Νευρο-Ασαφής Έλεγχος & Εφαρμογές
Εργαστηριακή Άσκηση
Q Learning

Όνομα: Μάριος Παπαχρήστου
Αριθμός Μητρώου: 03115101 Σχολή ΗΜΜΥ
e-mail: papachristoumarios@gmail.com

"Incorporating general intelligence, bodily intelligence, emotional intelligence, spiritual intelligence, political intelligence and social intelligence in AI systems are part of the future deep learning research. " – Amit Ray, Compassionate Artificial Intelligence

1 Σκοπός Εργασίας

Σκοπός της παρούσας εργασίας είναι η ανάπτυξη ενός ελεγκτή state-feedback διακριτού χρόνου για τη σταθεροποίηση ενός ΓΧΑ συστήματος με χρήση Q Learning βάσει τετραγωγνικού κριτηρίου κόστους. Συγκεκριμένα δίνεται το σύστημα

$$x_{k+1} = Ax_k + Bu_k$$

με δυναμικούς πίνακες

$$A = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{1}$$

Και το τετραγωνικό κριτήριο κόστους άπειρου ορίζοντα

$$J = \sum_{i=0}^{\infty} (x_i^T x_i + \rho u[i]^2) \qquad \rho > 0$$

2 Q-Learning

Θέλουμε να βρούμε βέλτιστο ελεγκτή $u_k^* = -Kx_k$ τέτοιο ώστε το σύστημα να είναι ασυμπτωτικά ευσταθές και να ελαχιστοποιείται το κριτήριο κόστους J. Με χρήση ενισχυτικής μάθησης, και συγκεκριμένα Q-Learning, μπορούμε να βρούμε αυτή την είσοδο, αγνοώντας τη δυναμική του συστήματος. Θέλουμε να λύσουμε το εξής πρόβλημα βελτιστοποίησης

$$\min_{u_1,\dots,u_T} \sum_{k=1}^{T} r_k(x_k, u_k)$$
$$\text{s.t. } x_{k+1} = f(x_k, u_k)$$

Στην περίπτωση μας η συνάρτηση $r_k(x_k, u_k)$ είναι η $r_k(u_k, x_k) = x_k^T x_k + \rho u_k^2$. Εισάγουμε την Q-function για την αξιολόγηση μιας πολιτικής K ως την cost το go συνάρτηση

$$Q_K(x_k,u_k) = \sum_{t\geq k} r_t(x_t,u_t) = r_k(x_k,u_k) + Q_K(x_{k+1},u_{k+1}) = r_k(x_k,u_k) + x_{k+1}^T P_K x_{k+1} \qquad (2)$$

όπου ο όρος $x_{k+1}^T P_K x_{k+1}$ είναι το cost-to-go. Η παραπάνω σχέση μπορεί να λυθεί με δυναμικό προγραμματισμό προκειμένου να λάβουμε την βέλτιση συνάρτηση $Q_K^*(x_k,u_k)$ από την εξίσωση του Bellman

$$Q_K^*(x_k,u_k) = r_k(x_k,u_k) + Q_K^*(x_{k+1},u_{k+1})$$

Για την ανεύρεση του βέλτιστου κόστους $J^* = x_0^T P^* x_0$ για το πρόβλημα. Η συνάρτηση ποιότητας μπορεί να γραφεί ως

$$Q_K(x_k,u_k) = z_k^T \begin{pmatrix} I + A^T P A & B^T P A \\ A^T P B & \rho + B^T P B \end{pmatrix} z_k = z_k^T \begin{pmatrix} H_{xx} & H_{xu} \\ H_{ux} & H_{uu} \end{pmatrix} z_k = z_k^T H z_k$$

όπου $z_k = \begin{pmatrix} x_k \\ u_k \end{pmatrix}$ το επαυξημένο διάνυσμα. Η βέλτιστη λύση για την πολιτική K δίνεται ως

$$\frac{\partial Q_K(x_k,u_k)}{\partial u_k} = 0 \iff K^* = -(H_{uu})^{-1} H_{xu}$$

Επαναληπτικά για την εύρεση μιας πολιτικής K_{i+1} από την K_i θεωρούμε μια ακολουθία N τυχαίων εισόδων $u_0,\ldots,u_{N-1}$. Θεωρώντας $t_k^{(i)} = (x_k, K_i x_k)^T$ η αναδρομική σχέση γράφεται ως

$$z_k^T H^{(i+1)} z_k = x_k^T Q x_k + u_k^T R u_k + \left(t_{k+1}^{(i)}\right)^T H^{(i)} t_{k+1}^{(i)} = W_k(H^{(i)})$$

Παρατηρούμε ότι ορίζοντας το διάνυσμα $\bar{z}_k = (z_k \otimes z_k)^T$ και $h^{(i)}$ την διανυσματική αναπαράσταση του $H^{(i)}$ έχουμε ότι

$$\begin{pmatrix} \bar{z}_1 \\ \bar{z}_2 \\ \vdots \\ \bar{z}_N \end{pmatrix} h^{(i+1)} = \begin{pmatrix} W_1 \\ W_2 \\ \vdots \\ W_N \end{pmatrix} \qquad h^{(i+1)} = \begin{pmatrix} \bar{z}_1 \\ \bar{z}_2 \\ \vdots \\ \bar{z}_N \end{pmatrix}^+ \begin{pmatrix} W_1 \\ W_2 \\ \vdots \\ W_N \end{pmatrix} \qquad (3)$$

Όπου Z^+ ο ψευδοαντίστροφος του Z. Το κέρδος τίθεται από τη σχέση του ακροτάτου και η διαδικασία επαναλαμβάνεται την επόμενη εποχή.

Στην υλοποίησή μας με MATLAB, η υλοποίηση της παραπάνω διαδικασίας γίνεται με τη χρήση της ακόλουθης συνάρτησης

```
function [ H, K_opt ] = q_learning(A, B, K, Q, rho, Niter, Hprev, x0
    , u_samples)

    % Initialize variables

    x = x0;

    Y = [];
    X = [];

    % Policy Iteration
    for n = 1 : Niter
        u = u_samples(:, n);
        % Calculate quadratic combinations
```

```
        phi = kron([x; u], [x; u])';

        % Calculate reward function
        r = x' * Q * x + u' * rho * u;

        % Forward pass to the model
        x = A * x + B * u;
        t = [x; K * x];

        % Calculate new quadratic combinations


        r = r + t' * Hprev * t;

        % Append to matrices
        X = [X; phi];
        Y = [Y; r];

    end

    % Solve Least Squares Problem
    psi = pinv(X) * Y;

    dim = length(x) + length(u);
    H = reshape(psi, dim, dim);

    % Get desired elements
    Hux = H(length(x) + 1 : end, 1:length(x));
    Huu = H(length(x) + 1 : end, length(x) + 1 : end);

    % Policy Improvement - Find optimal value of gain
    K_opt = inv(Huu) * Hux;

end
```

3 Ιδανική Περίπτωση

Το ζεύγος (A, B) είναι ελέγξιμο και το $(A, Q^{1/2})$ είναι ανιχνεύσιμο. Η εύρεση ελεκγτή $u_k = K\mathbf{x}_k$ ισοδυναμεί με την επίλυση της Αλγεβρικής Εξίσωσης Ricatti για συστήματα διακριτού χρόνου

$$P = A^T PA - A^T PB(B^T PB + R)^{-1} B^T PA + Q$$

όπου $Q = I \geq 0, R = \rho > 0$ για την ανεύρεση της βέλτιστης εισόδου

$$u_k^* = -(R + B^T PB)^{-1}) B^T PA x_k$$

Παρατηρούμε ότι για κάθε P είναι $A^T PB = 0$ επομένως η Ricatti γίνεται για κάθε $\rho > 0$

$$P = A^T PA + I$$

Αντικαθιστώντας λαμβάνουμε

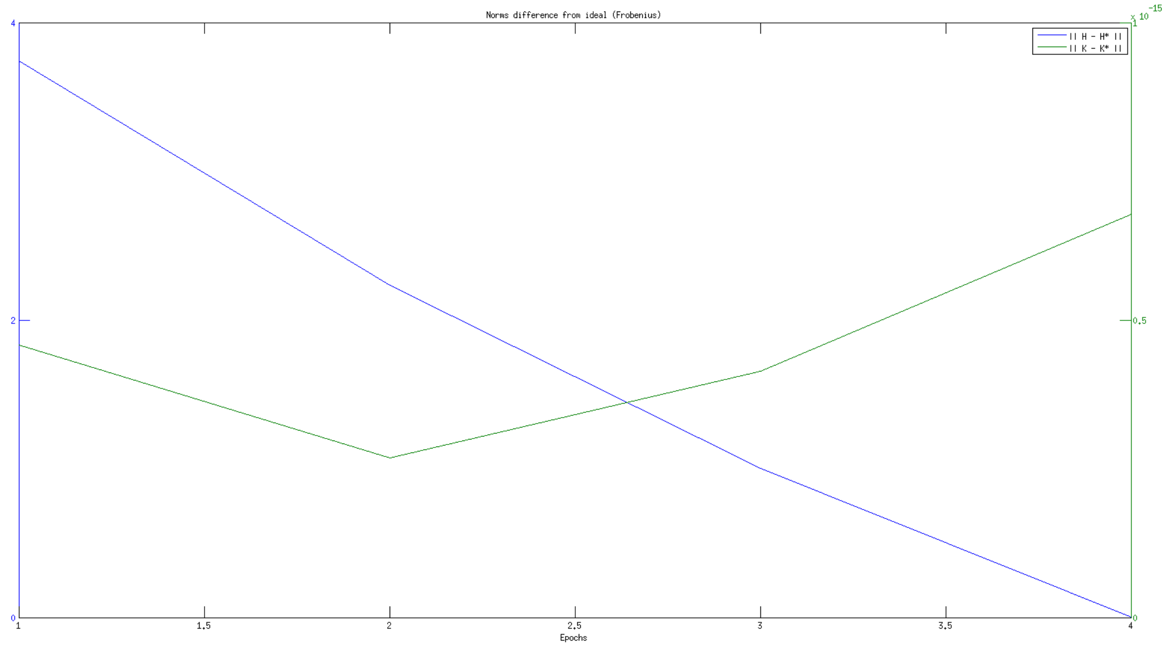
$$p_{ij} = 0 \text{ για κάθε } i \neq j$$

$$p_{11} = 1 \qquad p_{22} = p_{11} + 1 = 2 \qquad p_{33} = p_{22} + 1 = 3$$

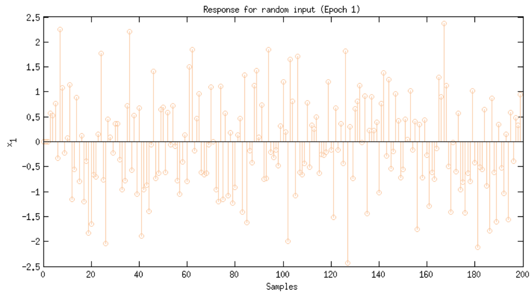
Το οποίο μας δίνει ότι $P = \text{diag}(1, 2, 3)$. Το βέλτιστο κέρδος είναι $K = (0, 0, 0)$ για κάθε $\rho > 0$ επομένως $u_k^* = 0$. Ο ιδανικός πίνακας H είναι ο $H = \text{diag}(1, 2, 3, 4)$

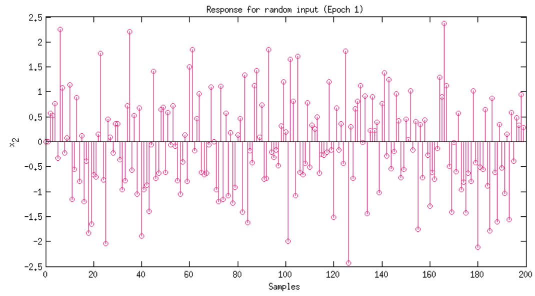
4 Σύγκριση Αποτελεσμάτων

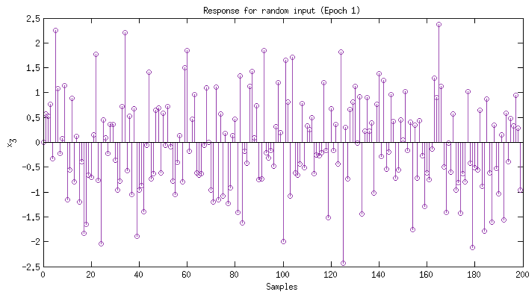
Η γραφική παράσταση των νορμών $\|H - H^*\|_F, \|K - K^*\|_F$ σε κάθε εποχή απεικονίζονται παρακάτω.

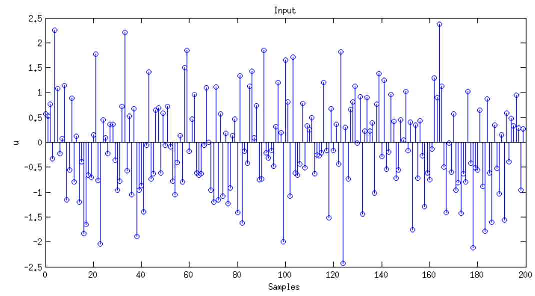


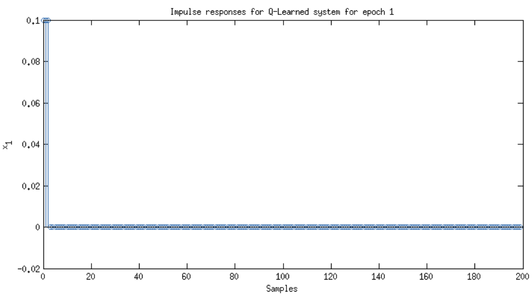
Θεωρήσαμε αρχική συνθήκη $x_0 = (0.1, 0.1, 0.1)^T$. Οι αποκρίσεις σε κάθε εποχή φαίνονται παρακάτω.

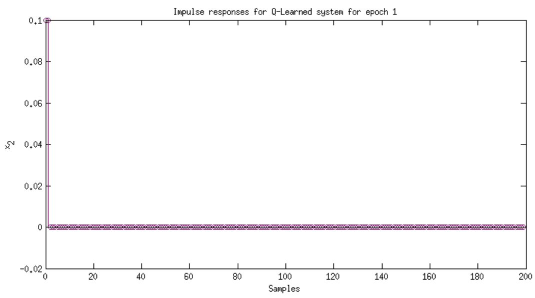

Response for random input (Epoch 1)
x_1
Samples

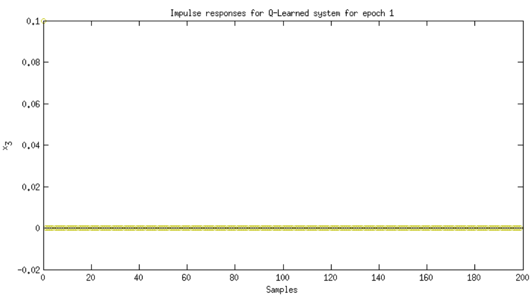

Response for random input (Epoch 1)
x_2
Samples

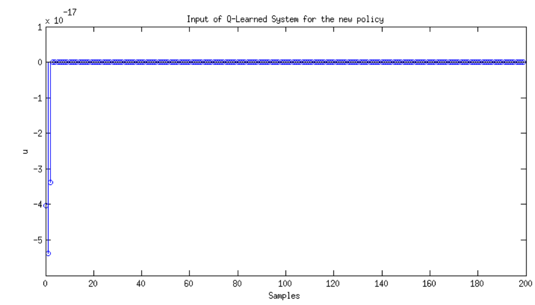

Response for random input (Epoch 1)
x_3
Samples

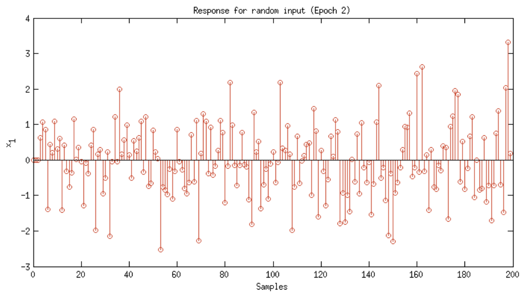

Input
u
Samples

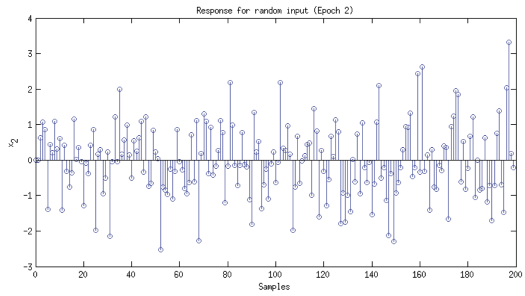

Impulse responses for Q-Learned system for epoch 1
x_1
Samples

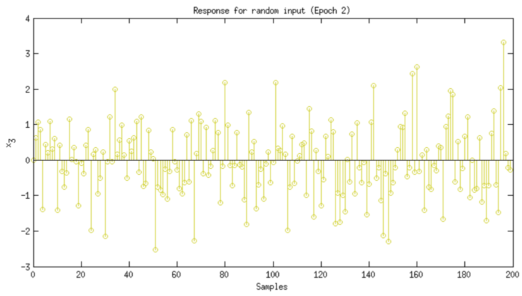

Impulse responses for Q-Learned system for epoch 1
x_2
Samples

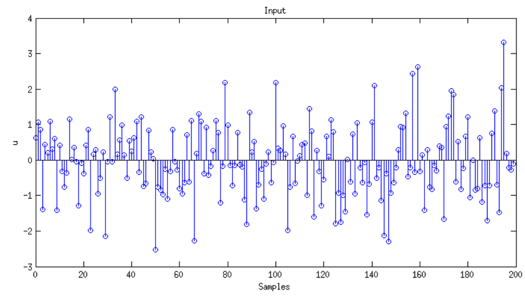

Impulse responses for Q-Learned system for epoch 1
x_3
Samples

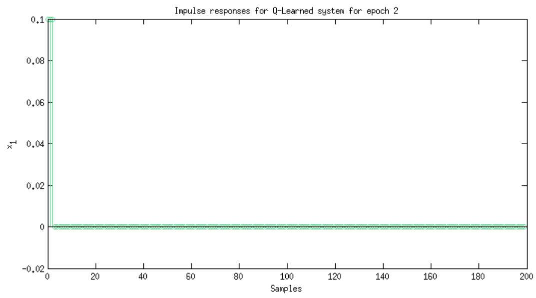

Input of Q-Learned System for the new policy
$\times 10^{-17}$
u
Samples

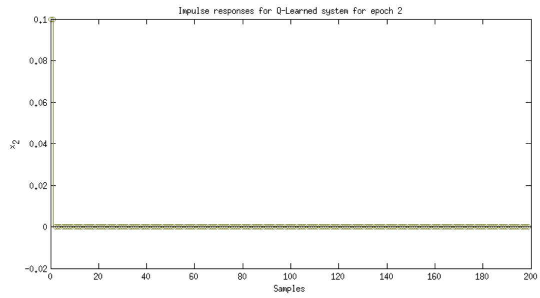

Response for random input (Epoch 2)
x_1
Samples

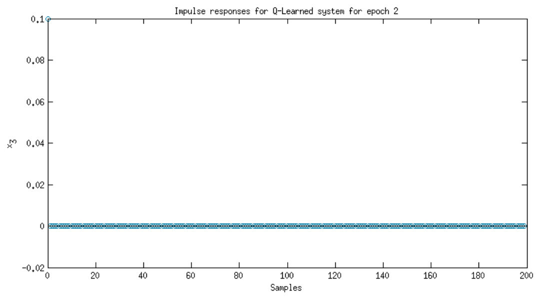

Response for random input (Epoch 2)
x_2
Samples

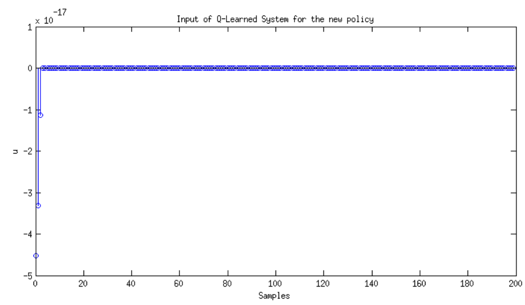

Response for random input (Epoch 2)
x_3
Samples

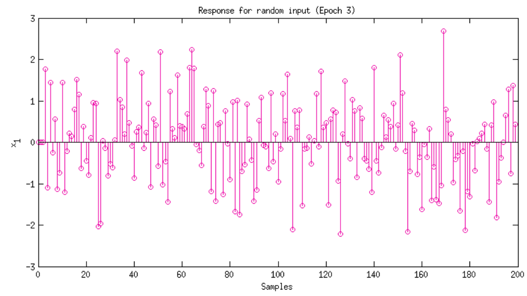

Input
u
Samples

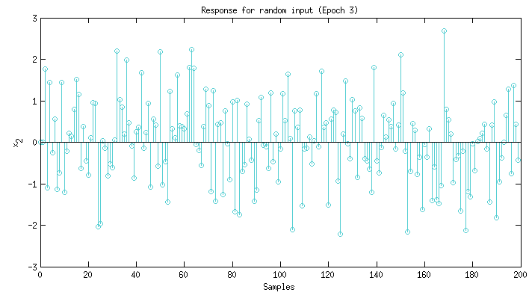

Impulse responses for Q-Learned system for epoch 2
x_1
Samples

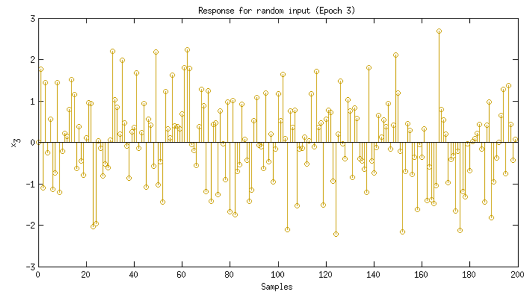

Impulse responses for Q-Learned system for epoch 2
x_2
Samples

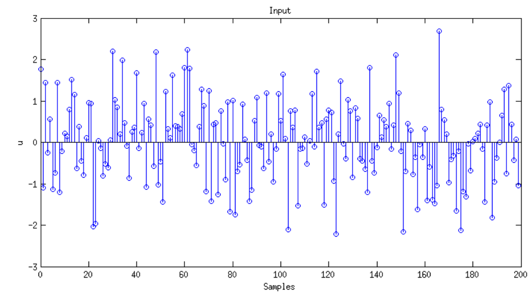

Impulse responses for Q-Learned system for epoch 2
x_3
Samples

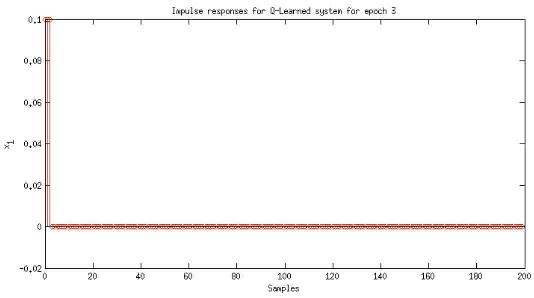

$\times 10^{-17}$
Input of Q-Learned System for the new policy
u
Samples

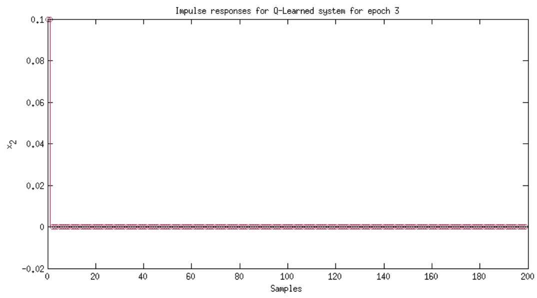

Response for random input (Epoch 3)
x_1
Samples

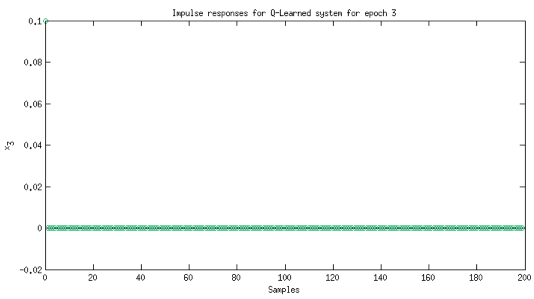

Response for random input (Epoch 3)
x_2
Samples

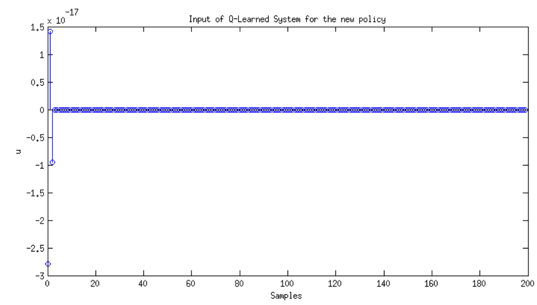

Response for random input (Epoch 3)
x_3
Samples

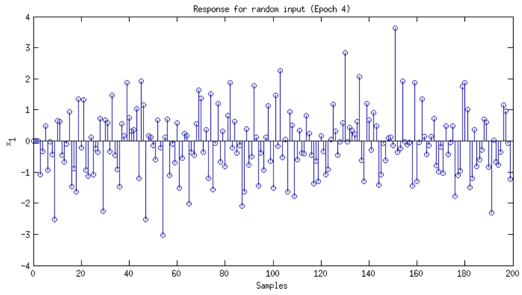

Input
u
Samples

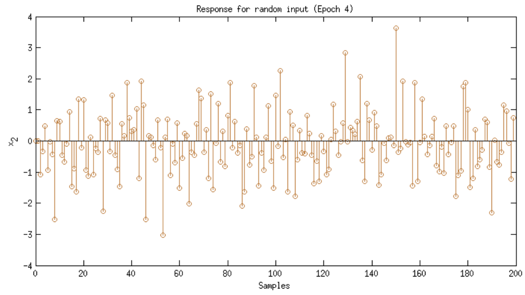

Impulse responses for Q-Learned system for epoch 3
x_1
Samples

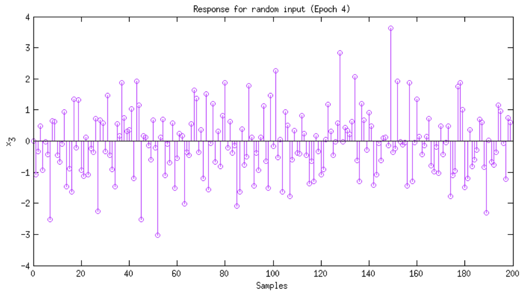

Impulse responses for Q-Learned system for epoch 3
x_2
Samples

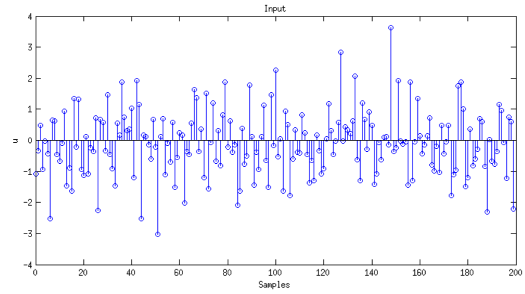

Impulse responses for Q-Learned system for epoch 3
x_3
Samples

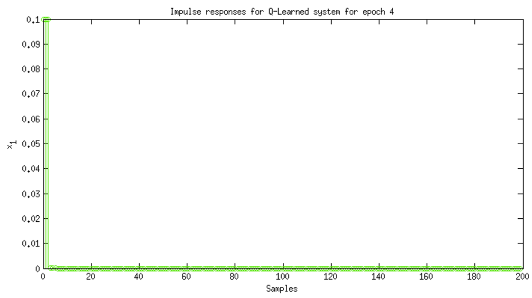

Input of Q-Learned System for the new policy
$\times 10^{-17}$
u
Samples

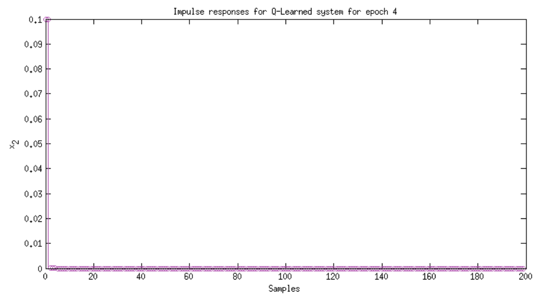


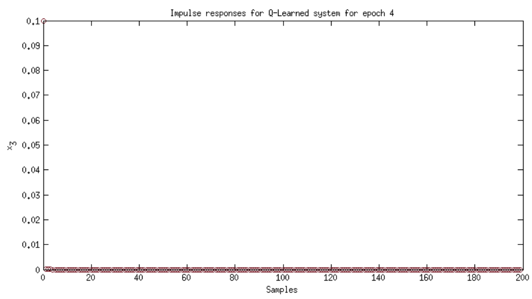


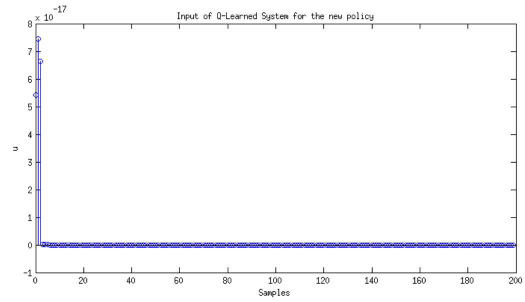












5 Πηγαίος Κώδικας

Το κύριο αρχείο main.m

```
% Q Learning for Optimal Control with LQR Criterion
% Author: Marios Papachristou
% AM: 03115101
% email: papachristoumarios@gmail.com

%% Parameters
% Actual System Dynamics
A = [0 1 0; 0 0 1; 0 0 0];
```

```
B = [0; 0; 1];
dim = size(A, 1) + size(B, 2);


% Sampling Period
Ts = 1;
sys = ss(A, B, zeros(size(A)), zeros(size(B)), Ts);

% Choose initial condition
x0 = 0.1 * ones(length(A), 1);

% LQR Parameters
rho = 1;
Q = eye(size(A));

%% Ideal LQR
% Solution to DARE
[Kid, Pid, e] = dlqr(A, B, Q, rho);

% Ideal H matrix
Hid = [Q + A' * Pid * A A' * Pid * B;
       B' * Pid * A rho + B' * Pid * B];


% Number of iterations
Niter = 200;
time = 0 : Niter - 1;
epochs = 4;

gain_norms = zeros(epochs, 1);
H_norms = zeros(epochs, 1);

%% Q Learning
L = randn(size(B'));
Hp = zeros(dim, dim);
for ep = 1 : epochs

    % Take random inputs
    u_samples = randn(size(B, 2), Niter);

    [H, K] = q_learning(A, B, L, Q, rho, Niter, Hp, x0, u_samples);

    L = K;
    Hp = H;

    % Update norms plot
    gain_norms(ep) = norm(L - Kid, 'fro');
    H_norms(ep) = norm(H - Hid, 'fro');

    % Plot results for epoch
    Aq_c = A - B * K;
```

```
    q_learning_model = ss(Aq_c, zeros(size(B)), zeros(size(A)), zeros(size(B)), Ts);

    [~, ~, x_q] = lsim(q_learning_model, zeros(1, Niter), time, x0);

    figure;
    for i = 1 : length(x0)
        subplot(2, 2, i);
        stem(time, x_q(:, i), 'color', rand(1,3));
        title(sprintf('Impulse␣responses␣for␣Q-Learned␣system␣for␣epoch␣%d', ep));
        xlabel('Samples');
        ylabel(sprintf('x_%d', i));
    end

    subplot(2,2,4);
    stem(time, - K * x_q');
    title('Input␣of␣Q-Learned␣System␣for␣the␣new␣policy');
    xlabel('Samples');
    ylabel('u');

    % Simulate response under random noise

    [~, ~, x_n] = lsim(sys, u_samples, time);
    figure;
    for i = 1 : length(x0)
        subplot(2, 2, i);
        stem(time, x_n(:, i), 'color', rand(1,3));
        title(sprintf('Response␣for␣random␣input␣(Epoch␣%d)', ep));
        xlabel('Samples');
        ylabel(sprintf('x_%d', i));
    end

    subplot(2,2,4);
    stem(time, u_samples);
    title('Input');
    xlabel('Samples');
    ylabel('u');

end

%% Plot norm differences

figure;
plotyy(1:epochs, H_norms, 1 : epochs, gain_norms);
title('Norms␣difference␣from␣ideal␣(Frobenius)');
legend('||␣H␣-␣H*␣||', '||␣K␣-␣K*␣||');
xlabel('Epochs');
```

```
%% Ideal LQR

Aid_c = A - B * Kid;

ideal_model = ss(Aid_c, zeros(size(B)), zeros(size(A)), zeros(size(B
   )), Ts);
[~, ~, x_id] = lsim(ideal_model, zeros(1, Niter), time, x0);


figure;
hold on;
for i = 1 : length(x0)
    stem(time, x_id(:, i), 'color', rand(1,3));
end
title('Impulse␣responses␣for␣ideal␣system')
legend('x1', 'x2', 'x3')
xlabel('Samples')
ylabel('State␣vector')

hold off

hold off

figure;
hold on
stem(time, - Kid * x_id');
title('Input␣of␣Ideal␣System')
xlabel('Samples')

hold off
```

Για τον υπολογισμό των H, K

```
function [ H, K_opt ] = q_learning(A, B, K, Q, rho, Niter, Hprev, x0
    , u_samples)

    % Initialize variables

    x = x0;

    Y = [];
    X = [];

    % Policy Iteration
    for n = 1 : Niter
        u = u_samples(:, n);
        % Calculate quadratic combinations
        phi = kron([x; u], [x; u])';

        % Calculate reward function
        r = x' * Q * x + u' * rho * u;

        % Forward pass to the model
        x = A * x + B * u;
        t = [x; K * x];

        % Calculate new quadratic combinations


        r = r + t' * Hprev * t;

        % Append to matrices
        X = [X; phi];
        Y = [Y; r];

    end

    % Solve Least Squares Problem
    psi = pinv(X) * Y;

    dim = length(x) + length(u);
    H = reshape(psi, dim, dim);

    % Get desired elements
    Hux = H(length(x) + 1 : end, 1:length(x));
    Huu = H(length(x) + 1 : end, length(x) + 1 : end);

    % Policy Improvement - Find optimal value of gain
    K_opt = inv(Huu) * Hux;

end
```

Αναφορές

[1] Bradtke, Steven J. "Reinforcement learning applied to linear quadratic regulation." Advances in neural information processing systems. 1993.