Intelligent control of a DC motor using a self-constructing wavelet neural network

Mohsen Farahani*, Amir Reza Zare Bidaki and Mohammad Enshaeieh

Sama Technical and Vocational Training College, Islamic Azad University, Karaj Branch, Karaj, Iran

(Received 14 November 2013; final version received 15 February 2014)

This paper proposes an intelligent method to control the speed of a DC motor. This controller is a self-constructing wavelet neural network (SCWNN) in which the self-constructing and training algorithms are simultaneously performed. At first, there are no wavelets in the wavelet layer; they are automatically generated in the online control process. In order to increase the convergence speed of the proposed controller, adaptive learning rates (ALRs) updated at each sampling time are used. In the online control process, no identifier is used to approximate the dynamic of the controlled plant, because of the learning ability of the proposed controller. Several simulations are used to demonstrate the effectiveness and adaptiveness of SCWNN.

Keywords: wavelet neural network; self-constructing algorithm; adaptive learning rates; DC motor

1. Introduction

In most applications, DC motors require speed controllers to perform tasks. The nonlinearity impacts in DC motors lead to the poor performance of conventional speed controllers. In order to decrease these impacts, many model-based control methods such as variable-structure control (Canudas, Astrom, & Braun, 1987) and model reference adaptive control (Hung, Gao, & Hung, 1993) have been proposed. However, the accurate modeling and the correct estimation of parameters affect the performance of such controllers. In general, it is difficult to obtain an accurate model of a DC motor and parameter values obtained from estimation methods are only estimated values.

Intelligent techniques such as adaptive proportional-integral-derivative controller (Hsu & Lee, 2011), neural networks (NNs, Madhusudhana Rao & SankerRam, 2009; Nouri, Dhiaoudi, & Braiek, 2008; Peng & Dubay, 2011; Reyes-Reyes, Astorga-Zaragoza, Adam-Medina, & Guerrero-Ramírez, 2010) and fuzzy logic (Rigatos, 2009) are an appropriate alternative to conventional methods. However, there are two basic problems in connection with using NNs. The first one is how to determine the structure of NNs. In general, the structure of NNs in these papers is determined based on the trial-and-error method. No systematic method has been proposed in these papers. The next problem is to use an identifier NN to approximate the dynamic of the controlled system. This method needs heavy computational efforts. In addition, this method requires an offline training which is difficult in practice.

In this paper, a self-constructing wavelet neural network (WNN) controller is used to control the speed of a DC motor. Recently, WNNs, which absorb the advantages such as the multi-resolution of wavelets and the learning of NN, were proposed to guarantee the good convergence and were used to identify and control nonlinear systems. The WNN is suitable for the approximation of unknown nonlinear functions with local nonlinearities and fast variations because of its intrinsic properties of finite support and self-similarity (Lin, 2009; Rigatos, 2009). The main idea is to use wavelet functions and scaling functions as the nonlinear functions required in the neurons. In the proposed controller, the self-constructing mechanism and adaptive learning rates (ALRs) derived by the Lyapunov stability method are employed. To achieve an effective controller and also the optimal size of the proposed controller, the self-constructing and learning algorithms are simultaneously performed in the online mode in the proposed controller. In the self-constructing algorithm, the degree measure method is used to calculate the optimal number of neurons. Accordingly, there is no concern about the structure of WNN. During the online control process, the identification and accurate modeling of system are not necessary, because of learning ability of the proposed controller. The theoretical basis of the proposed controller is completely explained and simulations are used to demonstrate the effectiveness of the proposed controller.

2. DC motor modeling

In a DC motor, the armature circuit consists of an inductor $L_a$ and resistor $R_a$ in series with a counter-electromotive force (CEMF) $E$ (Krause, Wasyczek, & Sudhoff, 1995; Peng & Dubay, 2011). The CEMF is proportional to the
machine speed (Krause, Wasynczuk, & Sudhoff, 1995; Peng & Dubay, 2011).

\[ E = K_E \omega, \]  

(1)

where \( K_E \) is the voltage constant and \( \omega \) is the machine speed. The CEMF \( E \) can also be calculated by the following equation:

\[ E = V_t - R_a \times I_a, \]  

(2)

In a DC machine model, the voltage constant \( K_E \) is proportional to the field current \( I_f \):

\[ K_E = L_{af} \times I_f, \]  

(3)

where \( L_{af} \) is the field-armature mutual inductance. The electromechanical torque developed by the DC machine is proportional to the armature current \( I_a \).

\[ T_e = K_T I_a, \]  

(4)

where \( K_T \) is the torque constant. The torque constant is equal to the voltage constant.

The mechanical part computes the speed of the DC machine from the net torque applied to the rotor. The speed is used to implement the CEMF voltage \( E \) of the armature circuit.

\[ J \frac{d\omega}{dt} = T_e - T_L - B\omega, \]  

(5)

where \( J \) is the motor rotational inertia, \( B \) is the viscous friction coefficient and \( T_L \) is the external load torque. A model of a DC motor driving an inertial load is illustrated in Figure 1. As seen in this figure, the DC motor is fed by an input voltage \( V_t \).

3. Intelligent control system

Figure 2 shows the intelligent method to control the speed of a DC motor. The intelligent control system contains the SCWNN controller, online training mechanism and ALRs. As seen in this figure, the output of the SCWNN is used to control the voltage value of a controlled DC voltage source. The controlled DC voltage source is a DC source whose voltage value can be controlled by an external actuator. In fact, the SCWNN controls the speed of DC motor via increasing and decreasing the voltage applied to the armature circuit. In this case, the speed of a DC motor is changed by the proposed controller according to Equations (1) and (2).

3.1. SCWNN controller

Figure 3 depicts a three-layer WNN which comprises an input layer, a wavelet (the \( j \) layer) and an output layer. The control problem is to design the SCWNN controller so that the power oscillation is quickly damped out. For this purpose, let us define the following control error:

\[ e = \Delta \omega_T - \Delta \omega. \]  

(6)

Here, implementation of the SCWNN is explained in detail. In the wavelet layer, the output of every node is computed as follows:

\[ \Phi_j = \phi(\text{net}_j), \]  

(7)

where \( x_i, m_{ij} \) and \( d_{ij} \) denote the input of WNN, the translation and dilation in the \( j \)th term of the \( i \)th input \( x_i \). A family of wavelets is constructed by translations and dilations performed on a single fixed function named as the mother wavelet. In this paper, the first derivative of a Gaussian function \( \phi(x) = -x \exp(-0.5x^2) \) is chosen as a mother wavelet.
wavelet. In the output layer, the single node calculates the overall output as the summation of all input signals:

$$ u = \sum_{j=1}^{N_v} w_j \Phi_j, \quad (8) $$

In the above equation, $w_j$ denotes the connection weights between the wavelet layer and output layer.

### 3.2. Self-constructing algorithm

The number of neurons existing in a hidden layer depends on the controlled system and is often determined by using the trial-and-error method. If the number of wavelets in the wavelet layer is chosen too large, the computations are heavy and time consuming so that they are not implementable for online practical applications. If the number of the wavelets is chosen too small, the learning process may be not effective enough to achieve the desired control performance.

To solve this problem, an online systematic algorithm is suggested to compute the optimum number of neurons in the hidden layer. The online algorithm is implemented to automatically construct the SCWNN. At first, there are no neurons or wavelets in the structure of the proposed controller. They are automatically generated throughout the online control process. In order to achieve this purpose, the degree measure method is used (Rigatos, 2009).

It is so important to decide when a new wavelet should be generated. The partition-based clustering technique is performed in a data set in this study. For every input $x_i$, the firing strength of a wavelet can be selected as the degree of the incoming pattern belonging to the corresponding wavelet. The spatial location nearer to the center of the wavelet belongs to an input $x_i$ with a higher firing strength. Accordingly, the firing strength obtained from Equation (7) is used as the degree measure

$$ F_j = |\Phi_j| \quad \text{for} \; j = 1, 2, \ldots, N_w, \quad (9) $$

Based on the degree measure, the criterion of a new wavelet generation for every new input is represented.

Find the maximum degree $F_{\text{max}}$

$$ F_{\text{max}} = \max F_j, \quad (10) $$

If $F_{\text{max}} \leq \bar{F}$, then a new wavelet will be generated. Where $\bar{F}$ is a pre-specified threshold that must be reduced during the online control process in order to limit the size of the SCWNN. The choice of the threshold value is an important issue. Since $\Phi_j \leq 1$, the threshold must be adjusted in the range $[0,1]$. A low threshold value leads to fewer generated wavelets and vice versa. Hence, the choice of the threshold value will affect the results. In this paper, $\bar{F}$ is selected as $[0.1^{n}, 0.5^{n}]$, where $n_i$ is the number of input variables of WNN (Lin, 2009). The online self-constructing based on the degree measure for the proposed controller can be summarized as follows:

- If $x$ is the first incoming pattern,
  - Generate a new wavelet;
  - With translation $m_j = x$;
  - Dilation $d_j = 1$;
  - Connection weight $w_j$ is randomly selected;

- for next incoming patterns
  - Compute the degree measure,
    - If $F_{\text{max}} \leq \bar{F}$,
      - Generate a new wavelet;
      - With translation $m_j = x$;
      - Dilation $d_j = 1$;
      - Connection weight $w_j$ is randomly selected;
      - $F_{\text{new}} = l_{\text{dec}} \times \bar{F}$;

Where $l_{\text{dec}}$ is a decrease coefficient of the threshold value which is selected in the range $(0,1)$.

### 3.3. Online training mechanism

The training algorithm is defined as the calculation of a gradient vector in which each component is the derivative of a cost function with respect to parameters of WNN controller so that the cost function is minimized. For this purpose, the chain rule is used and the general method is referred as the back error propagation learning rule, since the gradient vector is computed in the direction opposite to the flow of the output of each node. Before explaining the online training mechanism of the WNN controller using the gradient descent, let us define the following cost function:

$$ E = \frac{1}{2} \sum_{j} e_j^2 = \frac{1}{2} (\Delta \omega - \Delta \alpha)^2. \quad (11) $$

Now, the online training mechanism based on back error propagation is explained in detail. In the output layer, the error term to be propagated is computed as follows:

$$ \delta_o = -\frac{\partial E}{\partial u} = \left[ -\frac{\partial E}{\partial e} \frac{\partial e}{\partial u} \right] = \left[ -\frac{\partial E}{\partial e} \frac{\partial e}{\partial \omega} \frac{\partial \omega}{\partial u} \right]. \quad (12) $$

And the weight of the output layer can be updated according to the following equation:

$$ w_j(k+1) = w_j + \Delta w_j = w_j - \eta_o \frac{\partial E}{\partial w_j} $$

$$ = w_j + \left[ -\eta_o \frac{\partial E}{\partial u} \frac{\partial u}{\partial \omega} \frac{\partial \omega}{\partial w_j} \right] = \eta_o \delta_o \Phi_j, \quad (13) $$

where $\eta_o$ and $k$ are the learning rate of the weight and the number of iteration. Since there is one neuron in the output
layer, only the error term must be computed and propagated.

\[
\delta_j = -\frac{\partial E}{\partial \text{net}_j} = \left[ -\frac{\partial E}{\partial u} \frac{\partial u}{\partial \Phi_j} \frac{\partial \Phi_j}{\partial \text{net}_j} \right]
= \delta_o w_j (\text{net}_j - 1) \exp(-0.5 \text{net}_j^2).
\]  
(14)

And the update law of \( m_{ij} \) is

\[
\Delta m_{ij} = -\eta_m \frac{\partial E}{\partial m_{ij}} = \left[ -\eta_m \frac{\partial E}{\partial u} \frac{\partial u}{\partial \Phi_j} \frac{\partial \Phi_j}{\partial \text{net}_j} \frac{\partial \text{net}_j}{\partial m_{ij}} \right]
= -\eta_m \delta_j.
\]  
(15)

And the update law of \( d_{ij} \) is

\[
\Delta d_{ij} = -\eta_d \frac{\partial E}{\partial d_{ij}} = \left[ -\eta_d \frac{\partial E}{\partial u} \frac{\partial u}{\partial \Phi_j} \frac{\partial \Phi_j}{\partial \text{net}_j} \frac{\partial \text{net}_j}{\partial d_{ij}} \right]
= -\eta_d \delta_j \frac{(x_i - m_{ij})}{(d_{ij})^2}.
\]  
(16)

The translation and dilation of the wavelet layer can be updated according to the following equations:

\[
m_{ij}(k + 1) = m_{ij}(k) + \Delta m_{ij},
\]  
(17)

\[
d_{ij}(k + 1) = d_{ij}(k) + \Delta d_{ij},
\]  
(18)

where \( \eta_m \) and \( \eta_d \) are the learning rates of the translation and dilation of the wavelet layer.

The sensitivity of the system \( \partial \omega/\partial u \) cannot be exactly calculated, because of the nonlinear dynamic of the power system. Hence, an identifier should be used to compute the sensitivity of the power system. This method needs heavy computational effort. To overcome this problem and to improve the online training of the parameters, an approximation law is used as follows (Wai & Chang, 2002; Farahani, 2013a, 2013b):

\[
\delta_o \approx e + e(1 - z^{-1}).
\]  
(19)

### 3.4. Convergence analyses of WNN via ARLs

Analysis of the convergence of the WNN is presented in this section. A selection of the proper learning rate has a direct effect on the convergence of the proposed controller. Accordingly, the learning rate is an important factor in determining the convergence of the WNN trained through the back error propagation method. Therefore, it is important to obtain the optimal learning rate (Farahani, 2013a). However, proper learning rates are calculated with difficulty in the back error propagation method, because suitable learning rates are usually obtained by the trial-and-error method. Accordingly, the ARLs, which can be rapidly computed according to the change of the system, have attracted researchers’ attention (Farahani, 2013a). In this paper, the ARLs are used to better train the proposed controller.

Consider the following discrete Lyapunov function (Farahani, 2013a):

\[
V(k) = E(k) = \frac{1}{2} (e(k))^2.
\]  
(20)

The change in the Lyapunov function is as follows:

\[
\Delta V(k) = V(k + 1) - V(k) = \frac{1}{2} [e^2(k + 1) - e^2(k)].
\]  
(21)

The change in the error is as follows (Lin, 2009):

\[
\Delta e(k) = e(k + 1) - e(k) \approx \left[ \frac{\partial e(k)}{\partial W_i} \right]^T \Delta W_i,
\]  
(22)

where \( W_i \) is an arbitrary component of the vector \( W \) of the WNN and is equal to

\[
W = [w, m, d]^T.
\]  
(23)

And the corresponding change of \( W_i \) is defined by \( \Delta W_i \):

\[
\Delta W_i = \eta_W \delta_o \frac{\partial u}{\partial W_i},
\]  
(24)

where \( \eta_W \) is the learning rate corresponding to the vector component \( W_i \).

THEOREM 1 Let \( \eta_W = \text{diag}[\eta_1, \eta_2, \eta_3] = \text{diag}[\eta_w, \eta_m, \eta_d] \) be the learning rates for the parameters of the WNN and define \( C_{\text{max}} \) as

\[
C_{\text{max}} = [C^1_{\text{max}}, C^2_{\text{max}}, C^3_{\text{max}}]^T
= \left[ \max \| \frac{\partial u}{\partial w} \|, \max \| \frac{\partial u}{\partial m} \|, \max \| \frac{\partial u}{\partial d} \| \right]^T,
\]  
(25)

where \( \| \cdot \| \) denotes the Euclidean norm. Then, the convergence will be guaranteed if \( \eta_W \) are chosen to satisfy

\[
\eta_W = \lambda / (C_{\text{max}}^i)^2, \quad i = 1, 2 \text{ and } 3,
\]  

in which \( \lambda \) is a positive constant gain.

Proof See Appendix 1. ■

THEOREM 2 Let \( \eta_w \) be the learning rate for weights \( w \). The convergence of WNN will then be guaranteed if the following learning rate is used:

\[
\eta_w = \frac{\lambda}{N_w},
\]  
(26)

where \( N_w \) is the neurons number of the wavelet layer.

Proof See Appendix 2. ■

In order to prove Theorem 3, the following lemmas are used.
Table 1. Parameters used for the model of a DC motor.

| Parameter | Value |
|-----------|-------|
| $R_s$ (Ω) | 0.4832 |
| $L_a$ (H) | 0.0068 |
| $R_f$ (Ω) | 84.91 |
| $L_f$ (H) | 13.39 |
| $L_{af}$ (H) | 0.7096 |
| $\bar{J}$ (kg m²) | 0.2053 |
| $\bar{B}$ (N m s) | 0.0070 |

Figure 4. Simulated results of the WNN controller with fixed learning rates: (a) learning rates set at 0.01 in Case 1; (b) learning rates set at 0.01 in Case 2; solid (rotor position) and dashed (reference signal).

**Lemma 1** Let $f(t) = t \exp(-t^2)$. Then, $|f(t)| < 1 \forall t \in \mathbb{R}$.

**Lemma 2** Let $g(t) = t^2 \exp(-t^2)$. Then, $|g(t)| < 1 \forall t \in \mathbb{R}$.

**Theorem 3** Let $\eta_m$ and $\eta_d$ be the learning rates of the translation and dilation, respectively. The convergence of WNN will then be guaranteed if the following equations are used:

$$
\eta_m = \eta_w \left[ \frac{|d|_{\min}}{2|w|_{\max} \exp(-0.5)} \right]^2, \hspace{1cm} (27)
$$

$$
\eta_d = \eta_w \left[ \frac{|d|_{\min}}{2|w|_{\max} \exp(0.5)} \right]^2, \hspace{1cm} (28)
$$

where $N_i$, $|w|_{\max}$ and $|d|_{\min}$ are the input number of the WNN, the maximum of the absolute value of weights vector

Figure 5. Simulated results of the intelligent control system: (a) tracking response in Case 1; (b) performance of SCWNN in Case 1; (c) tracking response in Case 2 and (d) performance of SCWNN in Case 2.
and the minimum of the absolute value of dilations vector, respectively.

Proof  See Appendix 3.

4. Simulation results
In order to demonstrate the performance of the proposed controller, the computer simulations, including parameter variations and external load disturbance (\( T_L \)), are performed for the following cases.

Case 1 \( J = \bar{J}, B = \bar{B}, T_L = 10 \) N m, reference signal: \( 10 + 5 \sin(0.33t) \).

Case 2 \( J = 3\bar{J}, B = 3\bar{B}, T_L = \) from 10 to 50 N m at \( t = 50 \) s, reference signal: \( 50 + 5 \sin(0.33t), \) 0 \( \leq t \leq 20, 50 + 5 \sin t, \) \( t > 20 \).

The parameters used for the DC motor model are given in Table 1. Meanwhile, for all simulations, the threshold value and \( I_{dc} \) are selected as 0.1 and 0.3, respectively.

In the simulation, first to demonstrate the effectiveness of ALRs, the SCWNN with fixed learning rates is considered for comparison. The simulation results with learning rates set at 0.01 for the reference signals in Cases 1 and 2 are shown in Figure 4(a) and 4(b), respectively. These results show that the small learning rates settings at 0.01 lead to slow tracking responses shown in Figure 4(a) and 4(b). So, the choice of values for the learning rates affects the network performance significantly. If small values are selected as the learning rates, the convergence speed will be low. On the other hand, if large values are selected as the learning rates, the system may become unstable. Now, the intelligent control system was described in detail. As described, the control process for Case 2, while there are three wavelets at the end of the online control process for Case 1. Thus, the computation time is considerably reduced.

5. Conclusion
This paper has successfully demonstrated the application of an intelligent control system to control the speed of a DC motor. This intelligent controller is a self-constructing WNN in which the self-constructing and training algorithms both are simultaneously performed in the online mode. At first, an accurate nonlinear model of a DC motor was obtained. Then, the theoretical basis of the proposed intelligent control system was described in detail. As described, adaptive learning rated derived by the Lyapunov method is used to increase the convergence speed of the proposed controller. Meanwhile, there is no wavelet in the proposed controller at the beginning of the online control process; they are automatically generated. Furthermore, simulations were carried out using periodic reference trajectory to test the effectiveness and robustness of the proposed control system. The major merits of this control system are that the strict constrained conditions and prior knowledge of the controlled plant are not necessary in the whole design process, and convergence of the tracking error in the control system can be guaranteed.

References
Canudas, C., Astrom, K. J., & Braun, K. (1987). Adaptive friction compensation in DC-motor drives. IEEE Journal on Robotics and Automation, RA-3(6), 681–685.
Farahani, M. (2013a). A multi-objective power system stabilizer. IEEE Transactions on Power Systems, 28 (3), 2700–2707.
Farahani, M. (2013b). Intelligent control of SVC using wavelet network to enhance transient stability. Engineering Applications of Artificial Intelligence, 26 (1), 273–280.
Hsu, C.-F., & Lee, B-K. (2011). FPGA-based adaptive PID control of a DC motor driver via sliding-mode approach. Expert Systems with Applications, 38 (9), 11866–11872.
Hung, J. Y., Gao, W., & Hung, J. C. (1993). Variable structure control: A survey. IEEE Transactions on Industrial Electronics, 40 (1), 2–22.
Krause, P. C., Waszynzuk, O., & Sudhoff, S. D. (1995). Analysis of electric machinery (pp. 89–92). New York, NY: IEEE Press.
Lin, C.-J. (2009). Nonlinear systems control using self-constructing wavelet networks. Applied Soft Computing, 9 (1), 71–79.
Madhusudhana Rao, G., & SankerRam, B.V. (2009). A neural network based speed control for DC motor. International Journal of Recent Trends in Engineering, 2 (6), 121–124.
Nouri, K., Dhaouadi, R., & Briek, N. B. (2008). Adaptive control of a nonlinear dc motor drive using recurrent neural networks. Applied Soft Computing, 8, 371–382.
Peng, J., & Dubay, R. (2011). Identification and adaptive neural network control of a DC motor system with dead-zone characteristics. ISA Transactions, 50 (4), 588–598.
Reyes-Reyes, J., Astorga-Zaragoza, C-M., Adam-Medina, M., & Guerrero-Ramirez, G-V. (2010). Bounded neuro-control position regulation for a geared DC motor. Engineering Applications of Artificial Intelligence, 23 (8), 1398–1407.
Rigatos, G. G. (2009). Adaptive fuzzy control of DC motors using state and output feedback. Electric Power Systems Research, 79 (11), 1579–1592.
Wai, R-J., & Chang, J-M. (2002). Intelligent control of induction servo motor drive via wavelet neural network. Electric Power Systems Research, 61, 67–76.

Appendix 1. The proof of Theorem 1
From (20), \( V(k) > 0 \). Using Equation (22), we can obtain
\[
e(k+1) = e(k) + \left[ \frac{\partial e(k)}{\partial W_i} \right]^T \Delta W_i,
\]
\[
e(k+1) = e(k) - \left[ \frac{\partial e(k)}{\partial W_i} \right]^T \eta \delta \frac{\partial u}{\partial W_i}.
\]
\[ \| e(k + 1) \| = \| e(k) \left[ 1 - \eta w \left( \frac{\delta_0}{e(k)} \right)^2 \left( \frac{\partial u}{\partial W_i} \right)^T \frac{\partial u}{\partial W_j} \right] \| \]

\[ = \| e(k) \left[ 1 - \eta w \left( \frac{\delta_0}{e(k)} \right)^2 \left( \frac{\partial u}{\partial W_i} \right)^T \frac{\partial u}{\partial W_j} \right] \| \]

\[ = \| e(k) \| \gamma, \]

where

\[ \gamma = \left[ 1 - \eta w \left( \frac{\delta_0}{e(k)} \right)^2 \left( \frac{\partial u}{\partial W_i} \right)^T \frac{\partial u}{\partial W_j} \right] \]

\[ = \left[ 1 - \eta w \left( \frac{\delta_0}{e(k)} \right)^2 \left( C_{\text{max}}^l \right)^2 \right]. \quad (A1) \]

From Equation (A1) and using Equation (13), if \( \| 1 - \eta w (\delta_0/e(k))^2 C_{\text{max}}^l \| < 1 \) is satisfied, then the convergence of the WNN can be guaranteed. Therefore, we can calculate \( \eta w = \lambda / C_{\text{max}}^l, l = 1, 2 \) and 3. This completes the proof of Theorem 1.

**Appendix 2. The proof of Theorem 2**

\( C^1 = \partial u/\partial w = \Phi \), where \( \Phi = [\Phi_1, \Phi_2, \ldots, \Phi_N]^T \) is the output vector of the wavelet layer of the WNN. Then, because we have \( \Phi_j \leq 1 \) for all \( j \), \( \| C^1(k) \| \leq \sqrt{N_w} \). So, we can calculate \( \eta^w = \lambda / N_w \). This completes the proof of Theorem 2.

**Appendix 3. The proof of Theorem 3**

Since

\[ C^2 = \frac{\partial u}{\partial m_{ij}} = w_j \left( \frac{\partial \Phi_j}{\partial m_{ij}} \right) \]

\[ \leq |w_j| \left\{ \max \left[ \frac{\partial \Phi_j}{\partial m_{ij}} \right] \right\} \]

\[ = |w_j| \left\{ \max \left[ \frac{\partial \Phi_j}{\partial m_{ij}} \right] \left[ \frac{\partial \Phi_j}{\partial \text{net}_j} \frac{\partial \text{net}_j}{\partial m_{ij}} \right] \right\} \]

\[ \leq |w_j| \left\{ \max \left( \frac{\partial \Phi_j}{\partial \text{net}_j} \frac{\partial \text{net}_j}{\partial m_{ij}} \right) \right\} \]

\[ \leq |w_j| \left\{ \max \left( \frac{2 \exp(-0.5)}{d_{ij}} \right) \right\} \]

\[ \exp \left\{ \left[ \frac{1}{2} \left( \frac{x_i - m_{ij}}{d_{ij}} \right)^2 - \frac{1}{2} \right] \right\} \]

Thus

\[ \| C^2 \| \leq \sqrt{N_w} |w_j| \max \left( \frac{2 \exp(0.5)}{d_{ij}} \right). \]

Then, we have

\[ \eta_4 = \eta w \left( \frac{|d_{ij}|_{\text{min}}}{2|w_j|_{\text{max}} \exp(0.5)} \right)^2. \]

This completes the proof of Theorem 3.