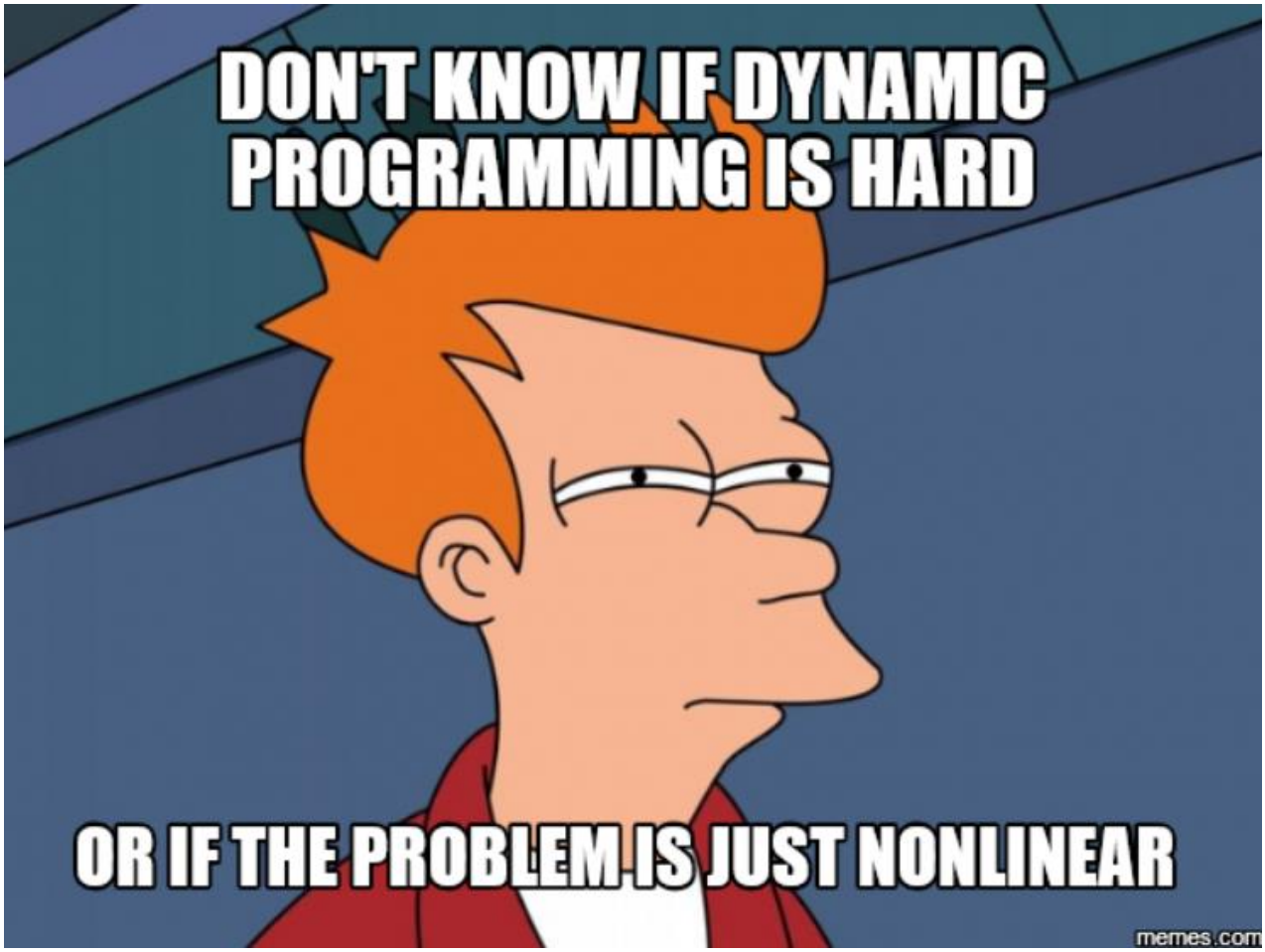
Ειδικά θέματα ΣΑΕ 2020

Τελική Μορφή Εργασίας Εξαμήνου

Σαββας Λιαπης 57403



Υπεύθυνος καθηγητής : Κοσματόπουλος Ηλίας

Κάθε φοιτητής/φοιτήτρια θα πρέπει να σχεδιάσει ελεγκτές για το παρακάτω σύστημα:

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = x_3$$
$$\dots$$
$$\dot{x}_5 = a_1 f_1(x) + a_2 f_2(x) + \cdots + a_n f_n + (b_1 g_1(x) + b_2 g_2(x) + \cdots + b_m g_m(x))^2 u$$
$$y = x_1$$

Από τα ζητούμενα της άσκησης προκύπτουν τα παρακάτω:

α_6	α_5	α_4	α_3	α_2	α_1
Λ	Ι	Α	Π	Η	Σ
11	9	1	16	7	18

b_6	b_5	b_4	b_3	b_2	b_1
Σ	Α	Β	Β	Α	Σ
18	1	2	2	1	18

Με τυχαίο τρόπο φτιάχνουμε τις συναρτήσεις $f_1 \dots f_6$ *και* $g_1 \dots g_6$:

$f_1 = x_5 \cos(x_2) \log(x_5) + x_3 \cos(x_3) \cos(x_5)$	$g_1 = x_2 \cos(x_2) \cos(x_3) + x_3 \cos(x_1) \log(x_1)$
$f_2 = x_3 \log(x_4) \log(x_5)$	$g_2 = x_5^2 \cos(x_1)$
$f_3 = x_2 x_4 \cos(x_2) + x_1^2 \log(x_4)$	$g_3 = \cos(x_5) \log(x_1) \log(x_4)$
$f_4 = x_4 \cos(x_1) \cos(x_4)$	$g_4 = x_1 x_4 \log(x_3) + x_2 x_5 \log(x_2)$
$f_5 = x_4 \cos(x_5) \log(x_1)$	$g_5 = \cos(x_3) \cos(x_4) \log(x_5)$
$f_6 = x_2 \cos^2(x_2)$	$g_6 = x_3 \cos(x_3) \cos(x_5) + \cos^2(x_1) \log(x_2)$

Οπότε το τελικό σύστημα που προκύπτει θα είναι :

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = x_3$$
$$\dot{x}_3 = x_4$$
$$\dot{x}_4 = x_5$$

$$\dot{x}_5 = 18[x_5 \cos(x_2) \log(x_5) + x_3 \cos(x_3) \cos(x_5)] + 7[x_3 \log(x_1) \log(x_5)] +$$

$$+16[x_2 x_5 \cos(x_2) + x_1^2 \log(x_3)] + [x_4 \cos(x_1) \cos(x_4)] + 9[x_4 \cos(x_5) \log(x_1)] +$$

$$+11[x_2 \cos^2(x_2)] + \{18[x_2 \cos(x_2) \cos(x_3) + x_3 \cos(x_1) \log(x_1)] + [x_5^2 \cos(x_1)] +$$

$$+2[\cos(x_5) \log(x_1) \log(x_4)] + 2[x_1 x_4 \log(x_3) + x_2 x_5 \log(x_2)] +$$

$$+[\cos(x_3) \cos(x_4) \log(x_5)] + 18[x_3 \cos(x_3) \cos(x_5) + \cos^2(x_1) \log(x_2)] \}^2 u$$
$$y = x_1$$

Ζητούμενο 1

Γραμμικοποίηστε το σύστημα γύρω από ένα σημείο ισορροπίας.

Απάντηση 1

Η γραμμικοποίηση του συστήματος σημαίνει να το φέρουμε σε μίας μορφή του τύπου

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

Όπου A,B,C,D πίνακες των οποίων τα στοιχεία θα βρούμε παρακάτω.

Τώρα για να γραμμικοποιήσουμε το σύστημα γύρω από ένα σημείο ισορροπίας θα χρησιμοποιήσουμε την σειρά Taylor η οποία δίνεται ως :

$$D(x_1, x_2, x_3, x_4, x_5, u) = D_{\Sigma.I.} + \frac{\partial D}{\partial x_1}\Big|_{x=\Sigma.I}(x_1 - x_{1\Sigma.I.}) + \frac{\partial D}{\partial x_1}\Big|_{x=\Sigma.I}(x_2 - x_{2\Sigma.I.}) +$$

$$+ \frac{\partial D}{\partial x_1}\Big|_{x=\Sigma.I}(x_3 - x_{3\Sigma.I.}) + \frac{\partial D}{\partial x_1}\Big|_{x=\Sigma.I}(x_4 - x_{4\Sigma.I.}) + \frac{\partial D}{\partial x_1}\Big|_{x=\Sigma.I}(x_5 - x_{5\Sigma.I.}) +$$

$$+ \frac{\partial D}{\partial u}(u - u_{\Sigma.I.}) \quad (1)$$

Αγνοούμε τις υψηλότερες τάξεις γιατί θέλουμε να βγάλουμε γραμμικό σύστημα.
Έχουμε λοιπόν :

$$D(x_1, x_2, x_3, x_4, x_5, u) =$$

$$= 18[x_5 \cos(x_2) \log(x_5) + x_3 \cos(x_3) \cos(x_5)] +$$

$$+7[x_3 \log(x_1) \log(x_5)] +$$

$$+16[x_2 x_5 \cos(x_2) + x_1^2 \log(x_3)] +$$

$$+[x_4 \cos(x_1) \cos(x_4)] +$$

$$+9[x_4 \cos(x_5) \log(x_1)] +$$

$$+11[x_2 \cos^2(x_2)] +$$

$$+\{18[x_2 \cos(x_2) \cos(x_3) + x_3 \cos(x_1) \log(x_1)]$$

$$+[x_5^2 \cos(x_1)] +$$

$$+2[\cos(x_5) \log(x_1) \log(x_4)] +$$

$$+2[x_1 x_4 \log(x_3) + x_2 x_5 \log(x_2)] +$$

$$+[\cos(x_3) \cos(x_4) \log(x_5)] +$$

$$+18[x_3 \cos(x_3) \cos(x_5) + \cos^2(x_1) \log(x_2)]\}^2 u$$

Βρίσκω τις μερικές παραγώγους του D ως προς όλες τις μεταβλητές :

$$\frac{\partial D}{\partial x_1} = 7\left[x_3 \frac{1}{x_1}\log(x_5)\right] + 16[2x_1\log(x_3)] + [-x_4\sin(x_1)\cos(x_4)] + 9\left[x_4\cos(x_5)\frac{1}{x_1}\right]$$

$$2\{18[x_2\cos(x_2)\cos(x_3) + x_3\cos(x_1)\log(x_1)] + [x_5^2\cos(x_1)] +$$

$$+2[\cos(x_5)\log(x_1)\log(x_4)] + 2[x_1x_4\log(x_3) + x_2x_5\log(x_2)] +$$

$$+[\cos(x_3)\cos(x_4)\log(x_5)] + 18[x_3\cos(x_3)\cos(x_5) + \cos^2(x_1)\log(x_2)]\}u$$

$$* \{18\left[-x_3\sin(x_1)\log(x_1) + x_3\cos(x_1)\frac{1}{x_1}\right] + [-x_5^2\sin(x_1)] + 2[x_4\log(x_3)] +$$

$$+2\left[\cos(x_5)\log(x_4)\frac{1}{x_1}\right] + 18[-2\cos(x_1)\sin(x_1)\log(x_2)]\}^2u$$

$$\frac{\partial D}{\partial x_2} = 18[-x_5\sin(x_2)\log(x_5)] + 16[x_4\cos(x_2) - x_2x_4\sin(x_2)] + 11[\cos^2(x_2) -$$

$$-2x_2\cos(x_2)\sin(x_2)] + 2\{18[x_2\cos(x_2)\cos(x_3) + x_3\cos(x_1)\log(x_1)] + [x_5^2\cos(x_1)] +$$

$$+2[\cos(x_5)\log(x_1)\log(x_4)] + 2[x_1x_4\log(x_3) + x_2x_5\log(x_2)] +$$

$$+[\cos(x_3)\cos(x_4)\log(x_5)] + 18[x_3\cos(x_3)\cos(x_5) + \cos^2(x_1)\log(x_2)]\}u +$$

$$* \{18[\cos(x_2)\cos(x_3) - x_2\sin(x_2)\cos(x_3)] + 2[x_5 log x_2 + +x_5] +$$

$$+18\left[\cos^2(x_1)\frac{1}{x_2}\right]\}^2u$$

$$\frac{\partial D}{\partial x_3} = 18[\cos(x_3)\cos(x_5) - x_3\sin(x_{35})\cos(x_5)] + 7[\log(x_1)\log(x_5)] + 16\left[x_1^2\frac{1}{x_3}\right] +$$

$$+2\{18[x_2\cos(x_2)\cos(x_3) + x_3\cos(x_1)\log(x_1)] + [x_5^2\cos(x_1)] +$$

$$+2[\cos(x_5)\log(x_1)\log(x_4)] + 2[x_1x_4\log(x_3) + x_2x_5\log(x_2)] +$$

$$+[\cos(x_3)\cos(x_4)\log(x_5)] + 18[x_3\cos(x_3)\cos(x_5) + \cos^2(x_1)\log(x_2)]\}u +$$

$$* \{18[-x_2\cos(x_2)\sin(x_3) + \cos(x_1)\log(x_1)] + 2\left[x_1x_4\frac{1}{x_3}\right] +$$

$$+[-\sin(x_3)\cos(x_4)\log(x_5)] + 18[-x_3\cos(x_5)\sin(x_3) + \cos(x_3)\cos(x_5)]\}^2u$$

$$\frac{\partial D}{\partial x_4} = 7\left[x_3\log(x_5)\frac{1}{x_4}\right] + 16[x_2\cos(x_2)] + [\cos(x_1)\cos(x_4) - x_4\cos(x_1)\sin(x_4)]$$

$$+9[\cos(x_5)\log(x_1)] + 2\{18[x_2\cos(x_2)\cos(x_3) + x_3\cos(x_1)\log(x_1)] + [x_5^2\cos(x_1)] +$$

$$+2[\cos(x_5)\log(x_1)\log(x_4)] + 2[x_1x_4\log(x_3) + x_2x_5\log(x_2)] +$$

$$+[\cos(x_3)\cos(x_4)\log(x_5)] + 18[x_3\cos(x_3)\cos(x_5) + \cos^2(x_1)\log(x_2)]\}u *$$

$$*\{2\left[\cos(x_5)\log(x_1)\frac{1}{x_4}\right] + 2[x_1\log(x_3)] + [-\cos(x_3)\sin(x_4)\log(x_5)]\}^2 u$$

$$\frac{\partial D}{\partial x_5} = 18[\cos(x_2)\log(x_5) + \cos(x_2)] + 7\left[x_3\log(x_1)\frac{1}{x_5}\right] + 16[x_2\cos(x_2)] +$$

$$+9[-x_4\sin(x_5)\log(x_1)] + [-\cos(x_3)\sin(x_4)\log(x_5)]\}^2 u\{\ [2x_5\cos(x_1)]$$

$$+2[-\sin(x_5)\log(x_1)\log(x_4)] + 2[x_2\log(x_2)] + \left[\cos(x_3)\cos(x_4)\frac{1}{x_5}\right] +$$

$$18[-x_3\cos(x_3)\sin(x_5)]\}^2 u$$

$$\frac{\partial D}{\partial u} = 18[x_2\cos(x_2)\cos(x_3) + x_3\cos(x_1)\log(x_1)] + [x_5^2\cos(x_1)] +$$

$$+2[\cos(x_5)\log(x_1)\log(x_4)] + 2[x_1x_4\log(x_3) + x_2x_5\log(x_2)] +$$

$$+[\cos(x_3)\cos(x_4)\log(x_5)] + 18[x_3\cos(x_3)\cos(x_5) + \cos^2(x_1)\log(x_2)]\}^2$$

Επιλέγουμε το σημείο ισορροπίας τυχαία : Σ.Ι. (8.7, 10, 1.5, 3.3, 4.9, 0.1)

x_1	x_2	x_3	x_4	x_5	u
8.7	10	1.5	3.3	4.9	0.1

$\cos(x_1)$	$\cos(x_2)$	$\cos(x_3)$	$\cos(x_4)$	$\cos(x_5)$
0.7486	-0.8391	0.0707	-0.9875	0.1865

$\sin(x_1)$	$\sin(x_2)$	$\sin(x_3)$	$\sin(x_4)$	$\sin(x_5)$
0.663	-0.544	0.9975	-0.1577	-0.9824

$\log(x_1)$	$\log(x_2)$	$\log(x_3)$	$\log(x_4)$	$\log(x_5)$
0.9395	1	0.1760	0.5185	0.6902

Η τιμή του D στο σημείο ισορροπίας θα είναι :

$D(\ 8.7, 10, 1.5, 3.3, 4.9, 1) = 3.2791 * 10^3$

Οι μερικές παράγωγοι στο σημείο ισορροπίας θα είναι : $*10^4$

$$\left.\frac{\partial D}{\partial u}\right|_{x=\Sigma.I} = 4.0396, \quad \left.\frac{\partial D}{\partial x_1}\right|_{x=\Sigma.I} = -0.0515, \quad \left.\frac{\partial D}{\partial x_2}\right|_{x=\Sigma.I} = 0.2065$$

$$\left.\frac{\partial D}{\partial x_3}\right|_{x=\Sigma.I} = 0.6405, \quad \left.\frac{\partial D}{\partial x_4}\right|_{x=\Sigma.I} = 0.0298, \quad \left.\frac{\partial D}{\partial x_5}\right|_{x=\Sigma.I} = 0.1731$$

από όλα τα παραπάνω καταλήγουμε τελικά :

$$(1) \Leftrightarrow D(x_1, x_2, x_3, x_4, x_5, u) = 3.2791 * 10^3 - 0.0515 * 10^4 \ (x_1 - 8.7) +$$
$$+0.2065 * 10^4 (x_2 - 10) + 0.6405 * 10^4 (x_3 - 1.5) + 0.0298 * 10^4 (x_4 - 3.3) +$$
$$+0.1731 * 10^4 (x_5 - 4.9) + 4.0396 * 10^4 (u - 0.1) \Leftrightarrow$$
$$\Leftrightarrow \boldsymbol{D(x_1, x_2, x_3, x_4, x_5, u) = -515.2043089\, x_1 + 2064.645514 x_2 + 6405.453101 x_3 + 298.1192552 x_4 + 1731.370896 x_5 + 40395.58932 u - -36000.29187}$$

Έχουμε φέρει λοιπόν το σύστημα στην γραμμική μορφή :

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

$$y = Cx + Du$$

Όπου :

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ -515.2043089 & 2064.645514 & 6405.453101 & 298.1192552 & 1731.370896 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 40395.58932 \end{bmatrix}, \quad C = [1 \quad 0 \quad 0 \quad 0 \quad 0], \quad D = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -36000.29187 \end{bmatrix}$$

Ολοκληρωμένα και σε μορφή πινάκων θα έχουμε :

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ -515.2043089 & 2064.645514 & 6405.453101 & 298.1192552 & 1731.370896 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}$$

$$+ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 40395.58932 \end{bmatrix} \upsilon + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -36000.29187 \end{bmatrix}$$

$$y = [1 \quad 0 \quad 0 \quad 0 \quad 0] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}$$

Ο υπολογισμός των παραπάνω έγινε για επαλήθευση και ευκολία με την βοήθεια του παρακάτω κώδικα :

```
clear;
clc;

%eidika sae question 1
%making the equations
syms x1 x2 x3 x4 x5 u

%this array contains the diffrent dimensions
dims=[x1,x2,x3,x4,x5,u];

%equilibrium point
eqpoint=[8.7,10,1.5,3.3,4.9,0.01];

%my non linear system
D= 18*(x5*cos(x2)*log(x5)+x3*cos(x3)*cos(x5))+7*(x3*log(x1)*log(x5))...
    +16*(x2*x5*cos(x2)+x1*log(x3))+(x4*cos(x1)*cos(x4))+9*(x4*cos(x5)...
    *log(x1))+11*(x2*cos(x2))+((18*(x2*cos(x2)*cos(x3)+x3*cos(x1)...
    *log(x1))+((x5^2)*cos(x1))+2*(cos(x5)*log(x1)*log(x4))+2*(x1*x4...
    *log(x3)+x2*x5*log(x2))+(cos(x3)*cos(x4)*log(x5))+18*(x3*cos(x3)...
    *cos(x5)+(cos(x1)^2)*log(x2)))^2)*u;

%the array with the partial derivatives
derD=[diff(D,x1),diff(D,x2),diff(D,x3),diff(D,x4),diff(D,x5),diff(D,u)];

%value of D in the equilibrium point
eqDval=eval(subs(D,dims,eqpoint))
%values of derivatives in the equilibrium point
eqdervals=[eval(subs(derD(1),dims,eqpoint)),...
           eval(subs(derD(2),dims,eqpoint)),...
           eval(subs(derD(3),dims,eqpoint)),...
           eval(subs(derD(4),dims,eqpoint)),...
           eval(subs(derD(5),dims,eqpoint)),...
           eval(subs(derD(6),dims,eqpoint))];

%linearized function
digits(10)
Dlinear=vpa(eqDval+eqdervals(1)*(x1-eqpoint(1))+eqdervals(2)*(x2...
-eqpoint(2))+eqdervals(3)*(x3-eqpoint(3))+eqdervals(4)* ...
(x4-eqpoint(4))+eqdervals(5)*(x5-eqpoint(5))+eqdervals(6) ...
*(u-eqpoint(6)))
```

Ζητούμενο 2 (a)

Σχεδιάστε τα κέρδη ενός PID ελεγκτή κάνοντας χρήση του γραμμικού συστήματος:
Βελτιστοποίησε τα κέρδη του PID ελεγκτή έτσι ώστε οι πόλοι να έχουν επιθυμητές τιμές.

- Υπολόγισε συνάρτηση μεταφοράς του γραμμικού συστήματος.
- Υπολόγισε συνάρτηση μεταφοράς του γραμμικού συστήματος κλειστού βρόχου.

Απάντηση 2(α)

Υπολογισμός συστήματος ανοιχτού βρόγχου

Στο σύστημα ανοιχτού βρόγχου η είσοδος δεν εξαρτάται από την έξοδο. Από την γραμμική μορφή που έχουμε καταλήξει στο πρώτο ερώτημα μπορούμε να βγάλουμε την συνάρτηση μεταφοράς με τον παρακάτω τρόπο. Αν έχουμε ένα σύστημα της μορφής :

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

Όπου :

$$A = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ -a_0 & -a_1 & -a_2 & \dots & -a_{n-1} \end{bmatrix}, \quad \mathrm{B}=\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ B \end{bmatrix}, \quad \mathrm{C}=[b_0 \quad b_1 \quad \dots \quad b_{n-1}]$$

Τότε η συνάρτηση μεταφοράς θα είναι :

$$G(s) = \frac{b_{n-1}s^{n-1} + b_{n-2}s^{n-2} + \cdots + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0}\, B$$

Οπότε στο δικό μας σύστημα η συνάρτηση μεταφοράς θα είναι :

$$G(s) = \frac{40395.58932}{s^5 - 1731.370896s^4 - 298.1192s^3 - 6405.453101s^2 - 2064.645514s + 515.2043089}$$

Υπολογισμός συστήματος κλειστού βρόγχου

Στο σύστημα κλειστού βρόγχου υπάρχει και ένας κλάδος ανάδρασης. Μαζί με αυτό τον κλάδο ανάδρασης τοποθετούμε και έναν PID ελεγκτή. Η συνάρτηση μεταφοράς του PID ελεγκτή είναι :

$$G_c(s) = K_p + K_D s + \frac{K_I}{s}$$

και η συνάρτηση μεταφοράς του γραμμικού συστήματος θα είναι :

$$T(s) = \frac{G(s)G_c(s)}{1 + G(s)G_c(s)}$$

Οπότε η συνάρτηση μεταφοράς του κλειστού βρόγχου του δοθέντος συστήματος γίνεται :

$$T(s) = \frac{40395.58932(K_D s^2 + K_P s + K_I)}{s^6 - 1731.370896s^5 - 298.1192s^4 - 6405.453101s^3 - (2064.645514 + 40395.58932K_D)s^2 + (515.2043089 + 40395.58932Kp)s + 40395.58932K_I}$$

Ευρεση βέλτιστων τιμών K_D K_P K_I

Για την αρχικοποίηση του προβλήματος δημιούργήθηκε μία συνάρτηση η οποία παίρνει ως ορίσματα τον αριθμό γονιδίων που θα περιέχει κάθε χρωμόσωμα και την fitness function . Από αυτή την συνάρτηση εξάγεται ο αρχικός πληθυσμός . Η έξοδος είναι στην ουσία ένα πίνακας που έχει έναν πληθυσμό μεγέθους totalPopulationSize από 3-μελή διανύσματα καθένα από τα οποία περιέχουν μία τυχαία τιμή από -1000 εώς 1000 για K_D μία για K_P και μία για K_I. (έγινε δοκιμή και με μεγαλύτερο και μικρότερο εύρος τιμών αλλά τα αποτελέσματα έδειχναν ότι αυτό το όριο είναι αρκετο).

```
function InitialPopulation = pid_create_permutations(NVARS,FitnessFcn,...
    options)
%the total population will be determined in the options
totalPopulationSize = sum(options.PopulationSize);
%the NVARS will be determinde by the options and is the number of the
%the variables of the problem (genes)
n=NVARS;
%the initial population is a set of as many chromosomes as
%the population size
InitialPopulation = cell(totalPopulationSize,1);
for i = 1:totalPopulationSize
    %random values for Kd Kp Ki between -1000 and 1000
    atempt=-1000+(2000).*rand(1,n);
    InitialPopulation{i} = atempt;
end
```

Για την αξιολόγηση των χρωμοσωμάτων χρησιμοποιήθηκε η fitness function όπου υπολογίζει τους πόλους του συστήματος κλειστού βρόγχου και κρίνει την ποιότητα του. Ένα χρωμόσωμα θα είναι καλύτερης ποιότητας αν ο μέγιστος πόλος του είναι μικρός και όσο μεγαλώνει ο μέγιστος πόλος η ποιότητα του χρωμοσώματος πέφτει. Όσο μεγαλύτερη είναι η τιμή της fitness function για ένα χρωμόσωμα τόσο λιγότερες πιθανότητες έχει να «διαιωνιστέι» :

```
function fitnessValue = pid_fitness(x)
%the fitness Values of each chromosome is contained into the vector fitness
%Value which is empty in the begining.
fitnessValue=zeros(size(x,1),1);
%the loop runs as many times as the population of the chromosomes
for i=1:size(x,1)
    %we check every chromosome of the population
    CurrentChromosome = x{i};
    %I want to check the roots
    r=roots([1 -1731.3709 -298.1192 -6405.4531 ...
            -(2064.6455+ 40395.58932*CurrentChromosome(1)) ...
            (515.2043+40395.58932*CurrentChromosome(2)) ...
            40395.58932*CurrentChromosome(3)]);
    %I am only intrested about the real part
    r=r(imag(r)==0);
    maxroot=max(r);
    %the biggest the maxroot the smaller the value of the chromosome
    fitnessValue(i)=maxroot;
end
```

Ο αλγόριθμος βελτιστοποιείται με την συνάρτηση pid_crossover_permutations. Η συνάρτηση αυτή αρχικοποιεί ένα σύνολο crossoverKids στο οποίο θα αποθηκευτούν τα χρωμοσώματα των νέων παιδιών-μελών του πληθυσμού. Έπειτα επιλέγουμε έναν από τους γονείς. Από το χρωμόσωμα του γονέα αυτού επιλέγουμε 2 γονίδια. Αυτή η διαδικασία επαναλαμβάνεται τόσες φορές όσα είναι και τα παιδιά. Η συνάρτηση επιστρέφει το σύνολο των χρωμοσωμάτων των νέων παιδιών crossoverKids.

```
function crossoverKids  = pid_crossover_permutation(parents,options,...
    NVARS,FitnessFcn,thisScore,thisPopulation)

%one child comes from 2 parents so the nuber of children will be
%half of the selected parents number
nunmberOfKids = length(parents)/2;
%crossoverKids will be set of new chromosomes (the kids).
crossoverKids = cell(nunmberOfKids,1);
index = 1;
for i=1:nunmberOfKids
    %we choose one of every 2 parents
    parent = thisPopulation{parents(index)};
    %the index jumps to the next 2 parents and chooses one
    index = index + 2;
    %we choose 2 genes gene1 and gene2 randomly of the chosen parent
    gene1 = ceil((length(parent) -1) * rand);
    %we should reserve that the gene 2 will be in greater place of
    %gene1
    gene2 = gene1 + ceil((length(parent) - gene1- 1) * rand);
    %we create the new childs choromosome which is identical to the parent
    child = parent;
    %we perform the flip of the two genes and get the new chromosome
    child(gene1:gene2) = fliplr(child(gene1:gene2));
    %the new child gets stored in the vector with all the new children
    crossoverKids{i} = child;
end
```

Η επιλογή των γονέων θα γίνει με τυχαίο τρόπο μέσω μίας συνάρτησης των γενετικών αλγορίθμων (εμείς θα επιλέξουμε συνάρτηση ρουλέτας by default). Κάθε μέλος του πληθυσμού θα έχει την ευκαιρία να γίνει γονέας όμως ανάλογα με την καταλληλόλητα του θα έχει περισσότερες πιθανότητες.

Εκτός από την ομαλή εξαγωγή χρωμοσωμάτων υπάρχουν και μεταλλάξεις. Αυτές πραγματοποιούνται με την συνάρτηση pid_mutate_permutation σε αυτή επιλέγουμε 2 γονίδια και τα αλλάζουμε θέση για κάθε γονέα και παίρνουμε ένα σύνολο μεταλλαγμένων παιδιών.

```
function mutatedChildren = pid_mutate_permutation(parents ,options,...
    NVARS,FitnessFcn, state, thisScore,thisPopulation,mutationRate)
% Here we swap two elements of the permutation
mutatedChildren = cell(length(parents),1);
for i=1:length(parents)
    %we get every parent
    parent = thisPopulation{parents(i)};
    %from the parents chromosome we choose two random genes and swap them
    p = ceil(length(parent) * rand(1,2));
    child = parent;
    child(p(1))= parent(p(2));
    child(p(2))= parent(p(1));
    mutatedChildren{i} = child;
end
```

Ο γεννητικός αλγόριθμος εκτελείται τελικά με τον παρακάτω κώδικα :

```
clear;
clc;
%Kd Kp Ki = 3 variables
numberOfVriables=3;

type pid_create_permutations.m
type pid_crossover_permutation.m
type pid_mutate_permutation.m
type pid_fitness.m
FitnessFcn = @(x) pid_fitness(x);
options = gaoptimset();
options = gaoptimset(  'PopulationType', 'custom',...
                       'PopInitRange',[1;numberOfVriables], ...
                       'CreationFcn',@pid_create_permutations, ...
                       'CrossoverFcn',@pid_crossover_permutation, ...
                       'MutationFcn',@pid_mutate_permutation, ...
                       'Generations',1000,'PopulationSize',1000, ...
                       'StallGenLimit',1000,'Vectorized','on');

[x,fval,reason,output] = ga(FitnessFcn,numberOfVriables,options)

kd=x{1,1}(1)
kp=x{1,1}(2)
ki=x{1,1}(3)
poles=roots([1 -1731.3709 -298.1192 -6405.4531 -(2064.6455+...
            40395.58932*kd) (515.2034+40395.58932*kp) ki])
```

Τρέχοντας τον αλγόριθμο για πληθυσμό 100 και 100 γενιές και έπειτα για πληθυσμό 1000 και 1000 γενιές παίρνουμε τα παρακάτω αποτελέσματα :

```
kd =

 -998.8731


kp =

  637.2565


ki =

 -150.7869


poles =

   1.0e+03 *

   1.7315 + 0.0000i
   0.0288 + 0.0000i
  -0.0142 + 0.0246i
  -0.0142 - 0.0246i
  -0.0006 + 0.0000i
   0.0000 + 0.0000i
```

```
kd =

 -999.5669


kp =

  760.5387


ki =

  136.2264


poles =

   1.0e+03 *

   1.7315 + 0.0000i
   0.0289 + 0.0000i
  -0.0141 + 0.0246i
  -0.0141 - 0.0246i
  -0.0008 + 0.0000i
  -0.0000 + 0.0000i
```

Βλέπουμε ότι είναι αδύνατον αν φέρουμε όλους τους πόλους στο αριστερό ημιεπίπεδο. Η αδυναμία αυτή επιβεβαιώνεται και από το θεώρημα του Vieta το οποίο λέει :

$$\sum_{i=1}^{n} root_i = -\frac{a_{n-1}}{a_n}$$

Όπου:

- $root_i$=ρίζα i του πολυωνύμου
- a_{n-1}, a_n= συντελεστές πολυωνύμου

Πράγματι οι ρίζες που έχουμε βγάλει επαληθεύουν το θεώρημα στην ακρίβεια ενός δεκαδικού.

Ζητούμενο 2(b)

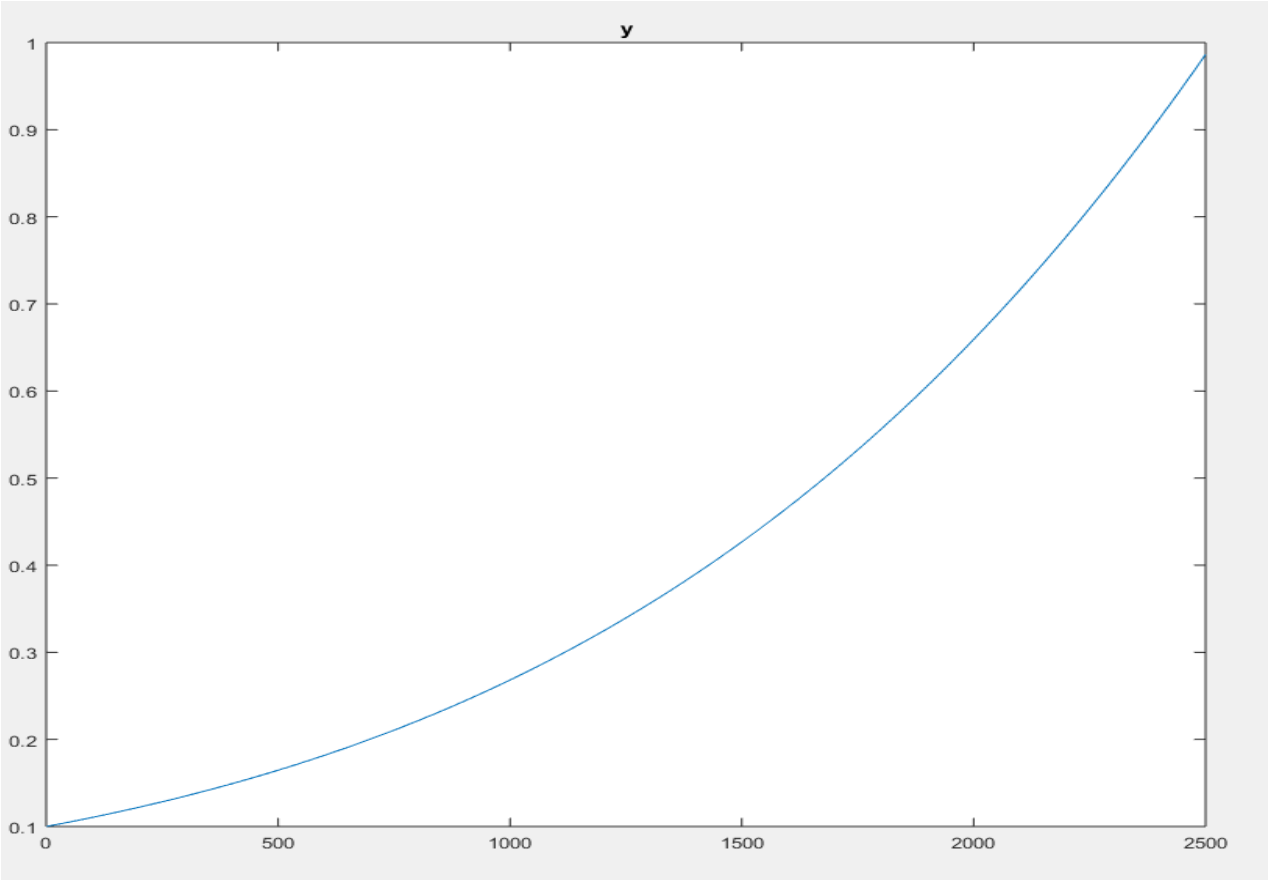
Προσομοιώστε τον ελεγκτή με το «πραγματικό» (μη-γραμμικό) σύστημα.

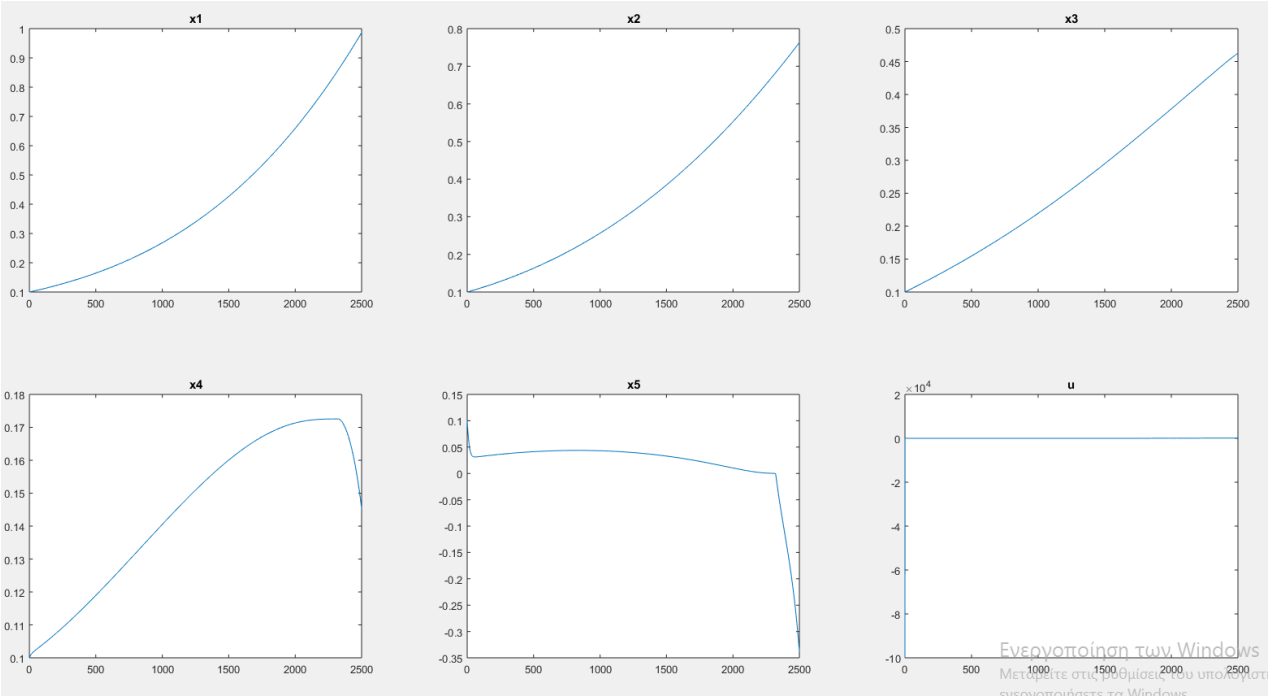
Απάντηση 2(b)

Η προσομοίωση του εκλεκτή πραγματοποιήθηκε με βάση τον παρακάτω κώδικα :

```
clear
clc
%time counter
tic

%initializing the time scope and the time step
dt=0.001;
%itterations of the simulation
simiter=2500;
t=0:simiter;

%giving Kd Kp Ki the values founded in the previous question
kd=-999.5669;
kp=760.5387;
ki=136.2264;

%initializing the x u y arrays
x=[0.1*ones(5,1) zeros(5,simiter)];
u=zeros(1,simiter);
y=[x(1,1) zeros(1,simiter-1)];
xdot=zeros(1,5);
%in order to compute ydot
yprev=0;

for i=1:simiter
    %computing the x1-x5dot
    xdot=[x(2,i) x(3,i) x(4,i) x(5,i) 0];
    xdot(5)=18*(x(5,i)*cos(x(2,i))*log(abs(x(5,i)))+x(3,i)*cos(x(3,i))...
        *cos(x(5,i)))...
        +7*(x(3,i)*log(abs(x(1,i)))*log(abs(x(5,i))))...
        +16*(x(2,i)*x(5,i)*cos(x(2,i))+x(1,i)*log(abs(x(3,i))))...
        +(x(4,i)*cos(x(1,i))*cos(x(4,i)))...
        +9*(x(4,i)*cos(x(5,i))*log(abs(x(1,i))))...
        +11*(x(2,i)*cos(x(2,i)))...
        +(18*(x(2,i)*cos(x(2,i))*cos(x(3,i))+x(3,i)*cos(x(1,i))*log(abs(x(1,i))))...
        +((x(5,i)^2)*cos(x(1,i)))...
        +2*(cos(x(5,i))*log(abs(x(1,i)))*log(abs(x(4,i))))...
        +2*(x(1,i)*x(4,i)*log(abs(x(3,i)))+x(2,i)*x(5,i)*log(abs(x(2,i))))...
        +(cos(x(3,i))*cos(x(4,i))*log(abs(x(5,i))))...
        +18*(x(3,i)*cos(x(3,i))*cos(x(5,i))+(cos(x(1,i))^2)*log(abs(x(2,i)))))^2)*u(i);

    %giving the next values for x1-x5
     for z=1:5
         x(z,i+1)=x(z,i)+xdot(z)*dt;
     end

    %computing the next output
    y(i+1)=x(1,i)+dt*x(1,i);
    ydot=(y(i)-yprev)/dt;
    %computing the control
    u(i)=kp*y(i)+ki*y(i+1)+kd*ydot;
    %the present y will be the previous y for the next loop
    yprev=y(i);
end
%making the plots
figure(1)
for z=1:5
    subplot(2,3,z);plot(t,x(z,:));title(['x',num2str(z)]);
end
subplot(2,3,6);plot(t(1:simiter),u);title('u');
figure(2);plot(t,y);title('y');
```

το αποτέλεσμα αυτής της προσομοίωσης για 2500 επαναλήψεις θα είναι :





Ζητούμενο 2(c)

Επανασχεδιάστε τα κέρδη αν χρειάζεται (μεταβάλετε τυχαία τα κέρδη του PID ελεγκτή και προσομοιώστε το σύστημα κλειστού βρόχου. Τυχαίες μεταβολές παρόμοιες με το βήμα α)

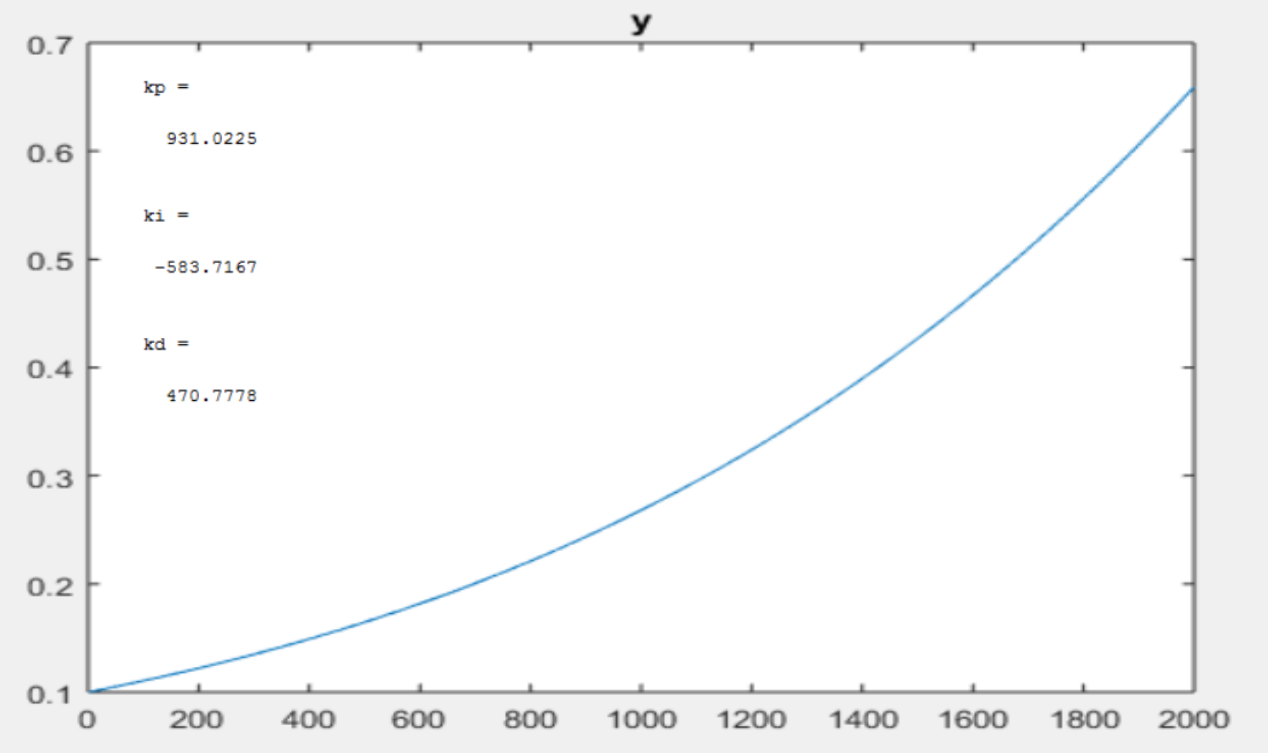
Απάντηση 2(c)

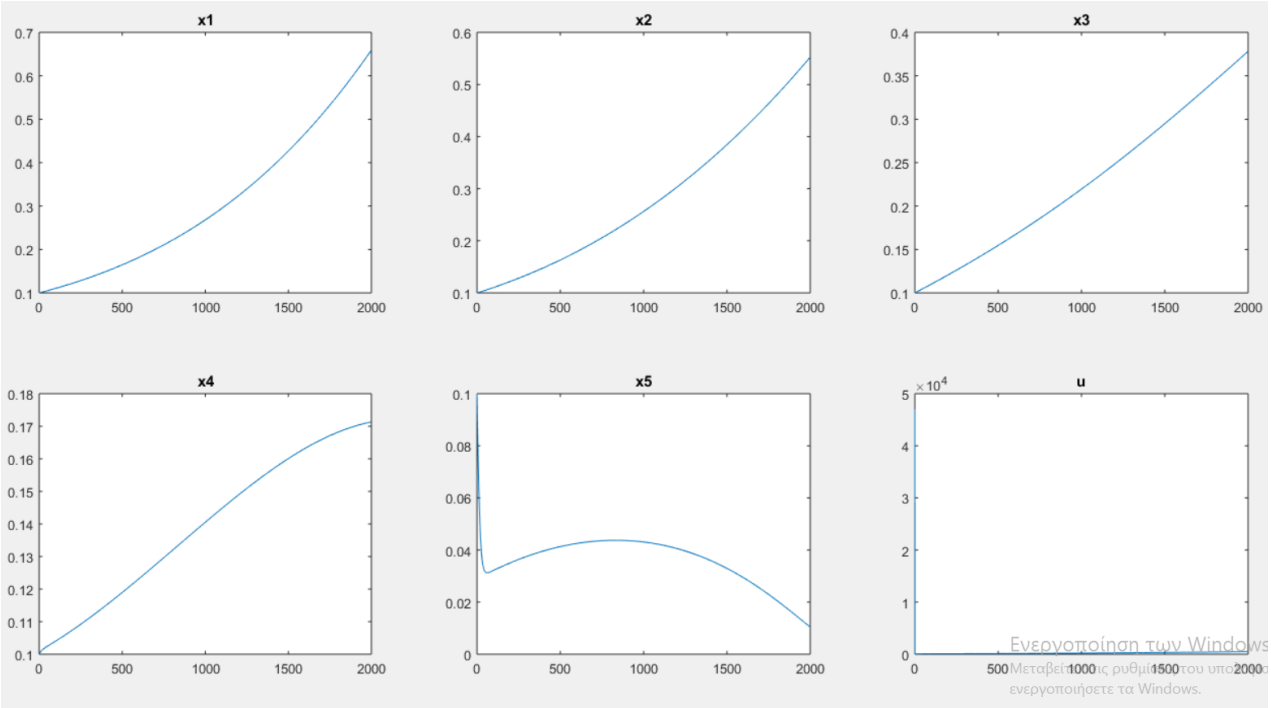
Ο αλγόριθμος που χρησιμοποιήσαμε για αυτό το ερώτημα θα είναι :

```
clear
clc
%time counter
tic

%initializing the time scope and the time step
dt=0.001;
%how many values will I check
iter=10000;
%itterations of the simulation
simiter=2000;
t=0:simiter;

pid=-1000+2000*rand(1,3);

bestperf=10000;
bestpid=pid;
bestx=zeros(5,simiter+1);
bestu=zeros(1,simiter);
besty=zeros(1,simiter);

for j=1:iter
    %initializing the x u y arrays
    x=[0.1*ones(5,1) zeros(5,simiter)];
    u=zeros(1,simiter);
    y=[x(1,1) zeros(1,simiter-1)];
    xdot=zeros(1,5);
    %in order to compute ydot
    yprev=0;
    currentperf=0;

    for i=1:simiter
        if (abs(currentperf)>2e+5)
            break
        end
        %computing the x1-x5dot
        xdot=[x(2,i) x(3,i) x(4,i) x(5,i) 0];
        xdot(5)=18*(x(5,i)*cos(x(2,i))*log(abs(x(5,i)))+x(3,i)*cos(x(3,i))...
            *cos(x(5,i)))...
            +7*(x(3,i)*log(abs(x(1,i)))*log(abs(x(5,i))))...
            +16*(x(2,i)*x(5,i)*cos(x(2,i))+x(1,i)*log(abs(x(3,i))))...
            +(x(4,i)*cos(x(1,i))*cos(x(4,i)))...
            +9*(x(4,i)*cos(x(5,i))*log(abs(x(1,i))))...
            +11*(x(2,i)*cos(x(2,i)))...
            +(18*(x(2,i)*cos(x(2,i))*cos(x(3,i))+x(3,i)*cos(x(1,i))*log(abs(x(1,i))))...
            +((x(5,i)^2)*cos(x(1,i)))...
            +2*(cos(x(5,i))*log(abs(x(1,i)))*log(abs(x(4,i))))...
            +2*(x(1,i)*x(4,i)*log(abs(x(3,i)))+x(2,i)*x(5,i)*log(abs(x(2,i))))...
            +(cos(x(3,i))*cos(x(4,i))*log(abs(x(5,i))))...
            +18*(x(3,i)*cos(x(3,i))*cos(x(5,i))+(cos(x(1,i))^2)*log(abs(x(2,i)))))^2)*u(i);

        %giving the next values for x1-x5
         for z=1:5
             x(z,i+1)=x(z,i)+xdot(z)*dt;
         end
```

```
%computing the next output
        y(i+1)=x(1,i)+dt*x(1,i);
        ydot=(y(i)-yprev)/dt;
        %computing the control
        u(i)=pid(1)*y(i)+pid(2)*y(i+1)+pid(3)*ydot;
        %the present y will be the previous y for the next loop
        yprev=y(i);

        currentperf=currentperf+abs(y(i));
    end
    if (currentperf<bestpid)
        bestpid=pid
        j
        bestx=zeros(5,simiter+1);
        bestu=zeros(1,simiter);
        besty=zeros(1,simiter);
    end
    pid(1)=bestpid(1)-1+2*rand;
    pid(2)=bestpid(2)-1+2*rand;
    pid(3)=bestpid(3)-1+2*rand;
end

kp=bestpid(1)
ki=bestpid(2)
kd=bestpid(3)

%making the plots
figure(1)
for z=1:5
    subplot(2,3,z);plot(t,x(z,:));title(['x',num2str(z)]);
end
subplot(2,3,6);plot(t(1:simiter),u);title('u');
figure(2);plot(t,y);title('y');
```

τα γραφήματα των χ1-χ5 καθώς και ο έλεγχος υ και η έξοδος y :





Αυτά τα αποτελέσματα παίρνουμε για επαναλήψεις προσομοίωσης που είναι γύρω στις 2 χιλιάδες , αν τρέξουμε για πάνω από 3 χιλιάδες επαναλήψεις δεν παίρνουμε κάποιο ουσιαστικό αποτέλεσμα

Ζητούμενο 3

Σχεδιάστε ελεγκτή Γραμμικού-Τετραγωνικού Ελέγχου για το γραμμικοποιημένο σύστημα. Προσομοιώστε τον ελεγκτή με το «πραγματικό» (μη-γραμμικό) σύστημα. Επανασχεδιάστε τον πίνακα κερδών κάνοντας χρήση τυχαίων διαταραχών όπως στο προηγούμενο ερώτημα.

Απάντηση 3

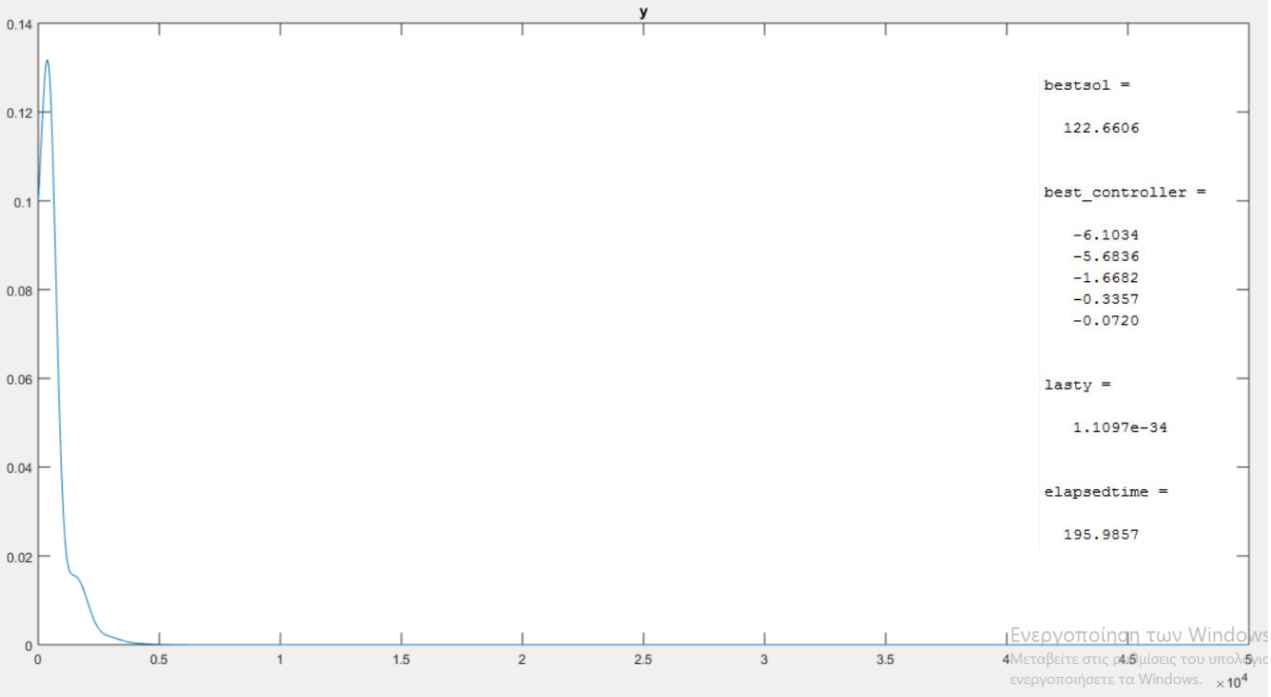
Γραμμικό Σύστημα

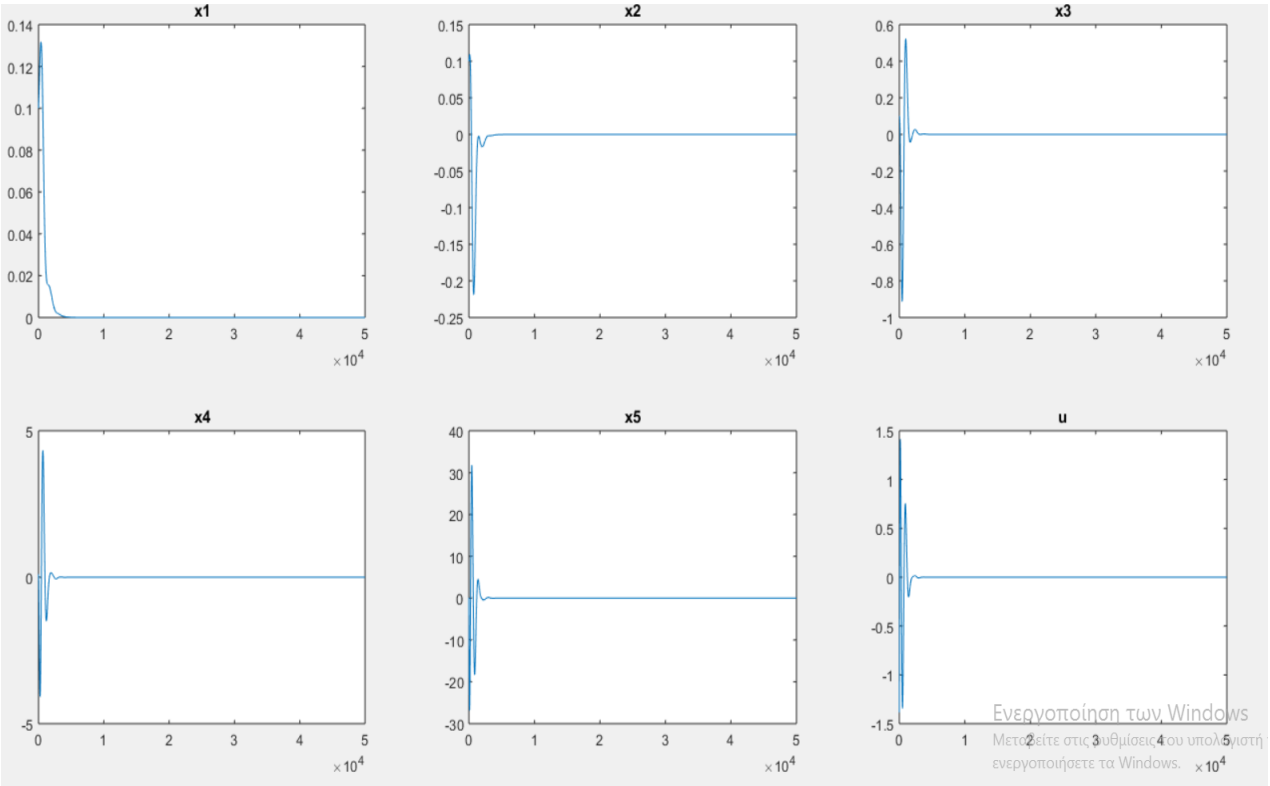
Ο αλγόριθμος που μας δίνει τον ελεγκτή για το γραμμικό σύστημα φαίνεται παρακάτω :

```
clear
clc
%time counter
tic
%initializing the time scope and the time step
dt=0.001;
%how many searches for K
iter=15000;
%itterations of the simulation
simiter=50000;
t=0:simiter;
%k-parameters
k=zeros(1,5);
%performance tracking
bestperf=10000;
bestk=k;
bestx=zeros(5,simiter+1);
bestu=zeros(1,simiter);
besty=zeros(1,simiter);
xdot=zeros(1,5);
counter =0;

for j=1:iter
   counter=counter+1;
    %initializing the x u y arrays
    x=[0.1*ones(5,1) zeros(5,simiter)];
    u=zeros(1,simiter);
    y=[x(1,1) zeros(1,simiter-1)];
    currentperf=0;
    for i=1:simiter
        if (abs(currentperf)>bestperf)
            break
        end
        %computing the control
        u(i)=[k(1) k(2) k(3) k(4) k(5)]*x(:,i);

        %computing the x1-x5dot
        xdot=[x(2,i) x(3,i) x(4,i) x(5,i) 0];

        %this is my linear system
        xdot(5)=1/100*(-515.2043*x(1,i)+2064.6455*x(2,i)+6405.4531*x(3,i)...
                +298.1192*x(4,i)+1731.3709*x(5,i)+40395.5893*u(i));

        %giving the next values for x1-x5
         for z=1:5
             x(z,i+1)=x(z,i)+xdot(z)*dt;
         end
         %computing the next output
         y(i+1)=x(1,i+1);
         currentperf=currentperf + abs(y(i));
    end
```

```
    if (abs(currentperf) < bestperf)
        bestperf=currentperf
        j
        bestk=k;
        bestx=x;
        bestu=u;
        besty=y;
    end
    for z=1:5
        k(z)=bestk(z)+2*rand-1;
    end

    if counter==1000
        j
        counter=0;
    end

end

bestsol=bestperf
best_controller=bestk'

% simulation_plots( performance,x1,x2,x3,x4,x5,u,y,t)
figure(1);
for z=1:5
    subplot(2,3,z);plot(t,bestx(z,:));title(['x',num2str(z)]);
end
subplot(2,3,6);plot(t(1:simiter),bestu);title('u');
figure(2);plot(t,besty);title('y');
elapsedtime=toc
```

Τα αποτελέσματα του αλγορίθμου για το γραμμικό σύστημα θα είναι :





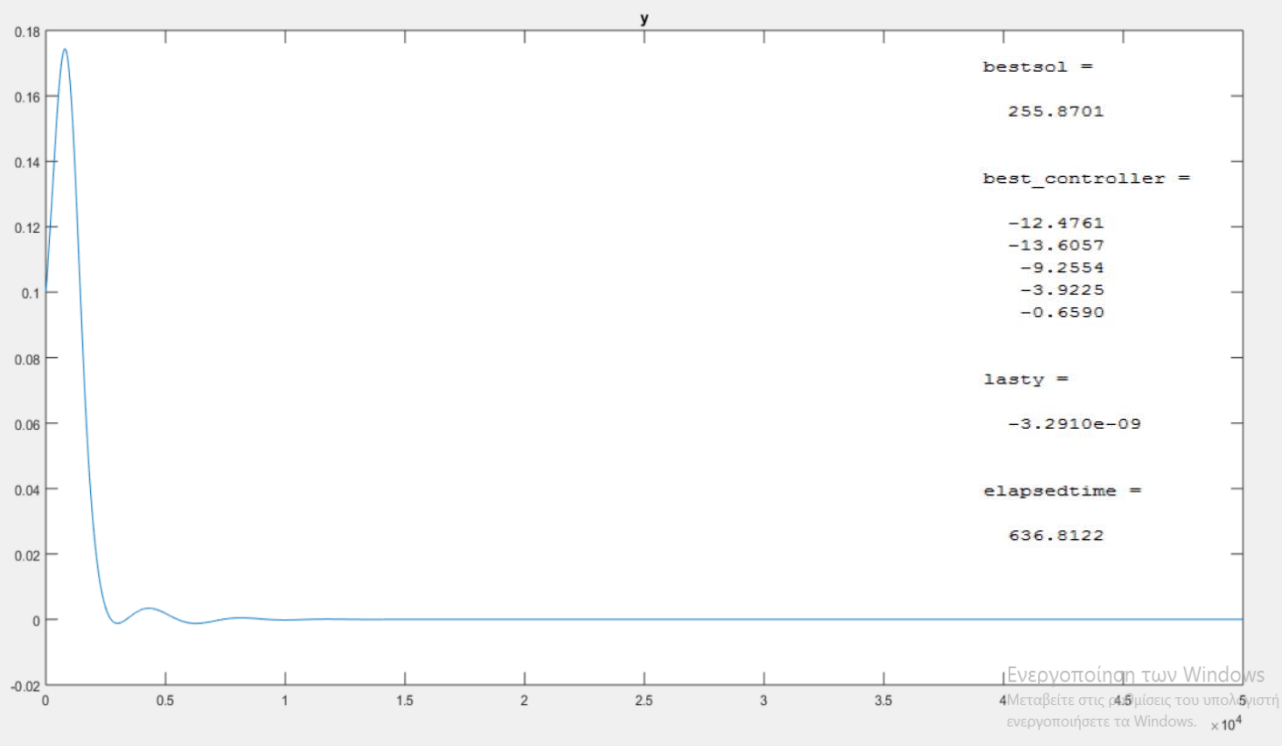
Μη Γραμμικό Σύστημα

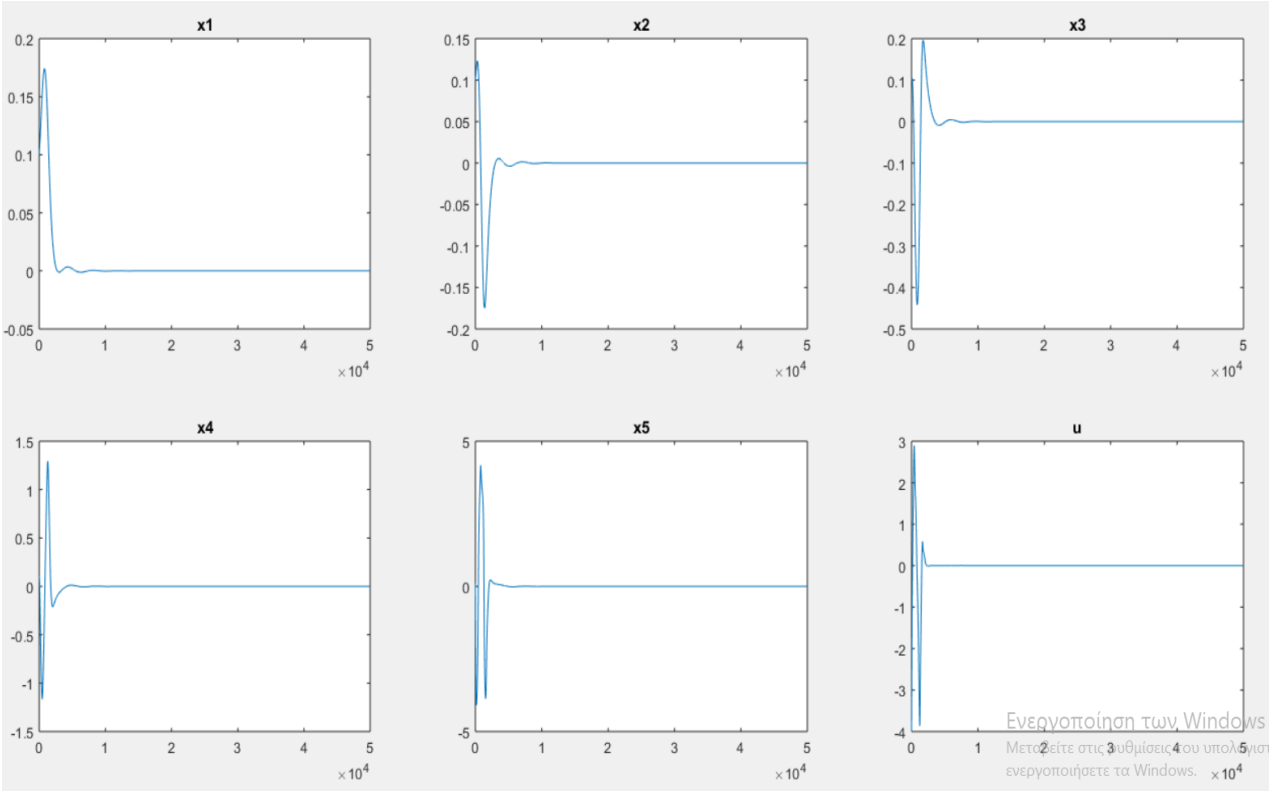
Ο κώδικας που τρέξαμε για το μη γραμμικό σύστημα θα είναι:

```
clear
clc
%time counter
tic
%initializing the time scope and the time step
dt=0.001;
%how many searches for K
iter=15000;
%itterations of the simulation
simiter=50000;
t=0:simiter;

%k-parameters
k=zeros(1,5);
%performance tracking
bestperf=2e+5;
bestk=k;
bestx=zeros(5,simiter+1);
bestu=zeros(1,simiter);
besty=zeros(1,simiter);
xdot=zeros(1,5);

counter=0;

for j=1:iter

    counter=counter+1;
    %initializing the x u y arrays
    x=[0.1*ones(5,1) zeros(5,simiter)];
    u=zeros(1,simiter);
    y=[x(1,1) zeros(1,simiter-1)];
    currentperf=0;

    for i=1:simiter
        if (abs(currentperf)>bestperf)
            break
        end
        %computing the control
        u(i)=[k(1) k(2) k(3) k(4) k(5)]*x(:,i);

        %computing the x1-x5dot
        xdot=[x(2,i) x(3,i) x(4,i) x(5,i) 0];

        %this is my linear system
        xdot(5)=1/1000*(18*(x(5,i)*cos(x(2,i))*log(abs(x(5,i)))+x(3,i)*cos(x(3,i))*cos(x(5,i)))...
            +7*(x(3,i)*log(abs(x(1,i)))*log(abs(x(5,i))))...
            +16*(x(2,i)*x(5,i)*cos(x(2,i))+x(1,i)*log(abs(x(3,i))))...
            +(x(4,i)*cos(x(1,i))*cos(x(4,i)))...
            +9*(x(4,i)*cos(x(5,i))*log(abs(x(1,i))))...
            +11*(x(2,i)*cos(x(2,i)))...
            +(18*(x(2,i)*cos(x(2,i))*cos(x(3,i))+x(3,i)*cos(x(1,i))*log(abs(x(1,i))))...
            +((x(5,i)^2)*cos(x(1,i)))...
            +2*(cos(x(5,i))*log(abs(x(1,i)))*log(abs(x(4,i))))...
            +2*(x(1,i)*x(4,i)*log(abs(x(3,i)))+x(2,i)*x(5,i)*log(abs(x(2,i))))...
            +(cos(x(3,i))*cos(x(4,i))*log(abs(x(5,i))))...
            +18*(x(3,i)*cos(x(3,i))*cos(x(5,i))+(cos(x(1,i))^2)*log(abs(x(2,i))))^2)*u(i));
```

```
        %giving the next values for x1-x5
         for z=1:5
             x(z,i+1)=x(z,i)+xdot(z)*dt;
         end


         %computing the next output
         y(i+1)=x(1,i+1);

         currentperf=currentperf + abs(y(i));
    end

    if (abs(currentperf) < bestperf)
        bestperf=currentperf
        j
        bestk=k;
        bestx=x;
        bestu=u;
        besty=y;
    end

    for z=1:5
        k(z)=bestk(z)-1+2*rand;
    end

    if counter==1000
        j
        counter=0;
    end
end

bestsol=bestperf
best_controller=bestk'
lasty=besty(simiter)

% simulation_plots( performance,x1,x2,x3,x4,x5,u,y,t)
figure(1);
for z=1:5
    subplot(2,3,z);plot(t,bestx(z,:));title(['x',num2str(z)]);
end
subplot(2,3,6);plot(t(1:simiter),bestu);title('u');
figure(2);plot(t,besty);title('y');
elapsedtime=toc
```

για το μη γραμμικό σύστημα θα έχουμε σαν προσομοίωση του y, x1-x5 και u:





Ζητούμενο 4

Σχεδιάστε ελεκγτή της μορφής $u = Kx + \frac{\theta_1 f_1(x)+\theta_2 f_2(x)+\cdots+\theta_6 f_6(x)}{u_1 g_1(x)+u_2 g_2(x)+\cdots+u_6 g_6(x)}$, όπου ο πίνακας K και οι παράμετροι$\theta_1 \dots \theta_6, u_1 \dots u_6$, υπολογίζονται κάνοντας χρήση τυχαιών διαταραχών όπως στο ερώτημα 2

Απάντηση 4

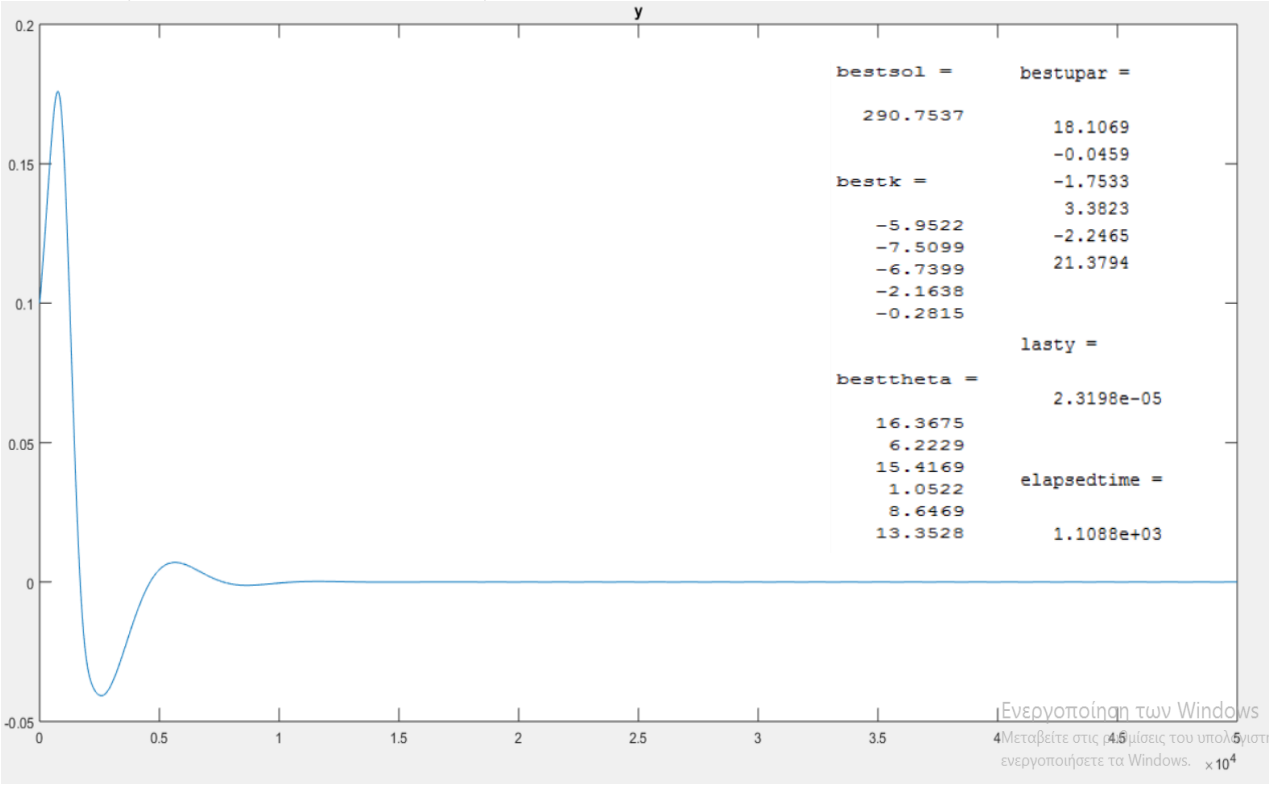
```
clear
clc
%time counter
tic
%initializing the time scope and the time step
dt=0.001;
%how many searches for K
iter=15000;
%itterations of the simulation
simiter=50000;
t=0:simiter;

%k-parameters
k=zeros(1,5);
%theta parameters
theta=[17.5 6.5 15.5 1.5 9.5 10.5];
%u parameters
upar=[17.5 0.5 1.5 2.5 1.5 18.5];
%performance tracking
bestperf=2e+5;
bestk=k;
besttheta=theta;
bestupar=upar;
bestx=zeros(5,simiter+1);
bestu=zeros(1,simiter);
besty=zeros(1,simiter);
xdot=zeros(1,5);

counter=0;

for j=1:iter

    counter=counter+1;
    %initializing the x u y arrays
    x=[0.1*ones(5,1) zeros(5,simiter)];
    u=zeros(1,simiter);
    y=[x(1,1) zeros(1,simiter-1)];
    currentperf=0;

    for i=1:simiter
        if (abs(currentperf)>bestperf)
            break
        end
        %computing the control
        u(i)=control(k,x(:,i),theta,upar);

        %computing the x1-x5dot
        xdot=[x(2,i) x(3,i) x(4,i) x(5,i) 0];

        %this is my linear system
        xdot(5)=1/1000*(18*(x(5,i)*cos(x(2,i))*log(abs(x(5,i)))+x(3,i)*cos(x(3,i))*cos(x(5,i)))...
            +7*(x(3,i)*log(abs(x(1,i)))*log(abs(x(5,i))))...
            +16*(x(2,i)*x(5,i)*cos(x(2,i))+x(1,i)*log(abs(x(3,i))))...
            +(x(4,i)*cos(x(1,i))*cos(x(4,i)))...
            +9*(x(4,i)*cos(x(5,i))*log(abs(x(1,i))))...
            +11*(x(2,i)*cos(x(2,i)))...
            +(18*(x(2,i)*cos(x(2,i))*cos(x(3,i))+x(3,i)*cos(x(1,i))*log(abs(x(1,i))))...
            +((x(5,i)^2)*cos(x(1,i)))...
            +2*(cos(x(5,i))*log(abs(x(1,i)))*log(abs(x(4,i))))...
            +2*(x(1,i)*x(4,i)*log(abs(x(3,i)))+x(2,i)*x(5,i)*log(abs(x(2,i))))...
            +(cos(x(3,i))*cos(x(4,i))*log(abs(x(5,i))))...
            +18*(x(3,i)*cos(x(3,i))*cos(x(5,i))+(cos(x(1,i))^2)*log(abs(x(2,i)))))^2)*u(i));
```

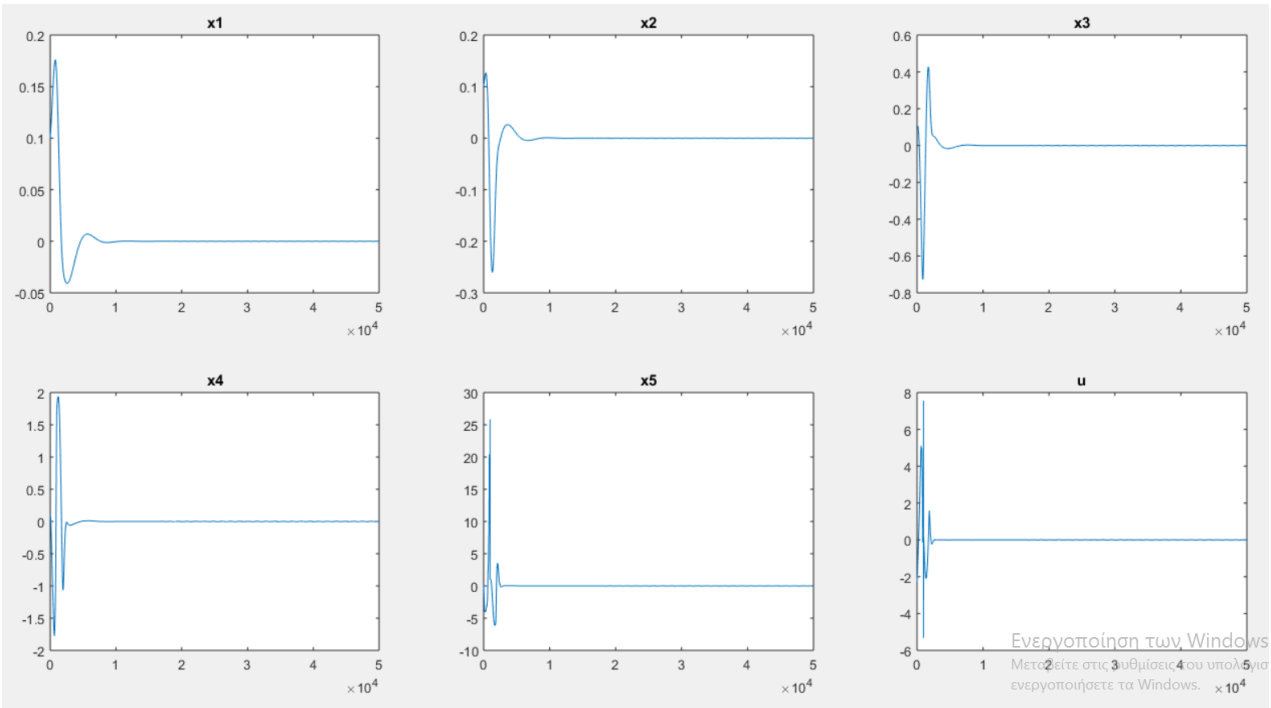
```
%giving the next values for x1-x5
         for z=1:5
             x(z,i+1)=x(z,i)+xdot(z)*dt;
         end


         %computing the next output
         y(i+1)=x(1,i+1);

         currentperf=currentperf + abs(y(i));
    end

    if (abs(currentperf) < bestperf)
        bestperf=currentperf
        j
        bestk=k;
        besttheta=theta;
        bestupar=upar;
        bestx=x;
        bestu=u;
        besty=y;
    end

    for z=1:5
        k(z)=bestk(z)-1+2*rand;
    end

    for z=1:6
        theta(z)=besttheta(z)-1+2*rand;
        upar(z)=bestupar(z)-1+2*rand;
    end

    if counter==1000
        j
        counter=0;
    end
end

bestsol=bestperf
bestk=bestk'
besttheta=besttheta'
bestupar=bestupar'
lasty=besty(simiter)
% simulation_plots( performance,x1,x2,x3,x4,x5,u,y,t)
figure(1);
for z=1:5
    subplot(2,3,z);plot(t,bestx(z,:));title(['x',num2str(z)]);
end
subplot(2,3,6);plot(t(1:simiter),bestu);title('u');
figure(2);plot(t,besty);title('y');
elapsedtime=toc
```

και η συνάρτηση control που χρησιμοποιείται παραπάνω είναι :

```
function [ u ] = control( k, x , theta, upar )
    u1=k*x;
    u2=theta(1)*(x(5)*cos(x(2))*log(abs(x(5)))+x(3)*cos(x(3))*cos(x(5)))...
       +theta(2)*(x(3)*log(abs(x(1)))*log(abs(x(5))))...
       +theta(3)*(x(2)*x(5)*cos(x(2))+x(1)*log(abs(x(3))))...
       +theta(4)*(x(4)*cos(x(1))*cos(x(4)))...
       +theta(5)*(x(4)*cos(x(5))*log(abs(x(1))))...
       +theta(6)*(x(2)*cos(x(2)));
    u3=(upar(1)*(x(2)*cos(x(2))*cos(x(3))+x(3)*cos(x(1))*log(abs(x(1))))...
        +upar(2)*((x(5)^2)*cos(x(1)))...
        +upar(3)*(cos(x(5))*log(abs(x(1)))*log(abs(x(4))))...
        +upar(4)*(x(1)*x(4)*log(abs(x(3)))+x(2)*x(5)*log(abs(x(2))))...
        +upar(5)*(cos(x(3))*cos(x(4))*log(abs(x(5))))...
        +upar(6)*(x(3)*cos(x(3))*cos(x(5))+(cos(x(1))^2)*log(abs(x(2)))))^2;
    u=u1+u2/u3;
end
```

ο αλγόριθμος έδωσε τα εξής αποτελέσματα :





για αυτόν τον ελεγκτή οι τιμές θ και u θα πρέπει να είναι κοντά (+ ή – 05) στις αρχικές παραμέτρους, για να δώσει κάποιο αποτέλεσμα. Αν βάλουμε τιμές που απέχουν >5 απόσταση από τις τιμές που έχουμε ο ελεγκτή δεν είναι αποτελεσματικός και δεν έχουμε κάποιο καλό αποτέλεσμα.

Συμπεράσματα

Παρατηρώντας τα αποτελέσματα από τις παραπάνω προσομοιώσεις καταλήγουμε στο εξής. Η εφαρμογή του pid ελέγχου σε συστήματα με συναρτήσεις μεταφοράς μεγαλύτερου του 3ου βαθμού είναι σχεδόν αδύνατο να είναι αποτελεσματική. Στο συγκεκριμένο σύστημα που ήταν 6ου βαθμού ο pid έλεγχος, ήταν αδύνατον να οδηγήσει στην ευστάθειά το σύστημα.
Όσον αφορά τον ελεγκτή γραμμικού τετραγωνικού ελέγχου όμως είδα ότι παίρνουμε πολύ καλά αποτελέσματα και για το γραμμικό, αλλά και για το μη γραμμικό σύστημα μας, αρκετά νωρίς και σε πολύ καλό χρόνο. Πολύ καλά αποτελέσματα παίρνουμε, όπως θα περιμέναμε άλλωστε και από τον ελεγκτή του τέταρτου ερωτήματος. Ωστόσο για το συγκεκριμένω σύστημα αρκεί ένας ελεγκτής του τρίτου ερωτήματος. Για το συγκεκριμένο σύστημα Λοιπόν θα διαλέγαμε το ελεγκτή του τρίτου ερωτήματος γιατί όχι μόνο δίνει εξίσου καλά αποτελέσματα με τον ελεγκτή του 4ου, αλλά χρησιμοποιώντας αυτόν γλυτώνουμε υπολογιστική πολυπλοκότητα, ισχύ και χρόνο.