

# DÙNG MẠNG NEURON ĐỂ NHẬN DẠNG HỆ THỐNG ĐỘNG NHIỀU CHIỀU CÓ TRỄ

Võ Phước Như An

Đại học Bách Khoa TP.Hồ Chí Minh

(Bài nhận ngày 21 tháng 6 năm 2001, hoàn chỉnh sửa chữa ngày 11 tháng 10 năm 2001)

## TÓM TẮT

Kỹ thuật dùng mạng neuron trong nhận dạng hệ thống động được áp dụng trong nhiều lĩnh vực như viễn thông, điều khiển tự động... Bài báo nghiên cứu và phát triển sơ đồ và các thuật toán huấn luyện mạng neuron cho bài toán nhận dạng đã nêu trong tài liệu [6] với hệ thống nhiều chiều và có trễ.

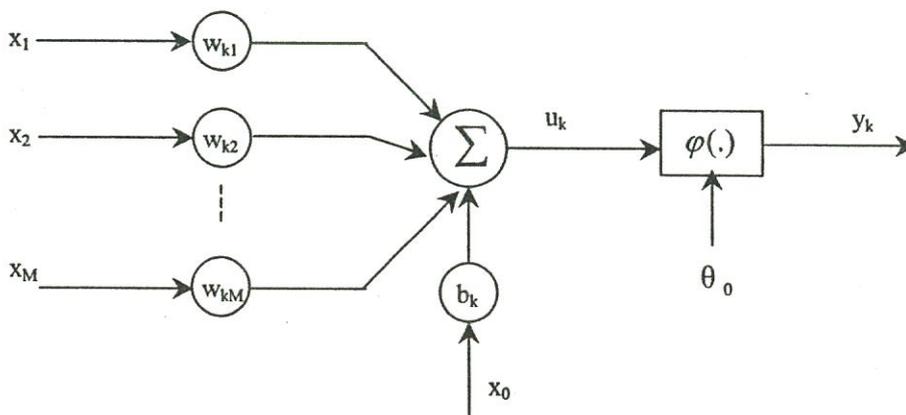
### 1. Giới thiệu

Mô hình mạng neuron được đưa ra đầu tiên vào năm 1943, từ đó đến nay các thuật toán mạng neuron đã được nghiên cứu, phát triển và ứng dụng vào nhiều lĩnh vực [1]-[6]. Những nghiên cứu về bài toán nhận dạng dùng mạng neuron trong tài liệu [6] cũng đã chứng minh được tính ưu việt của việc dùng mạng neuron để nhận dạng

Nhận dạng hệ thống động dùng mạng neuron là vấn đề thực sự rất được nhiều nhà bác học trên thế giới quan tâm và được ứng dụng nhiều trong các hệ thống viễn thông, tự động và các hệ thống động lực ... Bài toán nhận dạng trong tài liệu [6] chỉ đề cập đến hệ thống một đầu vào và không có trễ. Trên thực tế hệ thống nhiều đầu vào và có trễ thường rất phổ biến, trong bài này chúng ta phát triển thêm bài toán nhận dạng đã nêu trong tài liệu [6] cho hệ thống nhiều chiều và có trễ.

### 2. Sơ lược về mạng neuron

Có nhiều loại mạng neuron khác nhau nhưng chúng đều có chung một cấu trúc cơ bản, cấu trúc đó được mô tả ở hình 1.



Hình 1: Cấu trúc của một neuron

Trong đó:

$x_1, x_2, \dots, x_m$  là các tín hiệu đầu vào

$w_{k1}, w_{k2}, \dots, w_{km}$  là trọng số liên kết của neuron thứ k

$b_k$  là trọng số thiên lệch đầu vào của neuron thứ k

$x_0$  là tín hiệu thiên lệch đầu vào của neuron thứ k

$\theta_0$  là mức ngưỡng đầu vào của neuron thứ k

$\phi(\cdot)$  là hàm kích hoạt

$u_k$  là thông số tổng hợp tuyến tính đầu ra

$y_k$  là thông số đầu ra

Các thông số  $u_k$  và  $y_k$  được tính theo công thức:

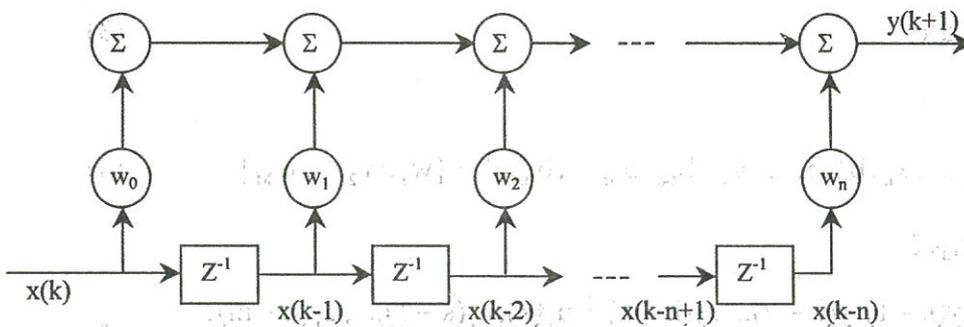
$$u_k = x_0 b_0 + \sum_{i=1}^M w_{ki} x_i$$

$$y_k = \phi(u_k - \theta_0)$$

Mạng neuron có thể có nhiều lớp, tín hiệu đầu ra của lớp này sẽ là tín hiệu đầu vào của lớp kia. Hàm kích hoạt  $\phi$  cũng có nhiều dạng khác nhau: tuyến tính hoặc phi tuyến.

Trường hợp đầu vào mạng neuron là một tín hiệu lấy mẫu rời rạc ta có thể áp dụng mạng neuron có thành phần trễ như hình 2. Mạng này thuộc loại mạng nhiều lớp.

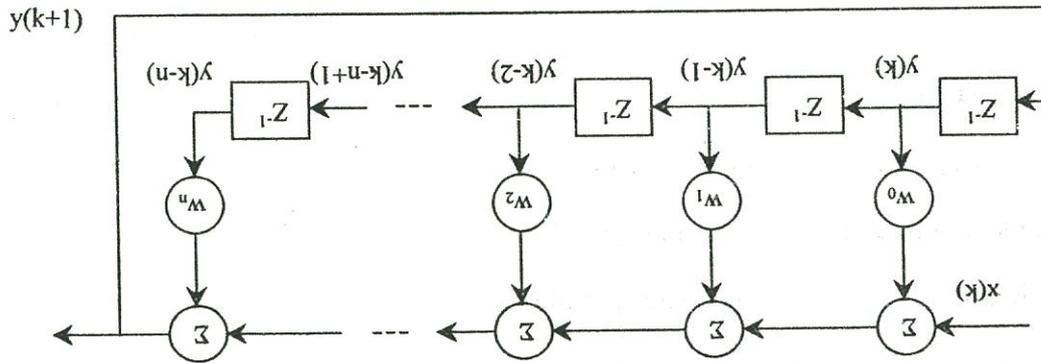
$$y(k+1) = \sum_{i=0}^n w_i x(k-i)$$



Hình 2: Cấu trúc mạng neuron với các thành phần trễ đầu vào

Trường hợp tín hiệu đầu ra của mạng neuron được đánh giá qua tập hợp tín hiệu đầu ra được tính toán hoặc thu thập trong quá khứ ta có thể dùng mạng neuron hồi quy như hình 3

$$y(k+1) = x(k) + \sum_{i=0}^n w_i y(k-i)$$



Hình 3: Sơ đồ mạng neuron hồi quy

Kết hợp mạng neuron có thành phần trễ (hình 2) và mạng neuron hồi quy (hình3) ta có mạng neuron có các thành phần trễ đầu vào và hồi quy.

### 3. Áp dụng mạng neuron để nhận dạng hệ thống động nhiều chiều có trễ

Xét hệ thống rời rạc trong miền thời gian được biểu diễn:

$$y(k) = f(y(k-1), y(k-2), \dots, y(k-n) : u_1(k), u_1(k-1), \dots, u_1(k-m) : u_2(k-d), u_2(k-d-1), \dots, u_2(k-d-m_d)) \tag{1}$$

Trong đó:  $y(k), u_1(k), u_2(k)$  lần lượt là đầu ra và hai đầu vào của hệ thống tại thời điểm rời rạc  $k$

$n, m, m_d > 0$  là bậc của hệ thống,  $d$  là thời gian trễ  $1 \leq d \leq m_d$

Giả sử hệ thống động là tuyến tính thì hàm  $f$  là tuyến tính. Vì vậy hệ thống có thể được biểu diễn dưới dạng sau :

$$y(k) = \sum_{i=1}^n a_i y(k-i) + \sum_{j=0}^m b_j u_1(k-j) + \sum_{j=0}^{m_d} b_{jd} u_2(k-d-j) \tag{2}$$

Phương trình (2) có thể viết gọn lại dạng sau:

$$y(k) = \beta(k-d)\theta \tag{3}$$

Trong đó:

$$\theta = [a_1, a_2, \dots, a_n : b_0, b_1, \dots, b_m : b_{0d}, b_{1d}, \dots, b_{m_d}]^T = [w_1, w_2, \dots, w_M]^T \tag{4}$$

với  $M = n + m + m_d + 2$

$$\beta(k-d) = [y(k-1), y(k-2), \dots, y(k-n) : u_1(k), u_1(k-1), \dots, u_1(k-m) : u_2(k-d), u_2(k-d-1), \dots, u_2(k-d-m_d)] \tag{5}$$

$$\beta(k-d) = [\beta_1(k-d), \beta_2(k-d), \dots, \beta_N(k-d)]$$

Để bình phương tối thiểu vectơ tham số,  $\hat{\theta}$  có thể được ước lượng dựa trên dữ liệu đầu vào và đầu ra theo thực nghiệm, chúng ta định nghĩa:

$$Y_N = [y(1), y(2), \dots, y(N)]^T \text{ và } \Phi_{Nd} = [\beta(1), \beta(2) \dots \beta(N)]^T \tag{6}$$

$$Y_N = \Phi_{Nd} \theta \tag{7}$$

với  $N \geq n + m + m_d + 2$  (8)

$N$  là số thời gian lấy mẫu, đáp ứng được ước lượng dựa trên bình phương tối thiểu  $\hat{\theta}$  được định nghĩa

$$\hat{Y}_N = \Phi_{Nd} \hat{\theta} \tag{9}$$

với  $\hat{\theta} = [\hat{a}_1(k), \hat{a}_2(k), \dots, \hat{a}_n(k) : \hat{b}_0(k), \hat{b}_1(k), \dots, \hat{b}_m(k) : \hat{b}_{od}(k), \hat{b}_{ld}(k), \dots, \hat{b}_{md}(k)]^T$  (10)  
 $= [\hat{w}_1(k), \hat{w}_2(k), \dots, \hat{w}_M(k)]^T$

Ta chọn tiêu chuẩn đánh giá tối ưu dựa theo phương pháp bình phương tối thiểu [3],[5],[6].

$$E_N = \frac{1}{2} e^T e \rightarrow \min_{\hat{\theta}} \tag{11}$$

ở đây  $e$  : sai số

$$e = Y_N - \Phi_{Nd} \hat{\theta} \tag{12}$$

Điều kiện để tối thiểu hóa (11) là gradient theo  $\hat{\theta}$  của  $E_N = 0$

$$\nabla_{\hat{\theta}} E_N = \Phi_{Nd}^T [Y_N - \Phi_{Nd} \hat{\theta}] = 0 \tag{13}$$

Ta có đánh giá tối ưu:

$$\hat{\theta}_N = \arg \min_{\hat{\theta}} \sum_{k=1}^N (y(k) - \beta(k-d)\hat{\theta})^2 \tag{14}$$

$$\hat{\theta}_N = (\Phi_{Nd}^T \Phi_{Nd})^{-1} \Phi_{Nd}^T Y_N \tag{15}$$

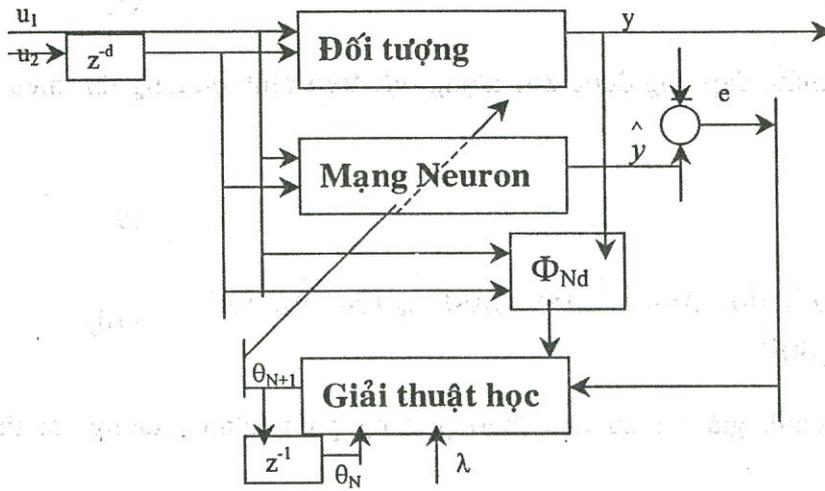
Khi tồn tại nghịch đảo của ma trận hệ số, phương trình (15) có thể được giải bằng chương trình đệ quy và nhận dạng trong thời gian thực. Có thể ước lượng tham số  $\hat{\theta}_{N+1}$  tại thời điểm  $N+1$  dựa trên tham số  $\hat{\theta}_N$  đã được ước lượng tại thời điểm  $N$ , điều này được biểu diễn sau:

$$\hat{\theta}_{N+1} = \hat{\theta}_N + \frac{(\Phi_{Nd}^T \Phi_{Nd})^{-1} \beta^T (N-d+1)}{\lambda^2 + \beta(N-d+1)(\Phi_{Nd}^T \Phi_{Nd})^{-1} \beta^T (N-d+1)} [y(N+1) - \beta(N-d+1)\hat{\theta}_N] \tag{16}$$

Trong đó  $\lambda$  : là thông số bé

Ta cũng có thể ước lượng tham số  $\hat{\theta}_{N+1}$  theo cách khác như sau :

$$\hat{\theta}_{N+1} = \hat{\theta}_N + \frac{(\Phi_{Nd}^T \Phi_{Nd})^{-1} \beta^T (N-d+1)}{\lambda^2 + (\Phi_{Nd}^T \Phi_{Nd})^{-1} \|\beta(N-d+1)\|^2} [y(N+1) - \beta(N-d+1)\hat{\theta}_N] \tag{17}$$



Hình 4: Sơ đồ khối nhận dạng hệ thống động nhiều chiều có trễ dùng mạng neuron.

Giải thuật học được biểu diễn theo công thức (16) hoặc (17)

Trong trường hợp hệ thống có nhiễu:

$$z(k) = y(k) + v(k) \tag{18}$$

với  $z(k)$  là đầu ra của hệ thống tại thời điểm  $k$  khi có nhiễu.

$y(k)$  được biểu diễn theo (2) là đầu ra của hệ thống tại thời điểm  $k$  khi không có nhiễu.

$$y(k) = \sum_{i=1}^n a_i y(k-i) + \sum_{j=0}^m b_j u_1(k-j) + \sum_{j=0}^{m_d} b_{jd} u_2(k-d-j) \tag{2}$$

$v(k)$  là nhiễu trắng phân bố Gauss, có trị trung bình bằng 0 và covariance  $\sigma_v^2$  là ma trận đối xứng xác định không âm.

$$E[v] = 0; \quad E[vv^T] = \sigma_v^2 < \infty;$$

Phương trình (18) có thể viết gọn lại dạng sau:

$$z(k) = \beta(k-d)\theta + v(k) \tag{19}$$

Trong đó:  $\theta$  được biểu diễn theo (4)

$$\theta = [a_1, a_2, \dots, a_n; b_0, b_1, \dots, b_m; b_{0d}, b_{1d}, \dots, b_{m_d}]^T = [w_1, w_2, \dots, w_M]^T \tag{4}$$

với  $M = n+m+m_d+2$

$$\beta(k-d) = [z(k-1) - v(k-1), z(k-2) - v(k-2), \dots, z(k-n) - v(k-n); u_1(k), u_1(k-1), \dots, u_1(k-m); u_2(k-d), u_2(k-d-1), \dots, u_2(k-d-m_d)] \tag{20}$$

$$\beta(k-d) = [\beta_1(k-d), \beta_2(k-d), \dots, \beta_N(k-d)]$$

Để bình phương tối thiểu vectơ tham số,  $\hat{\theta}$  có thể được ước lượng dựa trên dữ liệu đầu vào, đầu ra theo thực nghiệm, chúng ta định nghĩa  $Y_N$  và  $\Phi_{Nd}$  như (6):

$$Y_N = [y(1), y(2), \dots, y(N)]^T \text{ và } \Phi_{Nd} = [\beta(1), \beta(2) \dots \beta(N)]^T \quad (6)$$

$$Z_N = [z(1), z(2), \dots, z(N)]^T \text{ và } V_N = [v(1), v(2), \dots, v(N)]^T \quad (21)$$

$$Z_N = \Phi_{Nd} \theta + V_N \quad (22)$$

$$\text{với } N \geq n + m + m_d + 2 \quad (8)$$

$N$  là số thời gian lấy mẫu, đáp ứng được ước lượng dựa trên bình phương tối thiểu  $\hat{\theta}$  được định nghĩa

$$\hat{Y}_N = \hat{Z}_N = \Phi_{Nd} \hat{\theta} \quad (23)$$

$$\begin{aligned} \text{với } \hat{\theta} &= [\hat{a}_1(k), \hat{a}_2(k), \dots, \hat{a}_n(k) : \hat{b}_0(k), \hat{b}_1(k), \dots, \hat{b}_m(k) : \hat{b}_{0d}(k), \hat{b}_{1d}(k), \dots, \hat{b}_{md}(k)]^T \\ &= [\hat{w}_1(k), \hat{w}_2(k), \dots, \hat{w}_M(k)]^T \end{aligned} \quad (10)$$

Ta đánh giá các tham số của hệ thống bằng cách tối thiểu hóa  $E_N$

$$E_N = \frac{1}{2} e^T W^{-1} e \rightarrow \min_{\theta} \quad (24)$$

$$\text{Với } W = \begin{bmatrix} \frac{1}{\sigma_1^2} & 0 & 0 & 0 \\ 0 & \frac{1}{\sigma_2^2} & 0 & 0 \\ 0 & 0 & \vdots & 0 \\ 0 & 0 & 0 & \frac{1}{\sigma_N^2} \end{bmatrix} \quad (25)$$

$$e = Z_N - \Phi_{Nd} \hat{\theta} \quad (26)$$

$$E_N = \frac{1}{2} [Z_N - \Phi_{Nd} \hat{\theta}]^T W^{-1} [Z_N - \Phi_{Nd} \hat{\theta}] \quad (27)$$

$$\text{hay } E_N = \|Z_N - \Phi_{Nd} \hat{\theta}\|_W^2 \quad (28)$$

Điều kiện để tối thiểu hóa (27) là gradient theo  $\hat{\theta}$  của  $E_N = 0$

$$\nabla_{\hat{\theta}} E_N = W^{-1} \Phi_{Nd}^T [Z_N - \Phi_{Nd} \hat{\theta}] = 0 \quad (29)$$

Ta có đánh giá tối ưu:

$$\hat{\theta}_N = \arg \min_{\hat{\theta}} \sum_{k=1}^N (z(k) - \beta(k-d)\hat{\theta})^2 \quad (30)$$

$$\hat{\theta}_N = (\Phi_{Nd}^T W^{-1} \Phi_{Nd})^{-1} \Phi_{Nd}^T W^{-1} Z_N \quad (31)$$

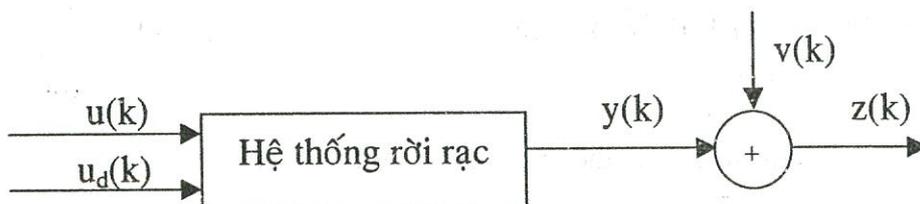
Khi tồn tại nghịch đảo của ma trận hệ số, phương trình (31) có thể được giải bằng chương trình đệ quy và nhận dạng trong thời gian thực. Có thể ước lượng tham số  $\hat{\theta}_{N+1}$  tại thời điểm  $N+1$  dựa trên tham số  $\hat{\theta}_N$  đã được ước lượng tại thời điểm  $N$ , điều này được biểu diễn sau:

$$\hat{\theta}_{N+1} = \hat{\theta}_N + \frac{(\Phi_{Nd}^T \Phi_{Nd})^{-1} \beta^T (N-d+1)}{\sigma_v^2 + \beta(N-d+1)(\Phi_{Nd}^T \Phi_{Nd})^{-1} \beta^T (N-d+1)} [z(N+1) - \beta(N-d+1)\hat{\theta}_N] \quad (32)$$

Ta cũng có thể ước lượng tham số  $\hat{\theta}_{N+1}$  theo cách khác như sau :

$$\hat{\theta}_{N+1} = \hat{\theta}_N + \frac{(\Phi_{Nd}^T \Phi_{Nd})^{-1} \beta^T (N-d+1)}{\sigma_v^2 + (\Phi_{Nd}^T \Phi_{Nd})^{-1} \|\beta(N-d+1)\|^2} [z(N+1) - \beta(N-d+1)\hat{\theta}_N] \quad (33)$$

#### 4. Ví dụ mô phỏng



Hình 5

Cho hệ thống có 2 đầu vào như hình 5.

- Trong đó:  $u, u_d$  là tín hiệu kích thích hệ thống
- $v$  là tín hiệu nhiễu cộng ở ngõ ra
- $y$  là tín hiệu ra không nhiễu của hệ thống
- $z$  là tín hiệu ra bị nhiễu

**Hệ thống trên được biểu diễn bởi phương trình sai phân sau:**

$$z(k) = a_1 y(k-1) + a_2 y(k-2) + b u(k) + b_d u_d(k) + v(k)$$

Vectơ thông số của hệ thống  $\theta = [a_1 \ a_2 \ b \ b_d]$

Dùng Matlab 5.3 để mô phỏng (Phần phụ lục).

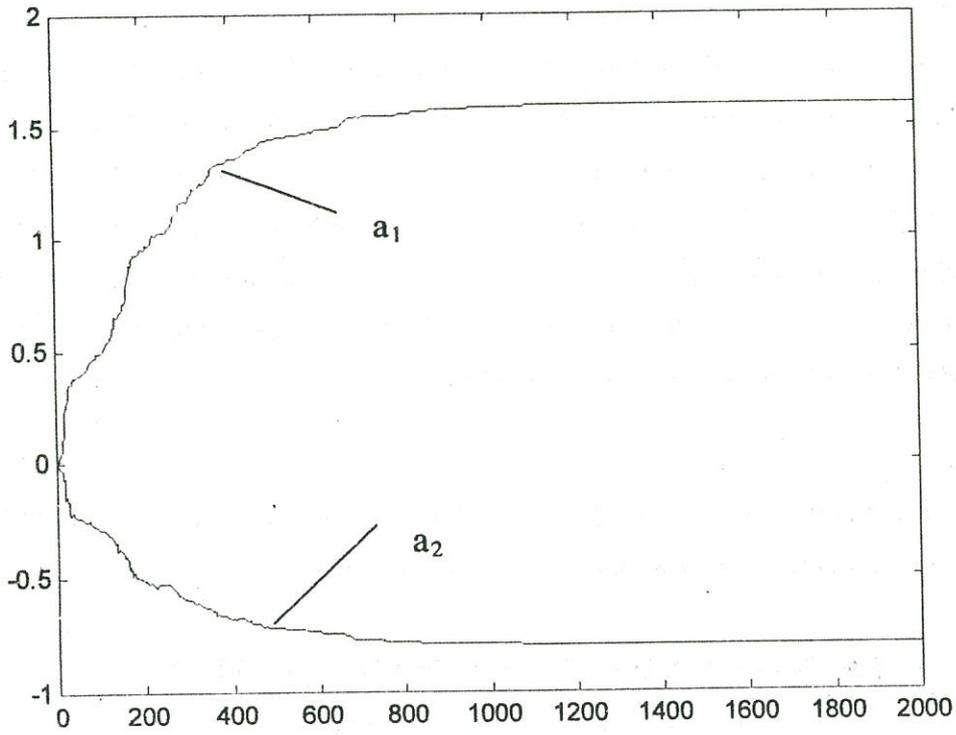
Giả sử hệ thống có vectơ tham số

$$\theta = [1.6000 \ -0.8000 \ 1.2000 \ 2.4000]$$

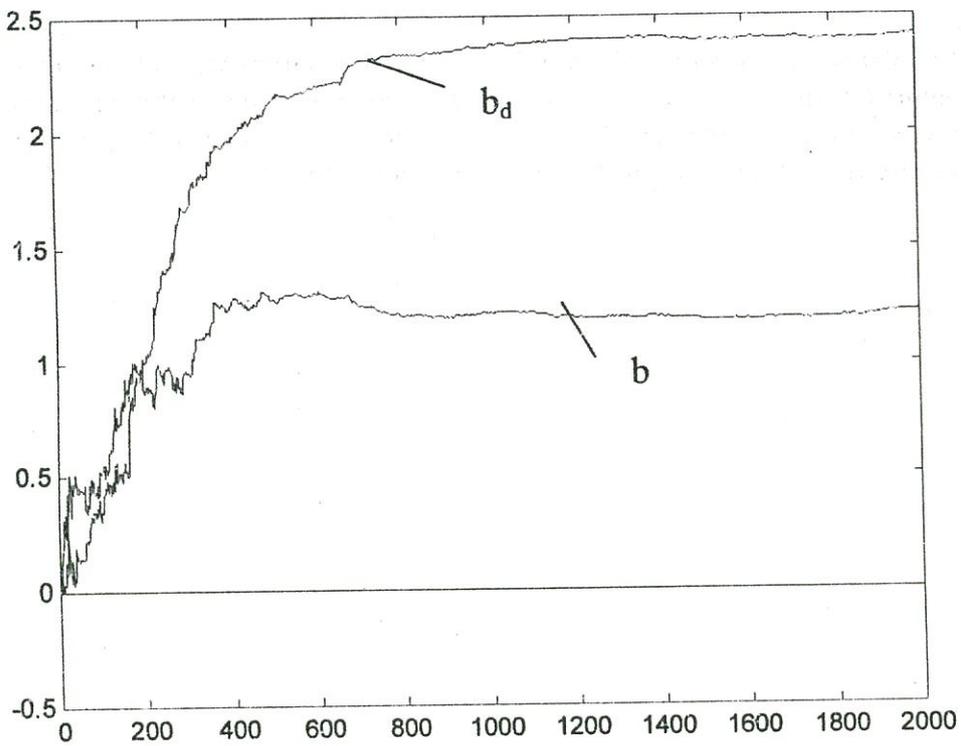
Áp dụng công thức (32), dựa trên dữ liệu đầu vào hệ thống là  $u, u_d$  và dữ liệu đầu ra bị nhiễu được xác định theo thực nghiệm là  $z$  ta ước lượng được vectơ thông số  $\hat{\theta}$  của hệ thống:

$$\hat{\theta}_N = [1.5965 \ -0.7964 \ 1.2202 \ 2.4174] \text{ với thời gian lấy mẫu } N=2000$$

Hình 6,7 trình bày quá trình nhận dạng thông số  $a_1, a_2, b$  và  $b_d$ . Kết quả mô phỏng cho thấy chất lượng nhận dạng cao: sai số nhỏ và hội tụ nhanh, hệ thống nhận dạng đáng tin cậy.



Hình 6: Quá trình nhận dạng thông số  $a_1$  và  $a_2$



Hình 7: Quá trình nhận dạng thông số  $b$  và  $b_d$



## 6. Kết luận

Trong bài này chúng ta đã áp dụng mạng neuron để nhận dạng cho hệ thống động nhiều chiều có trễ. Việc hoàn thiện kỹ thuật tính của mạng neuron cũng là một vấn đề rất quan trọng, điều đó thể hiện qua thời gian huấn luyện mạng. Qua công thức (16) ta thấy việc chọn thông số bé tối ưu  $\lambda$  là rất quan trọng trong việc huấn luyện mạng neuron, đảm bảo tính ổn định của quá trình huấn luyện mạng, làm cho bài toán hội tụ nhanh hay chậm quyết định đến thời gian huấn luyện. Chọn  $\lambda$  tối ưu  $\lambda_{opt}$  là một vấn đề rất đáng quan tâm, chúng ta sẽ xét trong các chương trình huấn luyện mạng neuron tiếp theo.

Tác giả chân thành cảm ơn các thầy GS.TSKH. Nguyễn Thúc Loan, TS. Dương Hoài Nghĩa và TS. Nguyễn Đình Thúc về những gợi ý khoa học quý giá.

## IDENTIFICATION OF DYNAMIC SYSTEM USING NEURAL NETWORK

Vo Phuc Nhu An

### ABSTRACT

Identification of dynamic system using Neural Network was applicated in many domains such as control of industrial process, equalization (in digital communication systems), fault detection ... This paper describes scheme and algorithms training Neural Networks for problem of identification of dynamic system [6], plant is multi-variable and delay.

## PHỤ LỤC

### Phần mềm mô phỏng dùng Matlab 5.3

```
clear all
```

```
% Tin hieu vao u, ud va nhieu v:
```

```
N = 100;
```

```
Q = N; nu = N; nud = N;
```

```
for i = 1 : N,
```

```
    u(i) = rand - 0.5;
```

```
    ud(i) = rand - 0.5;
```

```
    y(i) = 0;
```

```
    v(i) = 0.1*rand - 0.05;
```

```
end
```

```
% Kiem nghiem voi he thong teta
```

```
a = poly([0.8 + j*0.4, 0.8 - j*0.4]);
```

```
a      = [-a(2) -a(3)]      ; n      = size(a,2);
```

```
b      = [1.2]              ; m      = size(b,2);
```

```
bd     = [2.4]              ; md     = size(bd,2);
```

```
d      = 0;
```

```
covv   = 0.1;
```

```
teta   = [a b bd]
```

```
tetam  = [0 0 0 0];
```

```
% Mo phong ngo ra y theo mo hinh biet truooc teta
```

```
for k=1:Q
```

```
    t1=0;t2=0;t3=0;
```

```
    for i=1:n
```

```
        if k-i>0
```

```
            t1= t1+a(i)*y(k-i);
```

```
        end
```

```
    end
```

```
    for i=1:m
```

```
        if ((k-i+1)>0)&((k-i+1)<nu)
```

```
            t2= t2+b(i)*u(k-i+1);
```

```
        end
```

```
    end
```

```
    for i=1:md
```

```
        if ((k-d-i+1)>0)&((k-d-i+1)<nud)
```

```
            t3= t3+bd(i)*ud(k-d-i+1);
```

```
        end
```

```
    end
```

```
    y(k)=t1+t2+t3;
```

```
end
```

```

% Mo phong ngo ra khi co tac dong cua nhieu
z= y+v;

% Tinh phiNd
for k=1:Q
    for i=1:n
        if k-i>0
            phi(k,i)=z(k-i);    %phi(k,i)=z(k-i)-v(k-i)
        end
    end
    for i=n+1:n+m
        if (k+n+1-i)>0 &(k+n+1-i)<nu
            phi(k,i)=u(k+n+1-i);
        end
    end
    for i=n+m+1:n+m+md
        if (k-d+n+m+1-i)>0&(k-d+n+m+1-i)<nud
            phi(k,i)=ud(k-d+n+m+1-i);
        end
    end
end

% Cac du lieu y, v, z va phi da biet, uoc luong tham so tetam
for k=1:Q-1
    a      = inv(phi'*phi)*(phi(k+1,1:n+m+md));
    b(k)   = z(k+1)-phi(k+1,1:n+m+md)*tetam';
    c(k)   = covv + phi(k+1,1:n+m+md)*a;
    delta  = a*b(k)/c(k);
    tetam  = (tetam'+delta)';
    for i = 1:n      , am(k,i)= tetam(i);end
    for i = n+1:n+m  , bm(k,i)= tetam(i);end
    for i = n+m+1:n+m+md , bdm(k,i)= tetam(i);end
end
tetam
figure
time = 1:Q-1;
plot(time,am,'k')
figure
plot(time,bm,'b',time,bdm,'r')

```

k	U	u <sub>d</sub>	v	Z	a <sub>1</sub>	a <sub>2</sub>	b	b <sub>d</sub>
1	0.4620	-0.0242	0.0175	-0.1377	0.0000	0.0000	0.0000	0.0000
2	-0.3837	0.1955	-0.0182	1.3496	0.0226	-0.0191	0.4626	0.5312
3	0.3295	0.2905	0.0300	2.0221	0.0434	-0.0298	0.6716	0.2732
4	0.4895	0.3698	0.0320	2.6358	0.1973	-0.1509	1.5084	0.3728
5	0.4506	0.0689	-0.0394	1.9215	0.5227	-0.3840	0.4794	0.0641
6	0.1458	-0.2593	-0.0411	2.1691	0.7954	-0.5072	1.0901	0.5942
7	0.4876	0.2440	0.0046	2.0544	0.7998	-0.4396	0.7089	0.7007
8	-0.0781	0.2862	0.0310	0.9180	0.8487	-0.3699	0.4877	0.5157
9	0.1288	-0.1814	-0.0197	0.7443	0.8479	-0.3341	0.6595	0.6098
10	-0.4509	0.1021	-0.0276	0.6859	0.8005	-0.2479	0.6360	0.6410
11	0.3587	0.2536	-0.0237	-0.2216	0.9231	-0.3931	0.6319	0.7066
12	0.1691	0.3619	0.0341	-1.4773	0.9361	-0.4070	0.2238	1.1164
13	-0.4822	0.3048	0.0032	-1.5847	0.9649	-0.4320	0.3424	0.7620
14	-0.1900	-0.3762	-0.0421	-1.9273	0.9728	-0.4377	0.3788	1.0939
15	-0.1103	0.4921	-0.0347	-1.2286	0.9726	-0.4375	0.6029	0.7342
16	0.0044	-0.1741	0.0210	-1.5694	0.9741	-0.4391	0.9688	0.9856
17	0.3560	-0.3846	-0.0326	-1.9925	0.9805	-0.4446	1.3894	0.9830
18	0.3904	0.0155	0.0141	-2.3452	1.0039	-0.4646	1.4346	1.2207
19	-0.2782	0.3967	0.0160	-2.4115	1.0053	-0.4666	0.7195	1.5731
20	0.0672	-0.0327	0.0486	-0.9920	1.0203	-0.4823	0.6722	1.2272
21	0.1587	-0.2753	0.0417	0.8197	1.0291	-0.4895	0.6399	1.3423

k	U	$u_d$	v	Z	$a_1$	$a_2$	b	$b_a$
22	0.2283	0.0009	-0.0461	1.2430	1.0294	-0.4901	0.4117	1.2971
23	0.2535	0.0780	0.0460	-0.0544	1.0284	-0.4877	0.4213	1.3184
24	0.4281	0.2568	0.0174	-2.1430	1.0314	-0.4894	0.7532	1.5875
25	-0.3788	0.0659	0.0053	-4.0388	1.1053	-0.5419	0.7381	1.9773
26	0.3268	0.3852	0.0145	-5.0624	1.1141	-0.5372	1.2813	1.9104
27	-0.3055	-0.2638	-0.0360	-4.9262	1.1754	-0.5683	1.6664	1.5643
28	-0.1856	0.4354	0.0073	-5.1745	1.1660	-0.5473	2.0062	2.0407
29	-0.2777	-0.0498	0.0054	-5.5309	1.1657	-0.5406	2.3227	2.1887
30	0.0284	-0.2217	-0.0194	-5.2120	1.1745	-0.5523	2.1934	2.3327
31	0.1341	-0.0183	-0.0432	-5.1941	1.2167	-0.5949	2.3500	2.3921
32	0.4803	0.1566	0.0325	-4.9082	1.2361	-0.6096	2.0760	2.6425
33	0.3878	0.3161	-0.0181	-3.9569	1.2354	-0.6076	1.8107	2.7225
34	0.0912	0.1779	0.0204	-2.1467	1.2782	-0.6531	1.8634	2.6448
35	0.3869	0.2448	-0.0366	-0.1568	1.3348	-0.7015	1.8451	2.6589
36	-0.1332	0.2535	-0.0188	0.8426	1.4185	-0.7579	1.8444	2.6889
37	-0.1445	-0.2592	0.0109	1.1028	1.4792	-0.7799	1.8205	2.7130
38	0.1204	-0.2856	0.0129	0.4811	1.4709	-0.7480	1.7226	2.6640
39	0.1766	-0.2941	0.0400	-0.4066	1.4585	-0.7305	1.6942	2.6501
40	0.0707	-0.1943	-0.0176	-0.2950	1.4834	-0.7578	1.6894	2.6347
41	0.0481	-0.2206	-0.0325	-1.3870	1.5110	-0.7819	1.5859	2.5654
42	0.3186	-0.2317	0.0141	-2.5486	1.5261	-0.7915	1.5515	2.5431

k	U	u <sub>d</sub>	v	Z	a <sub>1</sub>	a <sub>2</sub>	b	b <sub>d</sub>
43	-0.4030	0.1855	0.0178	-2.6770	1.5499	-0.7918	1.5527	2.5331
44	0.3276	0.3505	-0.0375	-1.8091	1.5513	-0.7847	1.5325	2.5049
45	-0.2734	0.1422	0.0385	-0.3849	1.5503	-0.7833	1.5346	2.4789
46	0.2790	-0.3188	0.0209	-0.1790	1.5555	-0.7892	1.5342	2.4768
47	0.4902	-0.1032	0.0292	-1.0571	1.5697	-0.8020	1.5228	2.4680
48	0.1164	-0.1442	-0.0234	-0.8066	1.5739	-0.8047	1.4830	2.5289
49	0.1184	-0.3117	-0.0060	-1.3216	1.5771	-0.8032	1.4822	2.5484
50	-0.4517	0.3646	0.0017	-1.0602	1.5768	-0.8024	1.4908	2.5305
51	0.4431	0.0108	-0.0280	0.4112	1.5740	-0.7985	1.4744	2.5051
52	0.1868	0.4446	0.0279	1.9352	1.5709	-0.7941	1.4524	2.4865
k	u	u <sub>d</sub>	v	z	a <sub>1</sub>	a <sub>2</sub>	b	b <sub>d</sub>
53	0.1634	0.2053	-0.0470	1.5224	1.5688	-0.7926	1.3675	2.4370
54	0.3381	-0.0842	-0.0384	1.8175	1.5762	-0.7970	1.3582	2.4250
55	0.2054	0.3849	-0.0137	2.6098	1.5795	-0.7976	1.3582	2.4252
56	-0.1218	0.4339	0.0329	3.7940	1.5846	-0.7998	1.3516	2.4215
57	0.0180	-0.3639	0.0434	5.1381	1.5878	-0.8007	1.3615	2.4341
58	-0.2042	-0.3521	0.0078	5.3793	1.5878	-0.8007	1.3495	2.4386
59	0.1141	0.3680	0.0179	4.8400	1.5827	-0.7944	1.3174	2.4524
60	0.2484	0.3342	-0.0175	4.0189	1.5851	-0.7970	1.3271	2.4515
61	0.4668	-0.2257	-0.0471	2.1406	1.5850	-0.7970	1.2516	2.4305

k	U	u <sub>d</sub>	v	Z	a <sub>1</sub>	a <sub>2</sub>	b	b <sub>d</sub>
62	0.1836	0.2918	0.0144	-0.1392	1.5851	-0.7974	1.2499	2.4295
63	0.2225	-0.1172	0.0169	-1.6081	1.5842	-0.7966	1.2279	2.4398
64	0.2949	0.1242	-0.0491	-3.9344	1.5832	-0.7959	1.2384	2.4547
65	0.4165	0.1248	-0.0249	-6.4279	1.5877	-0.7983	1.2364	2.4489
66	0.2342	-0.0347	-0.0294	-6.3392	1.5885	-0.7987	1.2362	2.4526
67	0.2229	-0.4092	0.0237	-5.1707	1.5910	-0.7981	1.2464	2.4507
68	-0.3598	-0.4784	-0.0028	-3.9536	1.5921	-0.8004	1.2232	2.4666
69	0.2521	-0.1850	-0.0461	-2.8694	1.5949	-0.8035	1.2262	2.4591
70	-0.4445	-0.3991	-0.0270	-2.6806	1.5917	-0.8007	1.2034	2.4221
71	-0.4450	-0.0735	0.0332	-0.7939	1.5926	-0.8013	1.1649	2.3702
72	-0.2777	0.3780	0.0091	1.0711	1.6013	-0.8046	1.1719	2.3749
73	0.2876	0.0905	-0.0472	0.9052	1.6018	-0.8017	1.1930	2.4059
74	0.3594	-0.1709	0.0092	-0.0072	1.6055	-0.8084	1.2182	2.3733
75	-0.3944	-0.4672	0.0060	0.0289	1.5993	-0.8008	1.2246	2.3589
76	0.2963	0.1508	-0.0368	-0.5494	1.5973	-0.7981	1.2300	2.3421
77	0.3428	0.3860	0.0471	-0.6879	1.6040	-0.8047	1.2425	2.3281
78	0.1427	0.0447	0.0189	0.3343	1.5930	-0.7957	1.2335	2.3766
79	-0.4531	0.0017	-0.0011	1.7833	1.5966	-0.7975	1.2349	2.3740
80	0.0819	-0.3912	-0.0337	4.1389	1.5965	-0.7980	1.2223	2.3552
81	-0.0422	0.4315	0.0204	5.3925	1.5927	-0.7922	1.2374	2.3497
82	0.3701	-0.1372	0.0371	4.8552	1.5944	-0.7946	1.2811	2.3538

k	U	u <sub>d</sub>	v	Z	a <sub>1</sub>	a <sub>2</sub>	b	b <sub>d</sub>
83	0.0006	0.4603	0.0459	4.2895	1.5953	-0.7958	1.2548	2.3827
84	-0.0108	-0.4618	0.0430	3.5386	1.5955	-0.7958	1.2621	2.3736
85	0.1587	-0.3237	0.0145	2.1452	1.5963	-0.7967	1.2620	2.3738
86	-0.0219	0.0024	-0.0252	0.0400	1.5977	-0.7978	1.2571	2.3810
87	0.4814	0.0832	0.0256	-1.2715	1.5948	-0.7963	1.2544	2.3769
88	-0.2079	-0.4138	-0.0457	-3.2660	1.5927	-0.7925	1.2485	2.3600
89	-0.1571	-0.4166	-0.0217	-3.0897	1.5905	-0.7924	1.2692	2.3878
90	0.1000	-0.4452	-0.0015	-1.8200	1.5910	-0.7927	1.2900	2.3431
91	0.4780	-0.0697	-0.0263	-1.5375	1.5952	-0.7946	1.2897	2.3317
92	0.2590	-0.1171	-0.0001	-2.0900	1.5949	-0.7960	1.2813	2.3283
93	0.1489	-0.4114	0.0438	-1.7859	1.5958	-0.7978	1.2810	2.3306
94	-0.3246	0.1974	0.0332	-2.1237	1.5951	-0.7964	1.2786	2.3281
95	-0.1634	0.2994	-0.0471	-1.7693	1.5928	-0.7933	1.2554	2.3454
96	-0.4096	0.1533	-0.0066	-0.8148	1.6007	-0.8013	1.2520	2.3495
97	0.0085	0.0421	0.0120	-0.1168	1.5998	-0.8006	1.2273	2.3566
98	0.2403	0.4384	-0.0082	0.6544	1.5986	-0.8000	1.1914	2.3561
99	-0.0273	-0.1917	0.0441	1.4777	1.6039	-0.8033	1.2050	2.3556
100	-0.0265	-0.3181	-0.0234	1.8593	1.6021	-0.8054	1.2039	2.3558



## TÀI LIỆU THAM KHẢO

- [1] Tarun Khana - Foundations Of Neural Networks - Addison, Wesley Publishing Company 1990
- [2] David Zhang - Parallele VLSI Neural System Design - Springer, Verlag Singapore Pte-Ltd 1999
- [3] Simom Haykin - Nerual Networks Acomprehensive Foundation \_ Prentic Hall Inc 1994
- [4] Omid Omidvar, David L.Elliott - Neural Systems for Control - Academic Press, 1997
- [5] Toshio Tsuji, Bing Hong Xu and Makoto Kaneko – ‘Adaptative control and identification using one Neural Network for a class of plants with uncertainties’ - IEEE Transactions on systems, man and Cybernetics - Part A: systems and Humans Vol 28 NO, 4 July 1998
- [6] Khaldoon Bani-Hani, Jam Shid Ghaboussi, Stephen P.schneider – ‘Experimental study of indentification and control of structures using neural network’ - Earthquake Enng Struct Dyn 28,995 - 1018 (1999).