Νευροασαφής έλεγχος και εφαρμογές
Εργασία

Q-learning

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ
ΠΟΛΥΤΕΧΝΕΙΟ

Ρούλιος Χαράλαμπος
Α.Μ. *03114004*
Εξάμηνο 9^{o}

1 Θεωρητική ανάλυση και εκτίμηση της συνάρτησης Q

Αρχικά παρατηρούμε ότι το κριτήριο κόστους είναι της μορφής:

$$J = \sum_{k=0}^{\infty}(x_k^T M x_k + u_k^T R u_k)$$

με

$$M = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R = \rho$$

Με βάση τους πίνακες αυτούς και τους πίνακες A,B του συστήματος, η συνάρτηση $Q(x(k), u(k))$ ισούται με:

$$Q(x(k), u(k)) = x(k)^T M x(k) + u(k)^T R u(k) + \min_{u(k+1)} Q(x(k+1), u(k+1))$$

και θεωρώντας ότι το $\min_{u(k+1)} Q(x(k+1), u(k+1))$ είναι της μορφής:

$$\min_{u(k+1)} Q(x(k+1), u(k+1)) = (Ax(k) + Bu(k))^T P(Ax(k) + Bu(k))$$

τότε η συνάρτηση Q μπορεί να γραφεί ως:

$$\boxed{Q(x(k), u(k)) = [x(k)^T u(k)^T] H \begin{bmatrix} x(k) \\ u(k) \end{bmatrix}, \quad H = \begin{bmatrix} M + A^T P A & A^T P B \\ B^T P A & R + B^T P B \end{bmatrix}} \tag{1}$$

Με βάση τη θεωρία βελτίστου ελέγχου για διακριτού χρόνου συστήματα, θεωρώντας έλεγχο της μορφής $u = Kx$, το βέλτιστο κέρδος K ισούται με:

$$K = -(R + B^T P B)^{-1} B^T P A \tag{2}$$

ενώ το P θα πρέπει να ικανοποιεί την αντίστοιχη εξίσωση riccati. Παρατηρώντας τις εξισώσεις (1) και (2), το κέρδος μπορεί να γραφεί συναρτήσει του πίνακα H ως:

$$\boxed{K = -H_{22}^{-1} H_{21}} \tag{3}$$

Η μέθοδος Q-learning είναι μια μέθοδος η οποία ουσιαστικά προσπαθεί να εκτιμήσει επαναληπτικά το $Q(x(k), u(k))$ και συνεπώς τον πίνακα H και το κέρδος K. Η βασική λοιπόν εξίσωση του επαναληπτικού αλγορίθμου που συνδέει την προηγούμενη εκτίμηση i με την επόμενη $i+1$, είναι η:

$$\hat{Q}_{i+1}(x(k), u(k)) = x(k)^T M x(k) + u(k)^T R u(k) + \min_{u(k+1)} \hat{Q}_i(x(k+1), u(k+1))$$

η οποία στη τελική μορφή της, και αφού το u έχει αντικατασταθεί με $u(k) = \hat{K}_i x(k)$ (όπου $\hat{K}_i$ η i-οστή εκτίμηση του K) είναι:

$$\hat{Q}_{i+1}(x(k), u(k)) = [x(k)^T u(k)^T] \hat{H}_{i+1} \begin{bmatrix} x(k) \\ u(k) \end{bmatrix}$$

$$= x(k)^T M x(k) + u(k)^T R u(k) + x^T(k+1)[I_n \hat{K}_i^T]\hat{H}_i \begin{bmatrix} I_n \\ \hat{K}_i \end{bmatrix} x(k+1) \tag{4}$$

Τώρα υποθέτοντας ότι γνωρίζουμε l τιμές της εισόδου u ξεκινώντας από το δείγμα $k = 0$ και τις αντίστοιχες $l+1$ τιμές των καταστάσεων $x(k)$, μπορεί σύμφωνα με την εξίσωση (4) να σχηματιστεί ο πίνακας:

$$d_i = \begin{bmatrix} x(0)^T M x(0) + u(0)^T R u(0) + x^T(1)[I_n \hat{K}_i^T]\hat{H}_i \begin{bmatrix} I_n \\ \hat{K}_i \end{bmatrix} x(1) \\ \vdots \\ x(l)^T M x(l) + u(l)^T R u(l) + x^T(l+1)[I_n \hat{K}_i^T]\hat{H}_i \begin{bmatrix} I_n \\ \hat{K}_i \end{bmatrix} x(l+1) \end{bmatrix} \tag{5}$$

Έτσι πλέον η εξίσωση (4) μπορεί να γραφεί και ως:

$$[x(k)^T u(k)^T]\hat{H}_{i+1} \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} = d_i(k) \tag{6}$$

Στη συνέχεια, ορίζοντας ως Z τον πίνακα γινομένων όλων των μεταβλητών του επαυξημένου πίνακα κατάστασης $[x^T(k)u^T(k)]^T$ για όλα τα δείγματα από 0 έως l, καταλήγουμε στην εξίσωση (7), στην οποία το W_{i+1} είναι μια ¨ξεδιπλωμένη' μορφή του πίνακα $\hat{H}_{i+1}$ και τα στοιχεία του είναι τα $(n+m)(n+m+1)/2$ άγνωστα στοιχεία του $\hat{H}_{i+1}$ σε μορφή διανύσματος (ο πίνακας $H_{(n+m)x(n+m)}$ είναι συμμετρικός, συνεπώς θα πρέπει να βρεθούν συνολικά $(n+m)(n+m+1)/2$ άγνωστοι):

$$Z = \begin{bmatrix} x_1^2(0) \cdots x_n^2(0) \;\; u_1^2(0) \cdots u_m^2(0) \;\; 2x_1(0)x_2(0) \cdots 2u_{m-1}(0)u_m(0) \\ \vdots \\ x_1^2(l) \cdots x_n^2(l) \;\; u_1^2(l) \cdots u_m^2(l) \;\; 2x_1(l)x_2(l) \cdots 2u_{m-1}(l)u_m(l) \end{bmatrix}$$

και

$$\boxed{Z \cdot W_{i+1} = d_i} \tag{7}$$

Από την παραπάνω εξίσωση βλέπουμε ότι η λύση της, δηλαδή η εύρεση του W_{i+1} οδηγεί άμεσα στην εύρεση της επόμενης εκτίμησης του πίνακα $\hat{H}_{i+1}$. Για να μπορέσει λοιπόν να λειτουργήσει ο επαναληπτικός αλγόριθμος, θα πρέπει, όπως είδαμε και προηγουμένως, να υπάρχει ένας γνωστός αριθμός δειγμάτων της εισόδου και των καταστάσεων.

Από τη μια πλευρά οι πίνακες κατάστασης A και B είναι άγνωστοι σε μας, αλλά μπορούμε να λάβουμε δοκιμαστικές τιμές των καταστάσεων x βάζοντας δοκιμαστικές εισόδους $u(i)$ και παρατηρώντας την έξοδο $x(i+1)$ (γνωρίζοντας βέβαια την προηγούμενη κατάσταση $x(i)$).

Επιπλέον, για τον πίνακα Z παρατηρούμε ότι είναι ένας πίνακας σταθερός και ανεξάρτητος του i, δηλαδή της τρέχουσας επανάληψης του αλγορίθμου και ότι οι διαστάσεις του είναι $lx(n+m)(n+m+1)/2$. Αυτό σημαίνει ότι επιλέγοντας τον αριθμό των δοκιμών l να είναι ίσος με $l = (n+m)(n+m+1)/2$ και φροντίζοντας να κάνουμε κατάλληλες δοκιμές ώστε ο Z να είναι πλήρους τάξης, τότε ο Z προκύπτει αντιστρέψιμος και σταθερός, με αποτέλεσμα το W_{i+1} να μπορεί να προκύψει πάντα από την εξίσωση:

$$\boxed{W_{i+1} = Z^{-1} \cdot d_i} \tag{8}$$

Συνοψίζοντας, κάνοντας αρχικά τις απαραίτητες δοκιμές, βρίσκουμε τους πίνακες Z και Z^{-1} και έπειτα στην i-οστή επανάληψη του αλγορίθμου, βρίσκουμε το d_i από την εξίσωση (5), μετά το W_{i+1} άρα και τον $\hat{H}_{i+1}$ από την εξίσωση (8), και τελικά το κέρδος $\hat{K_{i+1}}$ από την εξίσωση (3).

2 Εκτίμηση των πινάκων Q, H, K μέσω του MATLAB

Θα ακολουθήσουμε τώρα στο MATLAB τη διαδικασία που περιγράφηκε προηγουμένως. Ο κώδικας που ακολουθεί, κάνει αρχικοποίηση των απαραίτητων μεγεθών και μετά υπολογίζει τον πίνακα Z και τον αντίστροφό του.

Ως αρχική τιμή του διανύσματος κατάστασης επιλέγεται η τιμή $x(0) = [1\ \ 2\ \ 3]^T$. Στη συνέχεια για την εύρεση του πίνακα Z, τρέχουμε ένα while loop το οποίο δοκιμάζει 10 τυχαίες ακέραιες τιμές (το $(n+m)(n+m+1)/2$ σε αυτή τη περίπτωση είναι 10, συνεπώς επιλέγουμε το ίδιο l έτσι ώστε να έχουμε τετραγωνικό Z) εισόδου στο διάστημα [-10 10] και για κάθε τυχαία είσοδο, υπολογίζει την επόμενη κατάσταση με βάση την εξίσωση $x_{k+1} = Ax_k + Bu_k$. Παράλληλα συμπληρώνεται η αντίστοιχη γραμμή του πίνακα Z.

Η διαδικασία αυτή συνεχίζεται όσες φορές χρειαστεί, έως ότου ο πίνακας Z που υπολογίζεται να έχει πλήρη τάξη (πράγμα που συνήθως συμβαίνει με τη πρώτη προσπάθεια). Έπειτα, αφού πλέον έχει βεβαιωθεί ότι ο πίνακας αυτός είναι αντιστρέψιμος, υπολογίζεται ο σταθερός αντίστροφος του Z.

```
%% A
%Constants
n = 3;
m = 1;
l = (n+m)*(n+m+1)/2;

A = [0 1 0; 0 0 1; 0 0 0]; %uknown
B = [0; 0; 1]; %uknown
M = eye(n); %from the cost criterion

%Finding Z
%Choosing initial x
x(:,1) = [1; 2; 3];
Z= zeros(l,l);
u = zeros(1,l);

while rank(Z) < l
    for i = 1:10
       %selecting random integer input between -10 and 10
```

```
        u(i) = randi([-10 10],1,1);

        Z(i,:) = [x(1,i)^2 x(2,i)^2 x(3,i)^2 u(i)^2 2*x(1,i)*x(2,i) 2*x(1,i)*x(3,i) 2*x(1,

        %testing input for new x
        x(:,i+1) = A*x(:,i) + B*u(i);
    end
end

Zinv = inv(Z);
```

Αφού έχει υπολογιστεί ο πίνακας $Zinv$, σειρά έχει η δημιουργία της συνάρτησης EstimateQ , η οποία δέχεται ως εισόδους η παράμετρο ro, τον $Zinv$, τις δοκιμαστικές εισόδους και καταστάσεις και τις υπόλοιπες σταθερές του προβλήματος και εκτελεί τον επαναληπτικό αλγόριθμο για την εύρεση του πίνακα H και του κέρδους K.

Ο αλγόριθμος ξεκινά με μηδενικές εκτιμήσεις των H, K ενώ παράλληλα ορίζεται μια μεταβλητή σφάλματος e, η οποία αποτελεί το τετράγωνο της διαφοράς της νέας εκτίμησης του K από τη παλιά και ένας μετρητής j. Ο επαναληπτικός αλγόριθμος λειτουργεί όπως περιγράφηκε προηγουμένως, ενώ η εκτέλεση του σταματά όταν εκτελεστεί τουλάχιστον ένας αριθμός επαναλήψεων (εδώ 10) και όταν το σφάλμα e γίνει μικρότερο από μια τιμή (εδώ 0.0001). Τελικά, η συνάρτηση αυτή επιστρέφει την τελευταία εκτίμηση που υπολογίστηκε για τους πίνακες H, K.

```
function [H, K] = EstimateQ(ro,Zinv,x,u,n,m,l,M)
    %Initial estimation
    H = zeros(n+m,n+m);
    K = zeros(m,n);
    d = zeros(l,1);
    e = 42;

    j = 0;
    %Main loop
    while e > 0.0001 || j < 10
        %First calculate d
        for i = 1:l
            d(i) = x(:,i)'*M*x(:,i) + u(i)'*ro*u(i) + x(:,i+1)'*[eye(n) K']*H*[eye(n); K]
        end

        W = Zinv * d;

        %New estimation of H and K
        H = [W(1) W(5) W(6) W(7)
             W(5) W(2) W(8) W(9)
```

```
                W(6) W(8) W(3) W(10)
                W(7) W(9) W(10) W(4)];

            Knew = -inv(H(4,4))*H(4,1:3);

            e = (Knew-K)*(Knew-K)';
            K = Knew;

            j = j+1;
        end
end
```

Η κλήση τη συνάρτησης από το κεντρικό κώδικα για $ro = 1$, γίνεται από τις παρακάτω γραμμές:

```
%Run the iterative algorithm to estimate H and K
ro = 1;
[Hest, Kest] = EstimateQ(ro,Zinv,x,u,n,m,l,M)
```

και το αποτέλεσμα της κλήσης είναι οι πίνακες:

$$Hest = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}, \quad Kest = 10^{-15} \begin{bmatrix} 0.1665 \\ 0.1665 \\ 0 \end{bmatrix}$$

3 Αναλυτικός υπολογισμός βέλτιστου K και προσομοίωση της συμπεριφοράς του συστήματος

3.1 Υπολογισμός Βέλτιστου K

Στη περίπτωση γραμμικού συστήματος διακριτού χρόνου με linear quadratic κόστος με όριο το άπειρο, το βέλτιστο κέρδος αποδεικνύεται ότι είναι της μορφής της εξίσωσης (3) όπου ο πίνακας P θα πρέπει να ικανοποιεί την εξίσωση Riccati, η οποία στη συγκεκριμένη περίπτωση συστήματος και κόστους είναι η:

$$\boxed{P = M + A^T PA - A^T PB(R + B^T PB)^{-1} B^T PA, \quad P = P^T > 0} \tag{9}$$

Στη περίπτωση μας, ο P είναι ένας πίνακας $3x3$ και επειδή είναι συμμετρικός, έχουμε 6 αγνώστους. Στον κώδικα που ακολουθεί, αρχικά ορίζονται οι συμβολικές μεταβλητές των 6 αγνώστων, και στη συνέχεια ο πίνακας P με συμβολικά στοιχεία. Στη γραμμή 10 τώρα, υπολογίζεται το δεξί μέλος της εξίσωσης (9), συναρτήσει του πίνακα P, εφόσον όλες οι άλλες ποσότητες στην εξίσωση είναι σταθερές και ορισμένες προηγουμένως. Η εκτέλεση της γραμμής αυτής, φέρει αποτέλεσμα:

```
ans =

[ 1,                          0,                          0]
[ 0, - p13^2/(p33 + 1) + p11 + 1,    p12 - (p13*p23)/(p33 + 1)]
[ 0,   p12 - (p13*p23)/(p33 + 1), - p23^2/(p33 + 1) + p22 + 1]
```

Από το αποτέλεσμα αυτό, βλέπουμε ότι η λύση της συγκεκριμένης Ricatti μπορεί να βρεθεί τετριμμένα, χωρίς να χρειαστεί να καταφύγουμε σε αριθμητικές μεθόδους επίλυσης. Βλέπουμε δηλαδή ότι $p_{11} = 1$, $p_{12} = 0$, $p_{13} = 0$ και ότι αντικαθιστώντας τις τιμές αυτές στα υπόλοιπα στοιχεία, προκύπτει κατευθείαν η λύση:

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix} \tag{10}$$

Με το P αυτό βρίσκουμε το βέλτιστο K, το οποίο απέχει μηδαμινά από την εκτίμηση του επαναληπτικού αλγορίθμου:

$$\boxed{K = -(R + B^T P B)^{-1} B^T P A = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}} \tag{11}$$

```
%% B

syms p11 p12 p13 p22 p23 p33

P = [p11 p12 p13
     p12 p22 p23
     p13 p23 p33];

%Ricatti P =
M + A'*P*A - A'*P*B*inv(ro+B'*P*B)*B'*P*A

%From ricatti
p = [1 0 0
     0 2 0
     0 0 3]

%Expected Theoretical optimal gain
R = 1;
Kexpected = -inv(ro+B'*p*B)*B'*p*A
```

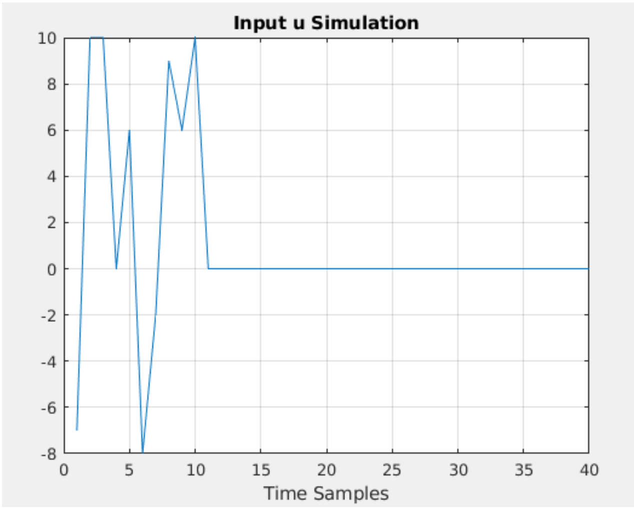
Παρατήρηση: Το βέλτιστο κέρδος παρατηρούμε ότι είναι μηδενικό, δηλαδή ότι η βέλτιστη είσοδος στο συγκεκριμένο σύστημα είναι πάντοτε η μηδενική. Αυτό μπορεί να γίνει εμφανές και από τους πίνακες A,B και το κριτήριο κόστους. Το κριτήριο κόστους αρχικά βλέπουμε ότι γίνεται ελάχιστο, όσο οι καταστάσεις και η είσοδος μικραίνουν και ιδανικά γίνονται μηδέν. Από τους πίνακες A,B βλέπουμε ότι το $x_3(k)$ ισούται πάντα με την προηγούμενη είσοδο $u(k-1)$, ενώ για τα x_1, x_2 ισχύει $x_1(k+1) = x_2(k)$ και $x_2(k+1) = x_3(k)$, δηλαδή το σύστημα ουσιαστικά πραγματοποιεί μια σταδιακή μεταβίβαση της εισόδου από το x_3 στα x_2, x_1. Έτσι τοποθετώντας πάντα είσοδο μηδέν, αναμένουμε σε 3 χρονικά δείγματα να έχουν μηδενιστεί όλες οι καταστάσεις, πράμα που όντως ελαχιστοποιεί το κριτήριο κόστους.

3.2 Προσομοίωση του συστήματος με το MATLAB

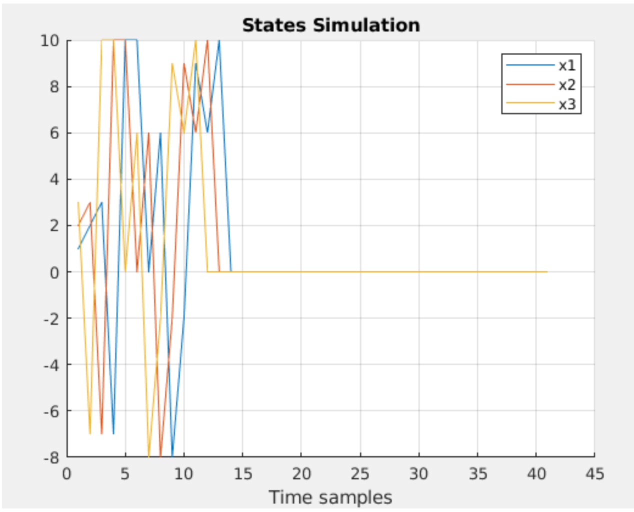
Σύμφωνα με την εκφώνηση η προσομοίωση θα πρέπει να έχει 2 στάδια, ένα στην οποία εφαρμόζεται η ίδια τυχαία είσοδος που είχε εφαρμοστεί για τις l δοκιμές, και στη συνέχεια η είσοδος $u = \hat{K}x$ με την εκτίμηση του βέλτιστου K. Στο MATLAB, πρακτικά η προσομοίωση για το πρώτο στάδιο έχει γίνει ήδη, τη στιγμή που γινόταν υπολογισμός για τον πίνακα Z. Έτσι, η προσομοίωση για τις επόμενες χρονικές στιγμές πραγματοποιείται από το ακόλουθο for loop στο οποίο εφαρμόζεται ο βέλτιστος έλεγχος που έχει εκτιμηθεί. Τέλος, γίνονται τα plot της προσομοίωσης για τις καταστάσεις και την είσοδο.

```
%Simulation on Matlab
simtime = 40;

umat = zeros(1,simtime);
umat(1:l) = u;
for i = (l+1):simtime
    umat(i) = Kest * x(:,i);
    x(:,i+1) = A*x(:,i) + B*umat(i);
end

%Plotting simulation
%States
figure;
hold on
plot(x(1,:))
plot(x(2,:))
plot(x(3,:))
title('States Simulation')
xlabel('Time samples')
legend('x1','x2','x3')
grid on
hold off

%Input
```

```
figure;
plot(umat)
title('Input u Simulation')
xlabel('Time Samples')
grid on

%Simulation on Simulink
usim = [[0:9]' u'];
```



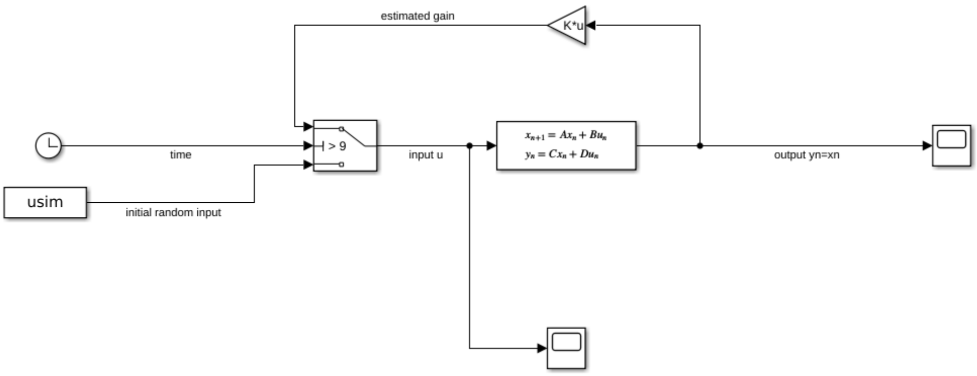
Σχήμα 1: Η είσοδος u στον χρόνο για τη προσομοίωση του συστήματος



Σχήμα 2: Οι καταστάσεις x στον χρόνο για τη προσομοίωση του συστήματος

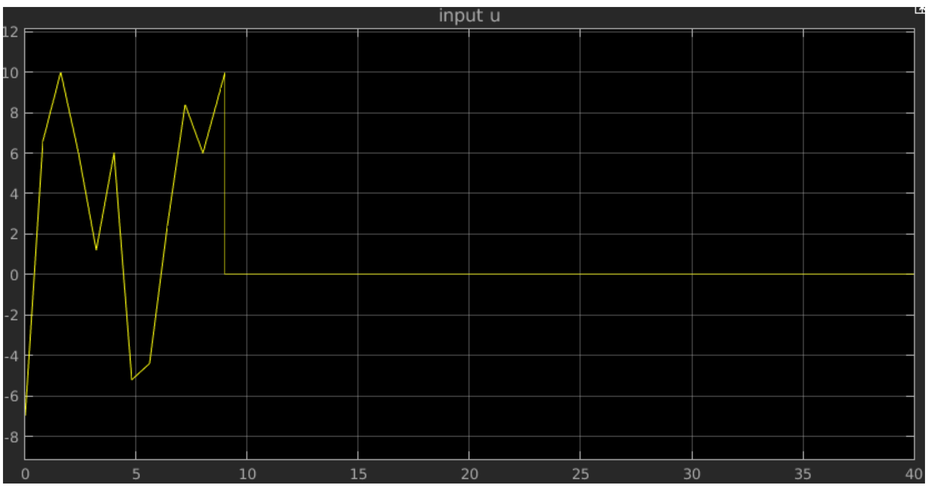
4 Προσομοίωση συστήματος με το Simulink

Για να γίνει η προσομοίωση με το Simulink, αρχικά θα πρέπει να εκτελεστεί ο κώδικας MATLAB έτσι ώστε να αποθηκευτούν στο Workspace οι απαραίτητες σταθερές του συστήματος (π.χ πίνακες A,B) και η μεταβλητή *usim*, η οποία είναι οι πρώτες 10 τυχαίες τιμές της εισόδου, σε μορφή φιλική για το *From Workspace* block του Simulink. Το μοντέλο Simulink που κατασκευάστηκε ώστε να πραγματοποιείται η προσομοίωση με όμοιο τρόπο όπως και στον κώδικα MATLAB, φαίνεται παρακάτω:

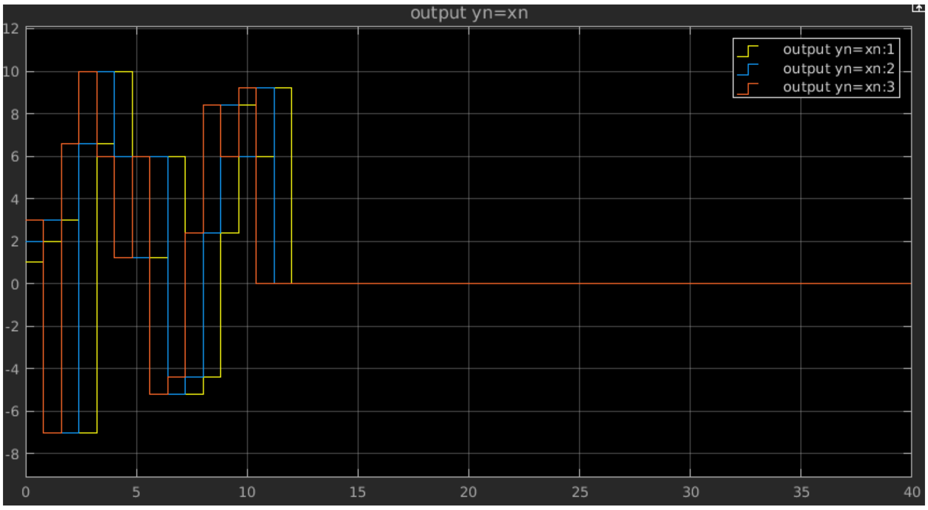


Σχήμα 3: Μοντέλο Simulink για τη προσομοίωση

ενώ οι αντίστοιχες αποκρίσεις της εισόδου και των καταστάσεων είναι:



Σχήμα 4: Η είσοδος u στον χρόνο για τη προσομοίωση του συστήματος στο Simulink



Σχήμα 5: Οι καταστάσεις x στον χρόνο για τη προσομοίωση του συστήματος στο Simulink