JOURNAL OF ARTIFICIAL NEURAL NETWORKS, 2(1&2), 81–95 (1995)

Preorganized Neural Networks: Error Back-Propagation Learning of Manipulator Dynamics

Toshio Tsuji
Hiroshima University

Koji Ito
Toyohashi University of Technology

In estimation of some mapping by neural networks, a part of nonlinear functions included in the mapping to be learned is often known beforehand. For example, the equation of motion of the manipulator includes particular nonlinear functions such as sinusoidal functions and multiplication. The present article discusses the method used to embed known nonlinear functions into the error backpropagation neural network to utilize the knowledge in terms of the mapping to be learned. The network proposed is able to learn the known part by using the preorganized layer and the unknown part by using the hidden layer separately.

Then the network is applied to the learning of the inverse dynamics of the direct-drive manipulator. When the preorganized layer is prepared corresponding to the equation of motion, the experimental results show that the network can improve the learning speed and the generalization ability and also can acquire the internal representation.

preorganized neural network, error backpropagation, robot manipulator, inverse dynamics

This work was supported partially by Science Research of the Ministry of Education (034452188) of Japan and the Tateishi Science and Technology Foundation.

Correspondence and requests for reprints should be sent to Toshio Tsuji, Computer Science and Systems Engineering, Faculty of Engineering, Hiroshima University, 1-4-1 Kagamiyama, Higashi-Hiroshima, 739, Japan.

1 INTRODUCTION

The dynamics of a multiarticular manipulator shows a high degree of nonlinearity, such as Coriolis and centrifugal forces, gravity force, and the change of inertia characteristics by the posture. Therefore, it has been general to adopt the method that computes the joint torques from the equation of motion of the manipulator and the desired trajectory. If the motion of the equation can be identified exactly, this computed torque method makes it possible to control the manipulator with high precision. In many industrial manipulators, however, there are often cases when the parameter values such as the position of center of mass and the inertia moment are not clear. Furthermore, there are unknown factors such as the joint friction and the structural gravity compensation (e.g., spring and counterbalanced), which in many cases are not described by the usual equation of motion.

To address the problem of manipulator control in uncertain environments, new approaches using neural networks have been applied. Albus [1] presented the cerebellar model articulation controller, which is essentially an adaptive, distributed table lookup method. Jordan and Rumelhart [2] proposed a recurrent backpropagation method. Kawato et al. [3] have implemented a feedback-error learning neural network that can solve the inverse dynamics problem. These methods, however, do not utilize the knowledge on the controlled objects, that is, the characteristics of the equation of motion.

In estimation of some mapping by neural networks, a part of nonlinear functions included in the mapping to be learned is often known beforehand. For example, as well known, the equation of motion of the manipulator includes particular nonlinear functions such as sinusoidal functions and multiplication. Therefore, in this article, considerations are made to embed the known functions into a network as preorganized knowledge. It is then expected that by structurally preparing preorganized knowledge, the learning speed can be accelerated and the generalization ability can be increased.

2 PREORGANIZED NEURAL NETWORK

Figure 1 shows the neural network proposed herein. The network consists of the input layer, the preorganized layer, the hidden layer, and the output layer. Under the evaluation function E that minimizes the square sum of the error between the target signal t_i and the output signal y_i of the network, the weighting coefficient w_{ij} among units is modified according to the following equations.

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial y_i} \frac{\partial y_i}{\partial x_i} y_j \tag{1}$$

$$\frac{\partial E}{\partial y_i} = \begin{cases} y_i - t_i & \text{(output layer)} \\ \sum_k \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial x_k} w_{ki} & \text{(other layers)} \end{cases} \tag{2}$$

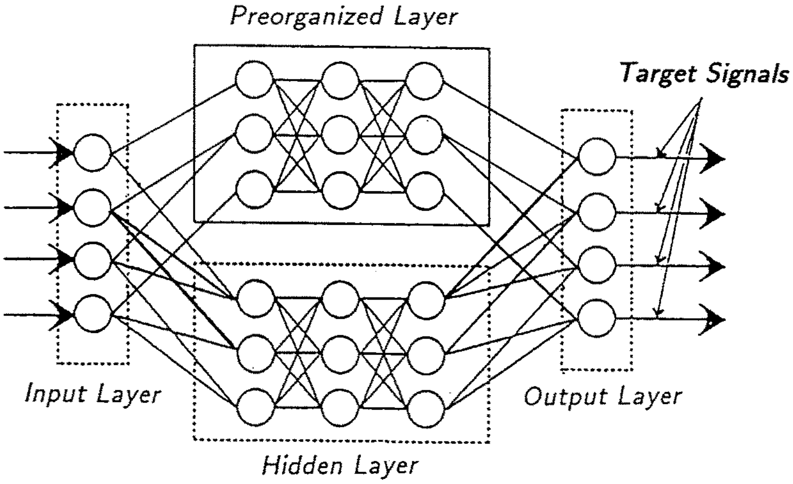


Figure 1. Preorganized neural network.

where x_i denotes the input to unit i and η denotes a learning rate. The output function of the units in the hidden layer is a sigmoid function,

$$f_i = 1/(1 + e^{-x}). \tag{3}$$

For the output function of the units in the preorganized layer, a known function

$$f_i(x) = \phi_i(x) \tag{4}$$

is used. In the following, the unit in which $\phi_i(x)$ is used as the output function is called the *preorganized characteristic unit.*

From the error backpropagation learning rule [Equations (1) and (2)], the output function of each unit is not restricted to the sigmoid function. If the function has the first derivative, it is possible to use it as the output function. So we assume that $\phi_i(x)$ is the first differentiable nonlinear function and is included in the mapping to be learned. Then we can apply the error backpropagation learning rule to the preorganized layer as well as the hidden layer. When an adequate learning rate is given, the weighting coefficient w_{ij} converges a value that minimizes the square sum of the error between the target signal t_i and the output signal y_i of the network.

Furthermore, if it is required to embed a nondifferentiable function into the network, the function should be learned in advance by using the error backpropagation–type neural network (hereinafter called a *subnet*), and then the subnet might be embedded into the preorganized network. When the overall network is learned, the weighting coefficients within the subnet should be fixed during learning; that is, the subnet is regarded as a preorganized unit.

This network is distinguished by the preorganized layer in which nonlinear functions known in advance are embedded. Then, the known parts in the mapping are learned by the preorganized layer and the unknown parts are learned by the hidden layer. So, it is expected that the network can learn more accurately and rapidly than the conventional network that consists of only the sigmoid functions (hereinafter called a *sigmoid-type network*).

3 LEARNING ABILITY OF PREORGANIZED NEURAL NETWORK

To confirm the learning ability of the preorganized neural network, simulation experiments were performed. The mapping to be learned is as follows.

$$t = 2\sin(x_1 + x_2) - 2\sin(x_1 - 2x_2) + \cos(2x_1 - x_2) + 2\cos(-x_1 - 2x_2) - (-x_1 + 2x_2)^2 + (-2x_1 + 2x_2)^2, \quad (5)$$

which is a system with two inputs and one output. The target signals are nine sets that consist of every combination of $x_1, x_2 = 0, \pm 1.0$. The delta-bar-delta rule [4] was used to choose the learning rate η. For the weighting coefficient, we used 10 sets of initial values that consist of uniform random numbers such that $|w_{ij}| < 1.0$. The output function of the unit in the hidden layer is a sigmoid function. The hidden layer consists of 10 units $\times$ 2 layers or 10 units $\times$ 1 layer. The preorganized layer is prepared in four cases as follows:

1. The output function of each unit is $\sin x$ or $\cos x$.
2. The output function of each unit is $\cos x$ or x^2.
3. The output function of each unit is $\sin x$, $\cos x$, or x^2.
4. The preorganized layer is not included in the network.

Note that nonlinear functions involved in the mapping to be learned [Equation (5)] but not prepared in the preorganized layer must be learned in the hidden layer.

Table 1 shows the learning results of the network. The mean iteration number (MIN) in the table represents the average of the learning iterations over 10 sets of initial weighting coefficients when the square sum E of the error is less than 10^{-7}. The table shows that our preorganized neural network converges more rapidly than the sigmoid-type network. The network converges more rapidly when more preorganized units are prepared.

The success rate in Table 1 shows the percentage where the learning has converged within 10^5 iterations. When the hidden layer is one, the sigmoid-type network cannot learn the given nonlinear mapping. From this it is known that the learning ability is improved by adding the preorganized units. In other words, the preorganized units can protect the network from being caught in local minima.

Figure 2 shows the interpolation ability of the network. Figure 2a is the sigmoid-type network (10 units $\times$ 2 layers); Figure 2b is the preorganized network (the case of Table 1C). The dashed line represents the nonlinear mapping to be learned, and the solid line shows the output mapping of the network after learning. In Figure 2a, the output of the network coincides with the mapping to be learned at only x_2

Table 1. Learning Ability of Preorganized Neural Networks

Type	Preorganized Layer	Hidden Layer	Success Rate (%)	M.I.N. ($E<10^{-7}$)
A	$\sin x, \cos x$ (2+2)	(10×1)	60	1078
	$\sin x, \cos x$ (2+2)	(10×2)	100	411
	$\sin x, \cos x$ (5+5)	(10×1)	100	155
	$\sin x, \cos x$ (5+5)	(10×2)	90	162
B	$x^2, \cos x$ (2+2)	(10×1)	60	1001
	$x^2, \cos x$ (2+2)	(10×2)	100	1174
	$x^2, \cos x$ (5+5)	(10×1)	100	1169
	$x^2, \cos x$ (5+5)	(10×2)	100	1335
C	$\sin x, \cos x, x^2$ (2+2+2)	---	60	239
	$\sin x, \cos x, x^2$ (5+5+5)	---	100	164
D	----	(10×1)	0	--
	----	(10×2)	100	4067

$= (-1, 0, 1)$ which is taught by the target signal. Conversely, the preorganized network in Figure 2b coincides with the target mapping almost everywhere. This also holds for the input variable x_1. Therefore, the preorganized neural network can more accurately interpolate the mapping other than the points given by the target signal.

4 APPLICATION TO THE INVERSE DYNAMICS LEARNING OF A ROBOT MANIPULATOR

The preorganized neural network mentioned earlier was applied to the inverse dynamics learning of a two-joint planar direct-drive (DD) robot.

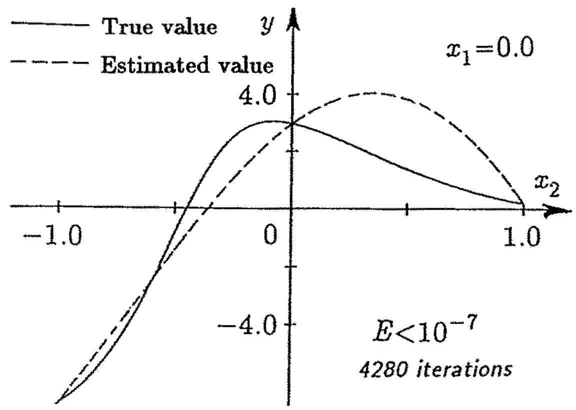
Equation of Motion of a Direct-Drive Robot

Figure 3 shows the DD robot used in the experiment. Since the DD robot has no reduction gears, the robot control is influenced significantly by nonlinear forces such as Coriolis, centrifugal, and inertia. Therefore, exact identification of the robot dynamics becomes important.

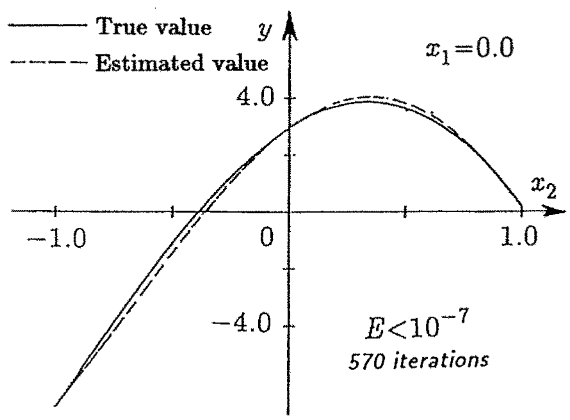
The equation of motion of a two-joint planar manipulator is given by

$$I(\theta)\ddot{\theta} + h(\theta, \dot{\theta}) = \tau, \tag{6}$$

where $\theta \in R^2$ is the joint angle, $\tau \in R^2$ the joint torque, $I(\theta) \in R^{2 \times 2}$ the inertia



(a) Sigmoid type neural network.



(b) Preorganized neural network.

Figure 2. Generalization ability of preorganized neural network.

matrix, and $h(\theta, \dot{\theta}) \in R^2$ the Coriolis and centrifugal forces, which are given as follows.

$$I(\theta) = \begin{bmatrix} p_1 + 2p_3 \cos\theta_2 & p_2 + p_3 \cos\theta_2 \\ p_2 + p_3 \cos\theta_2 & p_2 \end{bmatrix} \tag{7}$$

$$h(\theta, \dot{\theta}) = \begin{bmatrix} -(p_3\dot{\theta}_2^2 + 2p_3\dot{\theta}_1\dot{\theta}_2)\sin\theta_2 \\ p_3\dot{\theta}_1^2 \sin\theta_2 \end{bmatrix}. \tag{8}$$

And p_1, p_2, and p_3 in Equations (7) and (8) are given by

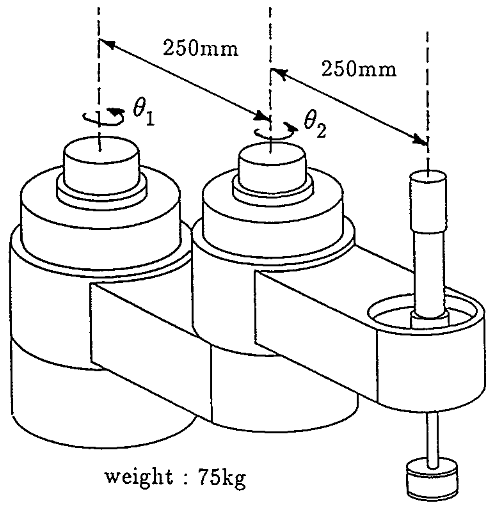


Figure 3. Two-joint planar manipulator.

$$p_1 = I_1 + I_2 + m_1 a_1^2 + m_2(l_1^2 + a_2^2) \tag{9}$$

$$p_2 = I_2 + m_2 a_2^2 \tag{10}$$

$$p_3 = m_2 l_1 a_2, \tag{11}$$

where m_i is the mass of link i, l_i the length of link i, a_i the distance from the joint to the center of mass, and I_i the inertia moment about the center of mass of link i.

In the cases of inverse dynamics learning of the manipulator, a priori knowledge is the equation of motion. Therefore, the sinusoidal function and multiplication commonly involved in the equation of motion are used as the known functions (ϕ_i). Then the preorganized layer is as shown in Figure 4. The correspondence between parameters p_1, p_2, and p_3 in Equations (9–11) and the weighting coefficient w_{ij} from the jth unit to the ith unit is shown in Table 2. The preorganized neural network is formed by the preorganized layer as well as the hidden layer, which consists of the sigmoid-type network, as shown in Figure 1. It is expected that the parameters in the equation of motion might be learned by the preorganized layer, and the unknown factors that are unable to be represented by the equation of motion might be learned by the hidden layer.

Learning of the Inverse Dynamics

Learning of the preorganized neural network was done with the use of 100 pieces of the target signal that had been obtained by manipulating the DD robot directly. The input signals to the network are joint angle θ, angle velocity $\dot{\theta}$, and angle

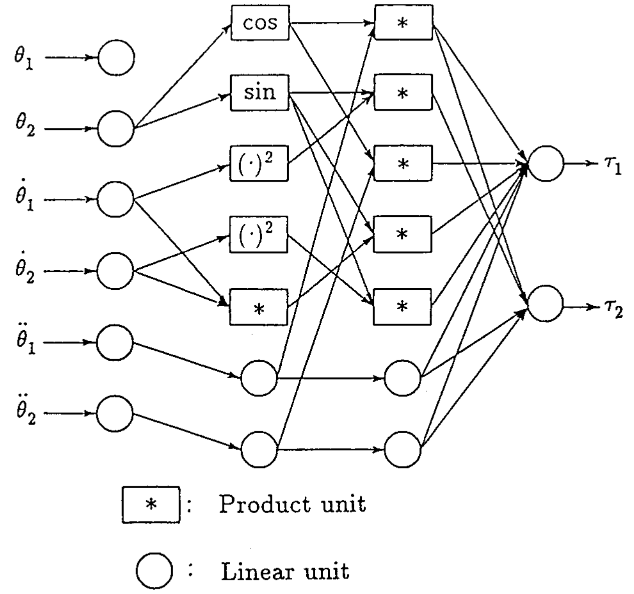


Figure 4. Preorganized layer for inverse dynamics learning.

Table 2. Relationships Between Link Parameters and Weighting Coefficients in Preorganized Layer

w_{ij}	w_{11}	w_{21}	w_{22}	w_{13}	w_{14}	w_{15}	w_{16}	w_{26}	w_{17}	w_{27}
p	$2p_3$	p_3	p_3	p_3	$-2p_3$	$-p_3$	p_1	p_2	p_2	p_2

acceleration $\ddot{\theta}$, and the output signal of the network is joint torque τ (see Figure 4). The sampling period of the signal is 125 ms.

First, to get appropriate initial weighting coefficients w_{ij} of the preorganized layer, only the learning of the preorganized layer shown in Figure 4 was performed. This is nothing but the parameter identification using the least squares method. Because of the noisy data, the estimated parameters would not be true values, but must be nearly true values. Therefore, this also can be regarded as a type of a priori knowledge and helps the network learning. In this learning stage, the initial values of the weighting coefficients w_{ij} between the output layer and the last layer of the preorganized network are set to random numbers and $|w_{ij}| < 1.0$. The other weighting coefficients are fixed at 1.0. The learning was discontinued when the error had shown no decrease. Next, the hidden layer was added to the network, and the overall network was learned by the error backpropagation rule. The initial values of the weighting coefficients of the hidden layer also were random numbers and $|w_{ij}| < 1.0$.

An example of the learning results is shown in Figure 5. The longitudinal axis denotes the error, and the transverse axis denotes the learning iterations. The solid line corresponds to the preorganized neural network, and the dashed line corresponds to the sigmoid-type network (hidden layer: 7 units $\times$ 2 layers). Furthermore, the error is normalized by the square sum of the target signal (torque) as follows:

$$E = \frac{\sum_{i=1}^{100} \sum_{j=1}^{2} (t_j(i) - y_j(i))^2}{\sum_{i=1}^{100} \sum_{j=1}^{2} t_j(i)^2}. \tag{12}$$

The addition of the hidden layer to the preorganized network occurred at 100 iterations, as shown by the solid line in Figure 5.

For 10 sets of initial values of the weighting coefficients, the average learning iteration where the error does not exceed 0.05 was 190.8 (standard deviation: 92.3) in the preorganized network and was 599.8 (standard deviation: 161.7) in the

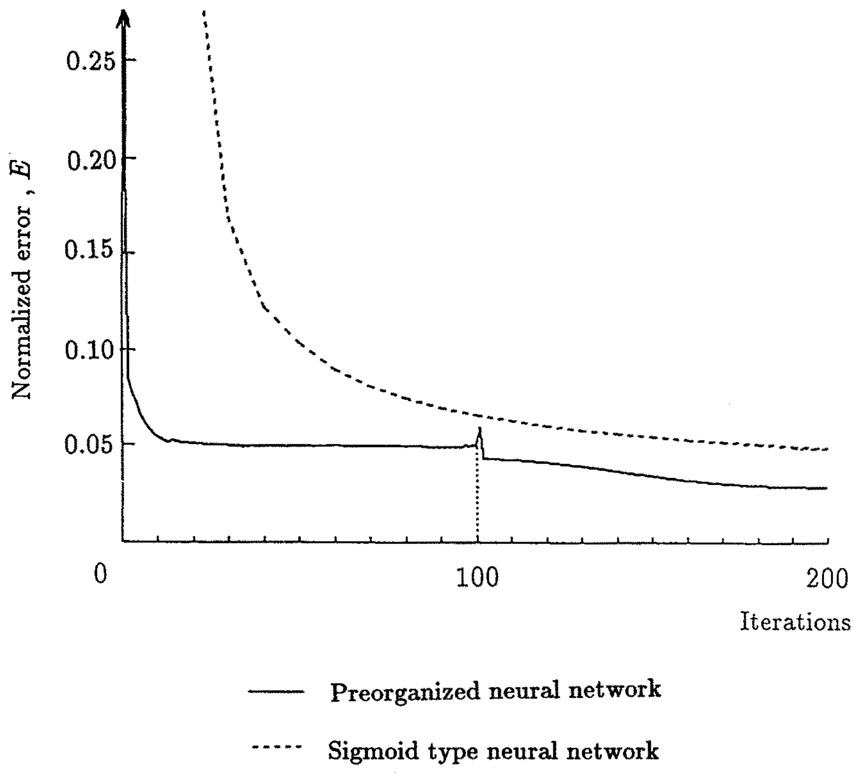


Figure 5. Example of learning history.

Table 3. Results of Inverse Dynamics Learning

	Normalized error, E										
	Teaching patterns	Unlearned patterns									
		1	2	3	4	5	6	7	8	9	Mean
Preorganized N.N.	0.03	0.05	0.08	0.09	0.05	0.10	0.05	0.10	0.06	0.10	0.07
Backpropagation type N.N.	0.04	0.18	0.16	0.34	0.13	0.23	0.21	0.34	0.24	0.36	0.24

sigmoid-type network. This shows that the former converges much more rapidly than the latter. Note that the number of the weights that can be modified is 105 for both networks, and only 10 weights of the preorganized network can be modified during the initial learning.

Generalization Ability

In Section 3, we showed that the interpolation ability of the preorganized network was better than that of the sigmoid-type network. However, data $(\theta, \dot{\theta}, \ddot{\theta}, \tau)$ used as the target signals would include some random noise in the measurement. Then the problem is when the learning should be stopped to investigate the generalization ability of the network.

In this article, we define a generalization index regarded as a modified cross-validation method [5]. Now we define the error signal $e_i(\omega) \in R^L$ for the ith input signal under the set ω of the weighting coefficients as follows:

$$e_i(\omega) = t_i - y_i(\omega), \tag{13}$$

where $t_i \in R^L$ denotes the ith target signal and $y_i(\omega) \in R^L$ denotes the ith output signal of the network under the weighting set ω. We assume that the probability density function of the error signal $e_i(\omega)$ is a channel-independent Gaussian with zero mean and variance σ_j $(j = 1, 2, \ldots, L)$,

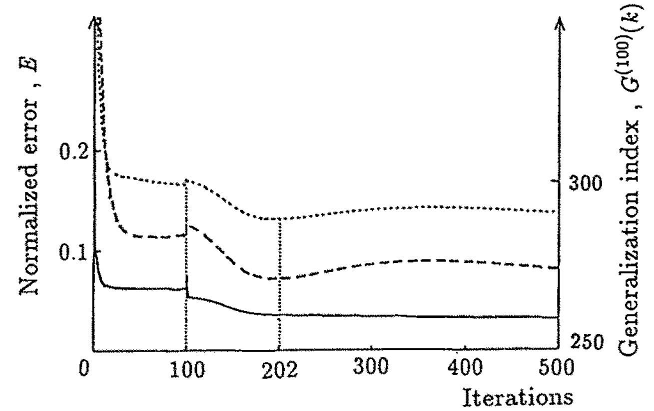
$$p(e_i | \omega, \sigma_1^2, \sigma_2^2, \ldots, \sigma_L^2) = \prod_{j=1}^{L} p(e_{ij} | \omega, \sigma_j^2), \tag{14}$$

$$p(e_{ij} | \omega, \sigma_j^2) = \left[\frac{1}{2\pi\sigma_j^2}\right]^{1/2} \exp\left[-\frac{e_{ij}^2(\omega)}{2\sigma_j^2}\right]. \tag{15}$$

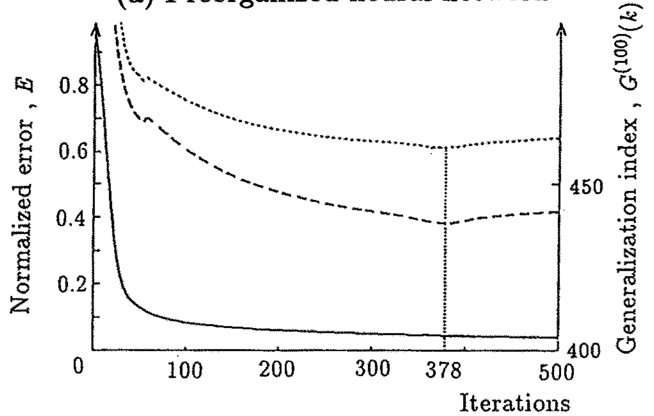
The log-likelihood of the samples $(e_1, e_2, \ldots, e_N)$ is given by

$$l(\omega, \sigma_1^2, \sigma_2^2, \ldots, \sigma_L^2) = -\frac{N}{2} \sum_{j=1}^{L} \log 2\pi\sigma_j^2 - \sum_{i=1}^{N} \frac{e_{ij}^2(\omega)}{2\sigma_j^2}. \tag{16}$$

If we estimate the network parameters ω and σ_j^2 that maximize the log-likelihood,



(a) Preorganized neural network.



—— Nomalized error for teaching patterns
---- Generalization index
...... Nomalized error for unlearned patterns

(b) Sigmoid type neural network.

Figure 6. Generalization index $G^{(N)}(k)$.

it is expected that the generalization ability of the network becomes maximum. In this article, we use the weighting coefficient set ω^k after the kth learning iteration instead of the maximum likelihood estimate of ω. Conversely, the maximum likelihood estimate of the variance can be derived by taking the partial derivative of the log-likelihood with respect to σ_j^2,

$$\hat{\sigma}_j^2 = \frac{1}{N} \sum_{i=1}^{N} e_{ij}^2(\omega^k). \tag{17}$$

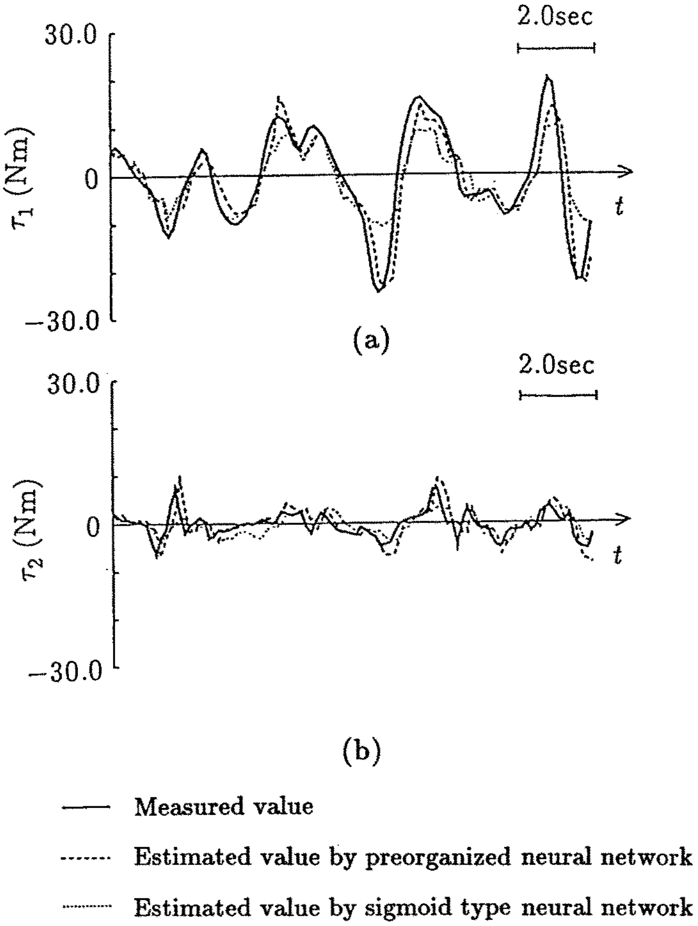


Figure 7. Estimated joint torques for unlearned patterns.

As a result, we can define a generalization index $G^{(N)}(k)$ for the kth iteration as follows:

$$G^{(N)}(k) = -l(\omega, \tilde{\sigma}_1^2, \tilde{\sigma}_2^2, \ldots, \tilde{\sigma}_L^2) = \frac{N}{2}\sum_{j=1}^{L} \log 2\pi \left[\frac{1}{N}\sum_{i=1}^{N} e_{ij}^2(\omega^k)\right] + \frac{LN}{2}. \tag{18}$$

The learning should be discontinued at the iteration when the estimated index $G^{(N)}(k)$ became minimum. Then it is expected that the generalization ability of the network becomes maximum.

Nine sets of data ($N = 100$) were measured from DD robot control. Then the test data were composed of 100 points randomly chosen from the measured data. The preorganized network was learned by the test data. When the generalization index $G^{(N)}(k)$ became minimum, the learning iteration was stopped. Figure 6 shows the change of $G^{(N)}(k)$ during learning. Figure 6a corresponds to the preorganized

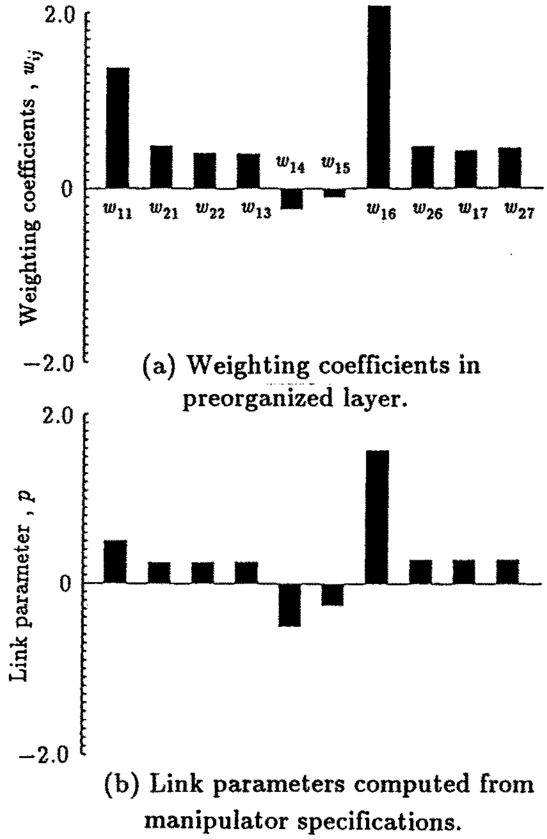


Figure 8. Link parameters learned in neural networks.

network, and Figure 6b corresponds to the sigmoid-type network. The left longitudinal axis denotes the error normalized by the target signals; the right longitudinal axis denotes $G^{(N)}(k)$. The horizontal axis denotes the learning iterations. The solid line denotes the error for the test data used as the target signal. The dotted line denotes the error for unknown data and the broken line denotes $G^{(N)}(k)$. From the figures, it is known that the error for the test data decreases with learning. Conversely, the error for unknown data begins to increase again after some iteration number because the target signal includes some random noise. Furthermore, the learning iterations at which $G^{(N)}(k)$ and the error for unknown data become minimum are almost the same, which means that $G^{(N)}(k)$ represents the generalization ability of the network rightly.

The learning of the network was discontinued at the iteration where the generalization index $G^{(N)}(k)$ became minimum (preorganized network: 202 times; sigmoid-type network: 859 times). The errors of both networks for nine sets of unknown data are shown in Table 3. It is shown that the generalization ability of the preorganized network is better than that of the sigmoid-type network.

Figure 7 shows the joint torques estimated by the preorganized network and the sigmoid-type network, respectively, using the unknown data set $(\theta, \dot{\theta}, \ddot{\theta})$. Figure 7a is the first joint torque, and Figure 7b is the second joint torque. The solid line denotes the measured torques. The broken line denotes the joint torque estimated by the preorganized network, and the dotted line denotes the joint torque estimated by the sigmoid-type network. It is shown that the preorganized network has the ability to estimate more accurate torque than the sigmoid-type network.

Finally, Figure 8a shows the values of 10 types of weighting coefficients from the preorganized layer to the output layer in the preorganized network. Furthermore, Figure 8b shows the corresponding parameter values calculated from the specification of the DD robot (see Table 2). Although the initial values of the weighting coefficients were random, the weighting coefficients after learning approximates closely to the true values. This means that the preorganized network has the ability to acquire the internal representation of the manipulator's dynamics.

5 CONCLUSION

In this article, we discussed the method to embed known nonlinear functions into the error back-propagation neural network to utilize the knowledge in terms of the mapping to be learned. The network proposed is able to learn the known part by using the preorganized layer and the unknown part by using the hidden layer separately. Then the network was applied to the learning of the inverse dynamics of the direct drive manipulator. When the preorganized layer was prepared corresponding to the equation of motion, the experimental results showed that the network could improve the learning speed and the generalization ability and also could acquire the internal representation. Additional research should be directed to clarify the effectiveness of the preorganized network theoretically.

REFERENCES

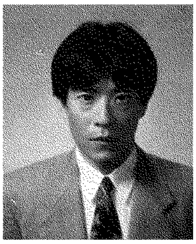
[1] J.S. Albus, "A New Approach to Manipulator Control: The Cerebellar Model Articulation Controller (CMAC)," *Trans. ASME, J. Dynam. Syst., Measur., Contr.*, Vol. 97, pp. 220–227, 1975.

[2] M.J. Jordan and D.E. Rumelhart, "Supervised Learning with a Distal Teacher," *Cognitive Science*, Vol. 16, pp. 307–354, 1992.

[3] M. Kawato, Y. Uno, M. Isobe, and R. Suzuki, "Hierarchical Neural Network Model for Voluntary Movement with Application to Robots," *IEEE Control Systems Mag.*, Vol. 8, pp. 8–16, 1988.

[4] M.A. Jacobs, "Increased Rates of Convergence Through Learning Rate Adaptation," *Neural Networks*, Vol. 1, pp. 295–307, 1988.

[5] N. Tishby, E. Levine, and S. Solla, "Consistent Inference of Probabilities in Layered Networks: Predictions and Generalization," in *Proc. IJCNN*, 1989, pp. 403–409.



Toshio Tsuji was born in 1959. He received the B.E. degree in industrial engineering in 1982, and the M.E. and Doctor of Engineering degrees in systems engineering in 1985 and 1989, all from Hiroshima University. From 1985, he was a Research Associate on the Faculty of Engineering at Hiroshima University, and since 1994, he has been an Associate Professor. He was a Visiting Researcher at the University of Genova, Italy, from 1992 to 1993. His interests include various aspects of motor control in root and human movement, and his current research interests have focused on distributed planning and learning of motor coordination.

Koji Ito was born in 1944. He received the B.S. degree from Nagoya Institute of Technology in 1967, and the M.S. degree in 1969 and the Dr.Eng. degree in 1976 from Nagoya University. From 1970, he was a Research Assistant at the Automatic Control Laboratory, Faculty of Engineering, Nagoya University. From 1979, he was an Associate Professor in the Department of Computer and System Engineering at Hiroshima University. Since 1992, he has been a Professor in the Information and Computer Sciences Department at Toyohashi University of Technology. Since 1993, he has also held an additional post of head, Laboratory for Bio-Mimetic Control Systems at RIKEN, Bio-Mimetic Control Research Center. His main research interests are in the design and control of robotics and prostheses, and computational neural sciences, in particular, biological motor control.