


%βήμα της προσομοίωσης
dt = 0.01;

%αρχικοποιώ την μεταβλητή της έξοδου
y = 0;


%πίνακας σημείου ισορροπίας
x = [1 0.05 0.01 0.17 0.08];

%ολοκλήρωμα του performance που χρησιμεύει ως συνάρτηση βελτιστοποίησης
του
%ελεγκτή
perf_int = 0;

 for i = 1:20000
    y = x(1);
```

```
    u = -(K_1(k)*x(1) + K_2(k)*x(2) + K_3(k)*x(3) + K_4(k)*x(4) +
K_5(k)*x(5)) + ( theta_1(k)*x(1)*x(3)*x(5) +
theta_1(k)*log(abs(x(1)+2)) + theta_2(k)*x(1)*x(3)*cos(x(4)) +
theta_2(k)*x(1)*x(2)*log(abs(x(1)+2)) + theta_3(k)*x(2)*x(3)*x(4) +
theta_3(k)*x(1)*cos(x(2)) + theta_4(k)*cos(x(1))*log(abs(x(1)+2)) +
theta_4(k)*x(2)*x(3)*x(4) + theta_5(k)*x(5)*x(2)*cos(x(3)) +
theta_5(k)*x(4)*x(1)*log(abs(x(1)+ 2)))/((u_1(k)*x(4)*x(3)*cos(x(2)) +
u_1(k)*x(1)*x(2)*log(abs(x(1)+2)) + u_2(k)*x(4)*x(5)*log(abs(x(1)+2)) +
u_2(k)*x(2)*x(4)*cos(x(2)) + u_3(k)*x(1)*x(5)*x(3) +
u_3(k)*x(4)*log(abs(x(1)+2)) + u_4(k)*cos(x(5))*log(abs(x(1)+2)) +
u_4(k)*x(2)*x(1)*x(3) + u_5(k)*x(2)*x(4)*x(5) +
u_5(k)*x(3)*cos(x(2)))^2);

    x1_dot = x(2);
    x2_dot = x(3);
    x3_dot = x(4);
    x4_dot = x(5);
    x5_dot = 18*x(1)*x(3)*x(5) +18*log(abs(x(1)+2)) +
7*x(1)*x(3)*cos(x(4)) +7*x(1)*x(2)*log(abs(x(1)+2)) +18*x(2)*x(3)*x(4)
+18*x(1)*cos(x(2)) +19*cos(x(1))*log(abs(x(1)+2)) +19*x(4)*x(2)*x(3)
+9*x(5)*x(2)*cos(x(3)) +9*x(4)*x(1)*log(abs(x(1)+2)) +((
18*x(4)*x(3)*cos(x(2))
+18*x(1)*x(2)*log(abs(x(1)+2))+15*x(4)*x(5)*log(abs(x(1)+2))+15*x(2)*x(
4)*cos(x(2))+10*x(1)*x(5)*x(3)+10*x(4)*log(abs(x(1)+2))+9*cos(x(5))*log
(abs(x(1)+2))+9*x(1)*x(2)*x(3)+13*x(2)*x(4)*x(5)+13*x(3)*cos(x(2)))^2)*
u;

    x(1) = x(1)+dt*x1_dot;
    x(2) = x(2)+dt*x2_dot;
    x(3) = x(3)+dt*x3_dot;
    x(4) = x(4)+dt*x4_dot;
    x(5) = x(5)+dt*x5_dot;


    perf_int = abs(y) + perf_int;
     %perf_int = abs(y)*dt;

    if(perf_int > tol)
        break;
    end

 end
    performance(k) = perf_int;

    %στην 1η επανάληψη αρχικοποιώ ως το best performance
    %το 1ο performance
    if(k == 1)
        best_perf = performance(k);
    end


%elegxos an to performance meiwnetai i oxi
if(performance(k) < best_perf)

    best_perf = performance(k);
```

```
    K1_best = K_1(k);
    K2_best = K_2(k);
    K3_best = K_3(k);
    K4_best = K_4(k);
    K5_best = K_5(k);


    theta1_best = theta_1(k);
    theta2_best = theta_2(k);
    theta3_best = theta_3(k);
    theta4_best = theta_4(k);
    theta5_best = theta_5(k);

    u1_best = u_1(k);
    u2_best = u_2(k);
    u3_best = u_3(k);
    u4_best = u_4(k);
    u5_best = u_5(k);

    iteration_of_best_perf = k;
end

   plot_best_perf(k) = best_perf;

   %ορίζω τυχαίες μεταβολές για το K
   disp_K1 = 0.1 - 0.2*rand;
   disp_K2 = 0.1 - 0.2*rand;
   disp_K3 = 0.1 - 0.2*rand;
   disp_K4 = 0.1 - 0.2*rand;
   disp_K5 = 0.1 - 0.2*rand;


   %ορίζω τυχαίες μεταβολές για το theta
   disp_theta_1 = 0.1 - 0.2*rand;
   disp_theta_2 = 0.1 - 0.2*rand;
   disp_theta_3 = 0.1 - 0.2*rand;
   disp_theta_4 = 0.1 - 0.2*rand;
   disp_theta_5 = 0.1 - 0.2*rand;

   %ορίζω τυχαίες μεταβολές για το u
   disp_u1  = 0.1 - 0.2*rand;
   disp_u2  = 0.1 - 0.2*rand;
   disp_u3  = 0.1 - 0.2*rand;
   disp_u4  = 0.1 - 0.2*rand;
   disp_u5  = 0.1 - 0.2*rand;

   %υπολογίζω τα K της επόμενης επανάληψης
   K_1(k+1) = K1_best + disp_K1;
   K_2(k+1) = K2_best + disp_K2;
   K_3(k+1) = K3_best + disp_K3;
   K_4(k+1) = K4_best + disp_K4;
   K_5(k+1) = K5_best + disp_K5;
```

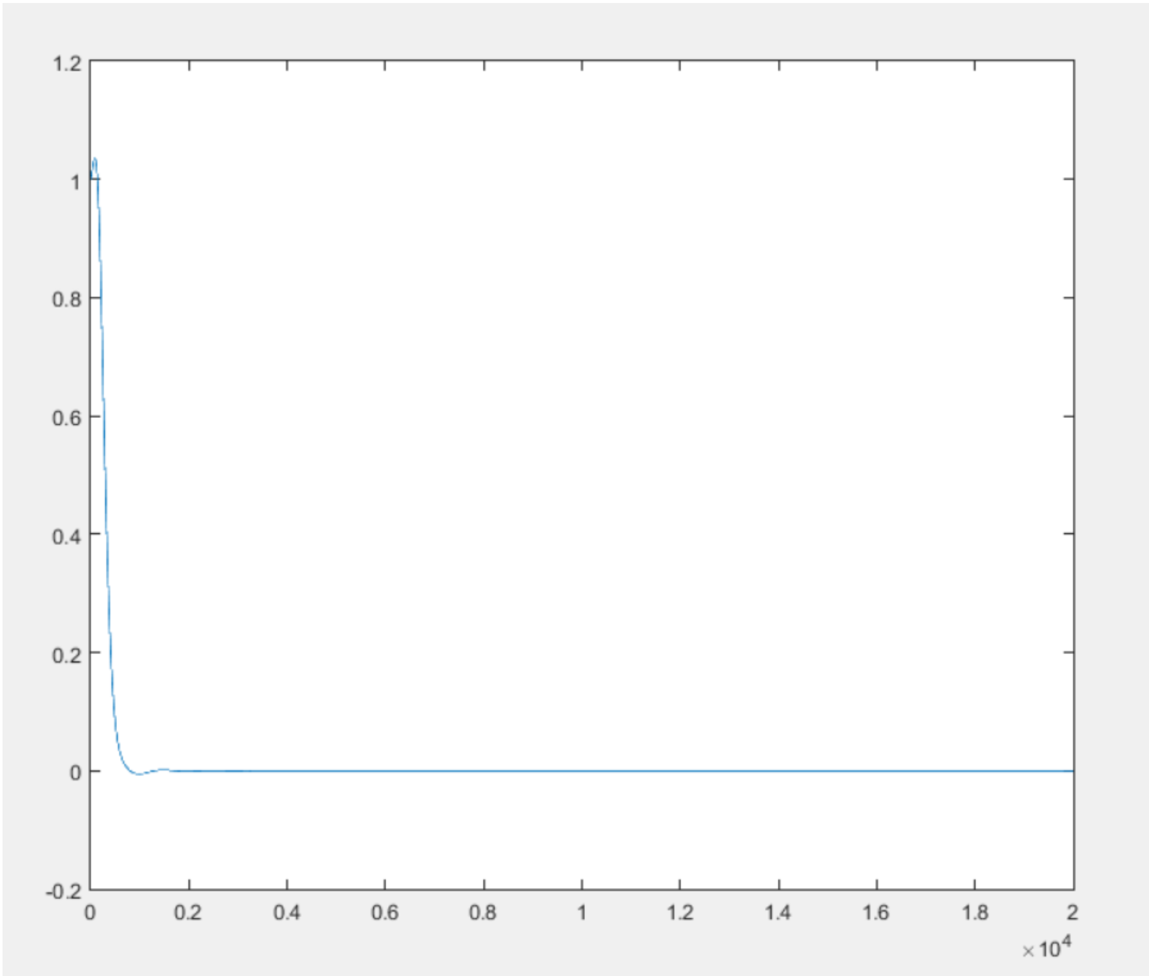
```
    %υπολογίζω τα θ της επόμενης επανάληψης
    theta_1(k+1) = theta1_best + disp_theta_1;
    theta_2(k+1) = theta2_best + disp_theta_2;
    theta_3(k+1) = theta3_best + disp_theta_3;
    theta_4(k+1) = theta4_best + disp_theta_4;
    theta_5(k+1) = theta5_best + disp_theta_5;

    %υπολογίζω τα u της επόμενης επανάληψης
    u_1(k+1) = u1_best + disp_u1;
    u_2(k+1) = u2_best + disp_u2;
    u_3(k+1) = u3_best + disp_u3;
    u_4(k+1) = u4_best + disp_u4;
    u_5(k+1) = u5_best + disp_u5;

end

%%%%%%%%%%%%%%%------------ΑΠΕΙΚΟΝΙΖΩ ΤΗΝ ΕΞΟΔΟ Υ ΓΙΑ ΤΟΝ ΚΑΛΥΤΕΡΟ
ΕΛΕΓΚΤΗ------------%%%%%%%%%%%%%%%

%πίνακας σημείου ισορροπίας
 x = [1 0.05 0.01 0.17 0.08];
%βήμα της προσομοίωσης
 dt = 0.01;

 for i = 1:20000
    y = x(1);

    u = -(K1_best*x(1) + K2_best*x(2) + K3_best*x(3) + K4_best*x(4) +
K5_best*x(5)) + ( theta1_best*x(1)*x(3)*x(5) +
theta1_best*log(abs(x(1)+2)) + theta2_best*x(1)*x(3)*cos(x(4)) +
theta2_best*x(1)*x(2)*log(abs(x(1)+2)) + theta3_best*x(2)*x(3)*x(4) +
theta3_best*x(1)*cos(x(2)) + theta4_best*cos(x(1))*log(abs(x(1)+2)) +
theta4_best*x(2)*x(3)*x(4) + theta5_best*x(5)*x(2)*cos(x(3)) +
theta5_best*x(4)*x(1)*log(abs(x(1)+ 2)))/((u1_best*x(4)*x(3)*cos(x(2))
+ u1_best*x(1)*x(2)*log(abs(x(1)+2)) +
u2_best*x(4)*x(5)*log(abs(x(1)+2)) + u2_best*x(2)*x(4)*cos(x(2)) +
u3_best*x(1)*x(5)*x(3) + u3_best*x(4)*log(abs(x(1)+2)) +
u4_best*cos(x(5))*log(abs(x(1)+2)) + u4_best*x(2)*x(1)*x(3) +
u5_best*x(2)*x(4)*x(5) + u5_best*x(3)*cos(x(2)))^2);

    x1_dot = x(2);
    x2_dot = x(3);
    x3_dot = x(4);
    x4_dot = x(5);
    x5_dot = 18*x(1)*x(3)*x(5) +18*log(abs(x(1)+2)) +
7*x(1)*x(3)*cos(x(4)) +7*x(1)*x(2)*log(abs(x(1)+2)) +18*x(2)*x(3)*x(4)
+18*x(1)*cos(x(2)) +19*cos(x(1))*log(abs(x(1)+2)) +19*x(4)*x(2)*x(3)
+9*x(5)*x(2)*cos(x(3)) +9*x(4)*x(1)*log(abs(x(1)+2)) +((
18*x(4)*x(3)*cos(x(2))
+18*x(1)*x(2)*log(abs(x(1)+2))+15*x(4)*x(5)*log(abs(x(1)+2))+15*x(2)*x(
4)*cos(x(2))+10*x(1)*x(5)*x(3)+10*x(4)*log(abs(x(1)+2))+9*cos(x(5))*log
(abs(x(1)+2))+9*x(1)*x(2)*x(3)+13*x(2)*x(4)*x(5)+13*x(3)*cos(x(2)))^2)*
u;
```

```
    x(1) = x(1)+dt*x1_dot;
    x(2) = x(2)+dt*x2_dot;
    x(3) = x(3)+dt*x3_dot;
    x(4) = x(4)+dt*x4_dot;
    x(5) = x(5)+dt*x5_dot;

    y_plot(i) = real(y);
 end

%απεικονίζω τις τιμές του performance και του βέλτιστου ελεγκτή
best_perf
iteration_of_best_perf
K1_best
K2_best
K3_best
K4_best
K5_best
theta1_best
theta2_best
theta3_best
theta4_best
theta5_best
u1_best
u2_best
u3_best
u4_best
u5_best


plot(y_plot)
```

Τιμή της συνάρτησης κόστους

```
best_perf =

   3.31115158246871e+02
```

Αποτελέσματα



Ο ελεγκτής καταφέρνει να φέρει την έξοδο στο 0 σε λιγότερες επαναλήψεις από τον προηγούμενο ελεγκτή όπως ανέμενα.

<u>Τιμές του ελεγκτή</u>

```
K1_best =

   2.441245944500263


K2_best =

   4.027616015117046


K3_best =

   4.126290402100705


K4_best =

   2.123748353130688


K5_best =

   1.859266120001981
```

```
theta1_best =

  -4.929760721131498


theta2_best =

  -0.392796384380468


theta3_best =

   1.778723619149703


theta4_best =

   0.531948988007702


theta5_best =

  -2.497041913230558
```

```
u1_best =

  -0.743533589893267


u2_best =

   1.957337352409651


u3_best =

   5.133256170611470


u4_best =

  -3.102383162661379


u5_best =

   0.417033137694414
```

<u>Βιβλιογραφικές αναφορές</u>

1. *Βέλτιστος Έλεγχος, Φίλτρο Kalman, Στοχαστικός Έλεγχος*
2. *Διαφάνειες Μαθήματος Συστήματα Αυτόματου Ελέγχου (TMA 563) , https://eclass.duth.gr/courses/TMA563/ , καθ. Ηλίας Κοσματόπουλος*
3. *Διαφάνειες Μαθήματος Σύγχρονος Αυτόματος Έλεγχος (TMA 226) , https://eclass.duth.gr/courses/TMA226/ , καθ. Ηλίας Κοσματόπουλος και καθ. Ιωάννης Μπούταλης*