論文 5-2-4

# 신경회로망을 이용한 유도전동기의 속도 센서리스 방식에 대한 비교

鞠 潤 相, 金 倫 鎬, 崔 源 範

# Comparison of Different Schemes for Speed Sensorless Control of Induction Motor Drives by Neural Network

Yoon-Sang Kook, Yoon-Ho Kim, Won-Byum Choi

요 약

일반적으로 시스템 인식과 제어에 이용하는 다층 신경회로망은 기존의 역전파 알고리즘을 이용한다. 그러나 결선강도에 대한 오차의 기울기를 구하는 방법이기 때문에 국부적 최소점에 빠지기 쉽고, 수렴속도가 매우 늦으며 초기결선강도 값들이나 학습계수에 민감하게 반응한다. 이와 같은 단점을 개선하기 위하여 확장된 칼만 필터링 기법을 역전파 알고리즘에 결합하였으나 계산상의 복잡성 때문에 망의 크기가 증가하면 실제 적용할 수 없다. 최근 신경회로망을 선형과 비선형 구간으로 구분하고 칼만 필터링 기법을 도입하여 수렴속도를 빠르게 하고 초기 결선강도 값에 크게 영향을 받지 않도록 개선하였으나, 여전히 은닉층의 선형 오차값을 역전파 알고리즘에 의해서 계산하기 때문에 학습계수에 민감하다는 단점이 있다. 본 논문에서는 위에서 언급한 기존의 신경회로망 알고리즘의 문제점을 개선하기 위하여 은닉층의 목표값을 최적기법에 의하여 직접계산하고 각각의 결선강도 값은 반복최소 자승법으로 온라인 학습하는 알고리즘을 제안하고 이들 신경회로망 알고리즘과 비교하고자 한다. 여러가지 시뮬레이션과 실험을 통하여 제안된 방법이 초기 결선강도에 크게 영향을 받지 않으며, 기존의 학습계수 선정에 따른 문제점을 해결함으로써 신경회로망 모델에 기초한 실시간 제어기 설계에 응용할 수 있도록 하였다. 또한, 유도전동기의 속도추정과 제어에 적용하여 좋은 결과를 보였다.

ABSTRACT

In general, as the model for system identification, a widely used training method for a feedforward multi-layer neural network is back-propagation algorithm, which is an iterative gradient algorithm. Although it has worked successfully, the back-propagation algorithm have several limitations. The long and unpredictable training process is the most troublesome, for example the rate of convergence is seriously affected by the initial weights and the learning rate of the parameters. An extended Kalman filtering to improve this problem is incorporated, however, the computational complexity of this algorithm becomes intractable as the size of the multi-layer neural network increases. Recently, another modified algorithm to be partitioned into linear and nonlinear portions was derived. This technique is faster and more stable than the classical back-propagation algorithm. However, since it uses a modified form of the back-propagation algorithm to minimize the mean-square error, it is not a stable learning algorithm. In this paper, to determine the desired target in the hidden layers a new approach for the on-line learning process of multi-layer neural network using the recursive least squares type algorithm is proposed. This new technique is less sensitive to the initial weights and to the learning parameters.

Comparisons of the three algorithms are made through a system identification problem. The number of iterations required to converge and the mean-squared error between the desired and actual outputs is compared with respect to each method. The theoretical analysis and experimental results are discussed. Also, these algorithms are used to provide a real-time adaptive identification of motor speed for induction motor drives.

**Key words** : BP(back-propagation), EKF(Extended Kalman Filter), RLS(Recursive Least Squares), NN(Neural Network)

## 1. INTRODUCTON

System identification is a process aimed at establishing an adequate input/output relationship for unknown systems, and it is usually the first step taken by control engineers since control theory requires that we understand a system before we try to control it. Since the inception of artificial neural networks(ANN) many researchers have explored a wide variety of applications including identification of nonlinear dynamical system.[1] Some of the advantages of using ANN as the model for system identification are : (i) ability to approximate arbitrary nonlinear functions to any degree of accuracy; (ii) they are adaptive, thus they can take data and learn from it, often capturing subtle relationships; (iii) they can generalize, therefore they can handle corrupt or incomplete data, thus providing a measure of fault tolerance ; and (iv) they are highly parallel, which allows numerous independent operations to be executed simultaneously.[2]

In general, an artificial neural network has a multi-layer network structure. A widely used training method for a feed-forward multi-layer neural network (MNN) is the back-propagation algorithm developed by Rumelhart et al. in 1986, which is an iterative gradient algorithm designed to minimize the mean-square error between the desired output and the actual output for a particular input to the network with respect to the weights.[3] Although it has worked successfully for a wide variety of applications, the standard back-propagation learning algorithm has several limitations. The long and unpredictable training process is the most troublesome, for example the rate of convergence is seriously affected by the initial weights and the learning rate of the parameters. In general, increasing the learning step size can speed up the convergence rate of the learning process, but it may also lead to divergence, paralysis, or continuous instability.[4]

Many researchers have proposed modification of the classical back-propagation algorithm. Wasserman incorporates several heuristics laws in the back-propagation algorithm, but they are difficult to describe systematically.[4] Singhal and Wu incorporated a extended Kalman filering to improve the standard Steepest Descent technique.[5] However, the computational complexity of this algorithm becomes intractable as the size of the MNN increases. Recently, another modified algorithm was derived by Scalero and Tepedelenliogu as an alternative to the back-propagation algorithm. It uses a modified form of the back-propagation algorithm to minimize the mean-square error between the desired output and the actual output with respect to the summation output (inputs to the nonlinearities). However, it is not a stable learning algorithm in practical real-life applications.[5] Thus, a faster and more stable learning algorithm is desired and that is indeed the main purpose of this paper.

In this paper, a new approach for the on-line learning process of multi-layer neural network using the recursive least squares type algorithm is proposed. This method minimizes the global sum of the squared errors between the actual and the desired output values iteratively. This new technique is less sensitive to the initial weights and to the learning parameters. Comparisons of the three algorithms are made through a system identification problem. The number of iterations required to converge and the mean-squared error between the desired and actual outputs is compared with respect to each method. The theoretical analysis and experimental results are discussed. Also, these algorithms are used to provide a real-time adaptive identification of motor speed for induction motor drives.

Those considered in this paper include : gradient-descent back-propagation, the extended Kalman filter neural network algorithm, the recursive lease squares neural network algorithm.

## 2. FLUX ESTIMATOR

Induction motor rotor fluxes are selected to represent the desired and estimated state variable. The following two independent estimators, in the stationary frame, are generally used to derive these rotor fluxes.

### 2.1 Current Model of Rotor Circuit

The rotor flux estimator can be formed if the stator current and the rotor speed are measured in real time. It can be represented as follows.

$$\dot{\hat{\lambda}}^{s}{}_{dqr\_cm} = \left(-\frac{1}{\tau_r} I + \omega_r J\right) \hat{\lambda}^{s}{}_{dqr\_cm} + \frac{L_m}{\tau_r} i^{s}{}_{s} \qquad (1)$$

where the symbol "." denotes the time derivative, "^" denotes the estimated value, "s" denotes the stationary frame.

$\tau_r = L_r / R_r$ : rotor time constant,

$L_r$ : rotor inductance, $\omega_r$ : rotor angular velocity,

$R_r$ : rotor resistance, $i_s^s = \left[i_{ds}^s \quad i_{qs}^s\right]^T$ : stator current,

$\lambda_{dqr\_cm}^s = \left[\lambda_{dr\_cm}^s \quad \lambda_{qr\_cm}^s\right]^T$ : rotor flux,

$L_m$ : mutual inductance,

$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $J = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$.

### 2.2 Voltage Model of Stator Circuit

The voltage model utilizes the stator voltages and currents, but not the rotor velocity. It is commonly used to implement direct field orientation without speed sensors for low cost drive applications. The rotor fluxes in the stationary d-q reference frame can be obtained,

$$\dot{\hat{\lambda}}^{s}{}_{dqr\_vm} = \frac{L_r}{L_m}\{( v^{s}{}_{s} - R_s\, i^{s}{}_{s}) - \sigma L_s\, \dot{i}^{s}{}_{s}\} \qquad (2)$$

where $\sigma = 1 - L_m^2 / L_s L_r$ : leakage inductance, $L_s$, $R_s$ : stator inductance and resistance, $v_s^s = \left[v_{ds}^s \quad v_{qs}^s\right]^T$ : stator voltage, $\lambda_{dqr\_vm} = \left[\lambda_{dr\_vm} \quad \lambda_{qr\_vm}\right]^T$ : rotor flux.

## 3. THE NEWLY PROPOSED SPEED SENSORLESS CONTROL ALGORITHM

### 3.1 Learning algorithm via the Back-Propagation

The back-propagation algorithm can be summarized as follows.[3]

$$w_{ji}^{k-1,k}(t+1) = w_{ji}^{k-1,k}(t) + \Delta w_{ji}^{k-1,k}(t) \qquad (3)$$

where, $w_{ji}^{k-1,k}$ denotes the weight from the $i$th neuron at the $k-1$ layer to the $j$th neuron at the $k$th layer.

$$\Delta w_{ji}^{k-1,k}(t) = \eta \delta_j^k o_i^{k-1} + \alpha \Delta w_{ji}^{k-1,k}(t-1)$$

$$\delta_j = (t_j - o_j) f'(i_j^k) \quad \text{for the output layer}$$

$$\delta_j = f'(i_j^k) \sum_k \delta_k w_{kj} \quad \text{for the hidden layer}$$

where the training coefficient $\eta$ represents the learning rate, the momentum $\alpha$ determines the effect of the past weight changes on the current weight, $i_j^k$ is the total input to the $j$th neuron at the $k$th layer, $\delta_j$ is error signal in the $j$th neuron, $t_j$ is the desired output in the $j$th neuron, $o_j$ denotes the output of the $j$th neuron from the activation function.

where $f(x)$ is called the activation function represented as :

$$f(x) = \frac{1-e^{-x}}{1+e^{-x}}, \quad f'(x) = f(x)(1 - f(x)) \qquad (4)$$

The back-propagation training algorithm is an iterative gradient algorithm designed to minimize the mean square error between the actual output of a feed-forward net and the desired output. Since the learning rate is constant, the larger this constant, the larger the changes in weights. For practical purpose we choose a learning rate that is as large as possible without leading to oscillation. One way to increase the learning rate without leading to oscillation is to include a momentum term.

### 3.2 Learning algorithm via the Extended Kalman Filter

Our learning strategy is based on regarding of a network as an identification problem of constant parameters. Consider the estimation of weights between the $(M-1)$th layer and the $M$th layer. The multi-layer neural network is expressed by the following models with non-linear observation equations :

$$\hat{w}_{ji}(t+1)=\hat{w}_{ji}(t)+\zeta(t) \tag{5}$$

$$\begin{aligned}\hat{y}_j(t+1)&=h(w_{ji}(t+1))+\upsilon(t+1)\\&=o_j^M(t+1)+\upsilon(t+1)\end{aligned} \tag{6}$$

where $\hat{y}$ is the desired state of an output unit, $\{\zeta(t),\upsilon(t)\}$ are mutually independent, zero-mean noises with covariance matrix $Q$ and $R$ regarded as a modeling error. Note that they can be considered pseudo-noises for tuning the gain of the extended Kalman filter. The application of the EKF to (5) and (6) gives the following real-time learning algorithms :

$$w_{ji}(t+1)=w_{ji}(t)+K_{ji}(t)[y_j(t)-o_j^M(t)] \tag{7}$$

where

$$K_{ji}(t)=\frac{P_{ji}(t+1\mid t)H_{ji}(t)^T}{[H_{ji}(t)P_{ji}(t+1\mid t)H_{ji}(t)^t+R]}$$

$$P_{ji}(t+1\mid t)=P_{ji}(t\mid t)+Q$$

$$P_{ji}(t+1\mid t+1)=[I-K_{ji}(t)H_{ji}(t)^T]P_{ji}(t+1\mid t)$$

$$H_{ji}(t)^T=f'(i_j^k)o_i^{k-1}$$

The filtered estimates of $w_{ji}^{k-1,k}$, $k=M-1,\ldots,2$ at $t+1$ are obtained by the following extended Kalman filter :

$$\hat{w}_{ji}^{k-1,k}(t+1)=\hat{w}_{ji}^{k-1,k}(t)+\eta_{ji}^{k-1,k}(t)\delta_j^k o_i^{k-1} \tag{8}$$

where,

$$\eta_{ji}^{k-1,k}(t)=\frac{P_{ji}^{k-1,k}(t+1/t)}{[H_{ji}(t)^T P_{ji}^{k-1,k}(t+1/t)H_{ji}(t)+R]}$$

$$\begin{cases}\delta_j^k=f'(i_j^k)\sum_{l=1}^{N_{k+1}}w_{jl}^{k,k+1}\delta_l^{k+1} & for\ k=M-1,\cdots,2\\ \delta_j^k=f'(i_j^k)(y_j-o_j^k) & for\ k=M\end{cases}$$

with initial conditions $\hat{w}_{ji}^{k-1,k}(0)=\overline{w}_{ji}^{k-1,k}$ and $P_{ji}^{k-1,k}(0\mid 0)=P_{ji}^{k-1,k}(0)$.

As previously mentioned, the classical back-propagation algorithm has several limitations : (i) the rate of convergence is seriously affected by initial weights and the learning rates of the parameters; and (ii) paralysis, divergence, and continuous instability could result if the step size is too large. This algorithm has the featurc that the learning rate is time dependent, whereas the BP algorithm has a constant learning rate. The present method assures faster learning than the BP algorithm and works well even though the initial weights are relatively large.

### 3.3 Learning algorithm via the Recursive Least Square

A nonlinear neural network problem could be partitioned into linear and nonlinear portions. This means that if all the node inputs $x_{j-1}$ and summation outputs $y_{jk}$ were specified, the problem would be reduced to a linear problem, i.e. a system of linear equations that relate the weight vector $w_{jk}$ to the summation outputs and the node inputs.

We first calculate the desired summation output $y_{jk}$ of the nonlinear portion, then update and optimize the weights of the linear part between the desired summation outputs $y_{jk}$ and the inputs. The new algorithm can be summarized by the following steps.

1) Calculation of the desired summation output. The estimates of the desired summation outputs $d_{jk}^*$ of the hidden layers can be obtained by evaluating the gradient of $E$ with respect to the node inputs $x_{jk}^*$. The estimate of the desired summation outputs of the output layer $d_{Lk}^*$ can be calculated from the target

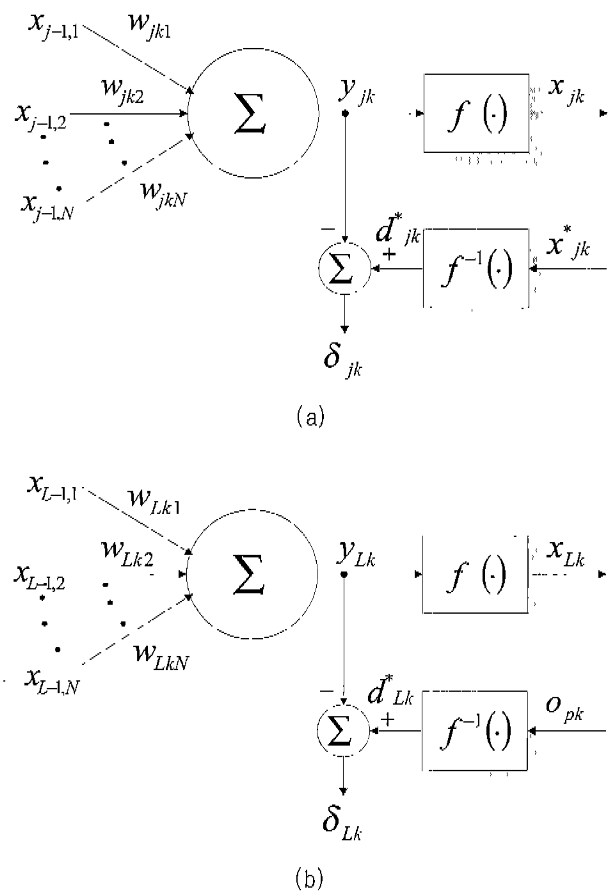



그림 1 은닉층과 출력층에서 신경소자의 구분
Fig. 1 Linear and nonlinear portion of the neurons in the hidden and output layers.
(a) Hidden layer (b) Output layer

output $o_k$ and the inverse of activation function $f^{-1}(\cdot)$ :

$$d_{Lk} = f^{-1}(o_k) \tag{9}$$

$$d_{jk}^* = f^{-1}(x_{jk}^*),\ j = 2,3,\cdots,(L-1) \tag{10}$$

where

$$x_j^* = W_L (W_L^T W_L)^{-1} d_L$$

$$W_L = [w_{L1}, w_{L2}, \cdots, w_{LM}]$$

$$d_L = [d_{L1}, d_{L2}, \cdots, d_{LM}]^T$$

$M$ : the number of output layer node.

2) Iterative learning with RLS algorithm. The recursive least squares method partitions the layers of an NN into a linear set of input-output equations and applies the common RLS algorithm to update the weights in each layer. The application of the RLS algorithm for a weight matrix update gives the following real-time learning algorithms :

$$K_j = P_j x_{j-1} / (\lambda + x^T{}_{j-1} P_j x_{j-1}) \tag{11}$$

$$P_j = (I - K_j x^T{}_{j-1}) P_j / \lambda \tag{12}$$

where $K_j$ is the gain matrix, $P_j$ is covariance matrix, the forgetting factor $\lambda$ can be used to improve the characteristics of the transient response as follows :

$$\lambda(k) = \lambda_0 \lambda(k-1) + (1 - \lambda_0) \tag{13}$$

where $\lambda_0 = 0.98$, $P_j(0|0) = 500I$.

3) Optimal value of the weights.

$$w_{Lk} = w_{Lk} + K_L (d_{Lk} - y_{Lk}) \tag{14}$$

$$w_{jk} = w_{jk} + K_j (d_{jk}^* - y_{jk}) \tag{15}$$

A second method, which is an alternative to the back-propagation algorithm, by minimizing the mean-square error between the desired output and the actual output with respect to the summation outputs $y_{jk}$. Unfortunately, it has an unstable or divergent learning algorithm because the estimates of the summation output $y_{jk}$ for the nonlinear part are not accurate enough.

In this paper, we propose to search for the optimal learning rate parameters. This procedure not only yields more accurate estimates for the desired summation outputs of each neuron, but also saves the trial and error of adjusting the learning rate parameters.

### 3.4 Speed sensorless control strategy

Two independent observers are used to estimate the rotor flux vectors: one based on (1) and the other based on (2). Since (1) does not involve the speed $\omega_r$, this observer generates the desired value of rotor flux, and (2) which does involve $\omega$, may be regarded as a neural model with adjustable weights. The error between the desired rotor flux $\lambda^s_{dqr\_vm}$ given by (1) and the rotor flux $\lambda^s_{dqr\_cm}$ provided by the neural model (2) is used to adjust the weights, in other words the rotor speed $\omega$.

The rotor speed can be derived using the NN. The overall block diagram of speed sensorless control is shown in Fig. 2.

The weights between neurons are tuned so as to minimize the energy function

$$E=\frac{1}{2}\varepsilon^2=\frac{1}{2}\left(\hat{\lambda}_{dqr\_vm}(k)-\hat{\lambda}_{dqr\_cm}(k)\right)^2 \tag{16}$$

The current model can be represented as a neural model.

$$\hat{\lambda}_{dqr\_cm}(k)=x(k)w^T(k) \tag{17}$$

where

$$x(k)=\left[\hat{\lambda}^s_{qr\_cm}(k)\quad \hat{\lambda}^s_{dr\_cm}(k)\quad i^s_{qs}(k)\right]$$
$$=\left[X_1(k)\quad X_2(k)\quad X_3(k)\right]$$
$$w(k)=\left[1-\frac{1}{\tau_r}T_s\quad \hat{\omega}_r T_s\quad \frac{L_m}{\tau_r}T_s\right]$$
$$=\left[\hat{w}_{11}(k)\quad \hat{w}_{12}(k)\quad \hat{w}_{13}(k)\right]^T$$

$T_s$ : sampling time

The new weight, $\hat{w}_{12}(k)$ is therefore given by

$$\hat{w}_{12}(k+1)=\hat{w}_{12}(k)+K_{ji}(k)\left[y_j(k)-\hat{y}_j(k)\right] \tag{18}$$

where $y_j(k)=\lambda^s_{dqr\_vm}(k)$.

The estimated rotor speed $\hat{\omega}_r(k)$ applied by RLS based on NN is computed as follows :

$$\hat{\omega}_r(k+1)=\hat{\omega}_r(k)+K_{ji}(k)\left[y_j(k)-\hat{y}_j(k)\right]/T_s \tag{19}$$

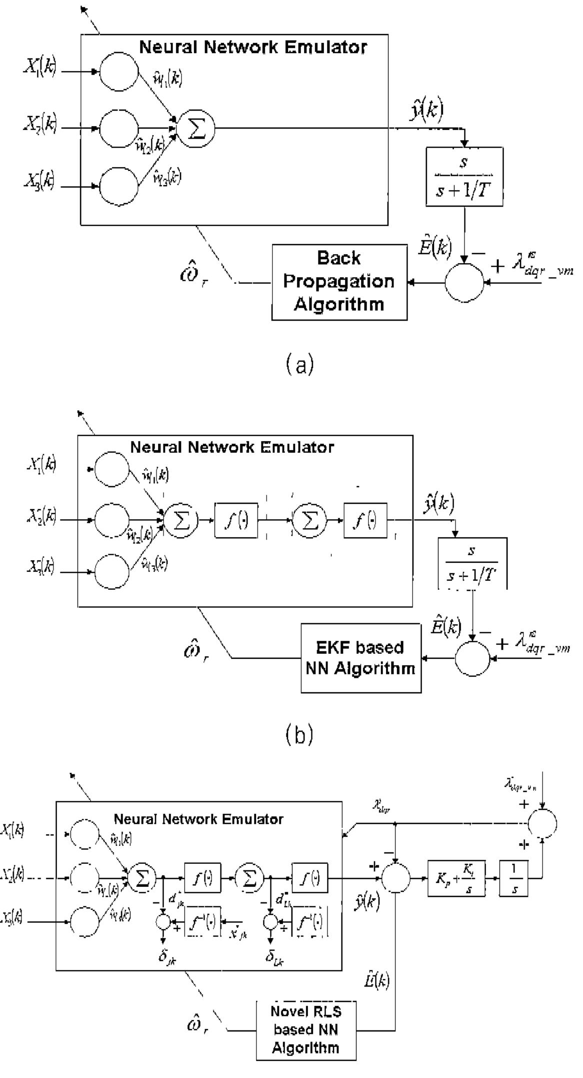



그림 2 $\hat{\omega}_r$ 추정을 위한 신경회로망의 구조

Fig. 2 Structure of NN for $\hat{\omega}_r$ estimation.
(a) Back-Propagation algorithm
(b) Extended KALMAN Filter algorithm
(c) Recursive Least Square algorithm

## 4. SIMULATION RESULTS

A 22kW 4-pole IM is used for the simulation and experiment simultaneously. The proposed sensorless control of IM is shown in Fig. 3. In this paper, a synchronous frame current regulator is used, and reference voltages are calculated from the current regulator. The estimated angular speed is used for the transformation between stationary reference frame and the synchronously rotating reference frame. In PWM techniques, the space vector voltage modulator is used to obtain constant switching frequency. The nominal parameters used for the simulations are given Table 1 as follows :

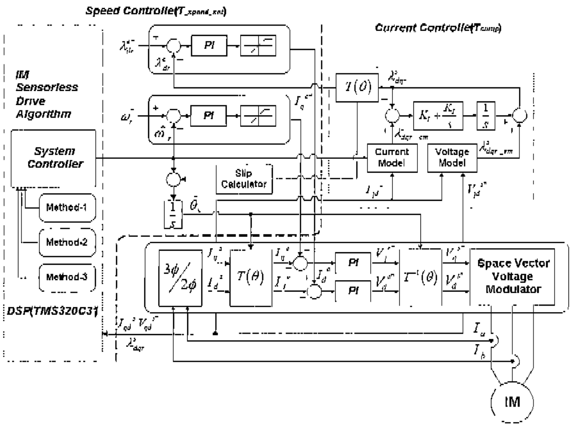



그림 3 전체 제어 알고리즘의 블록선도

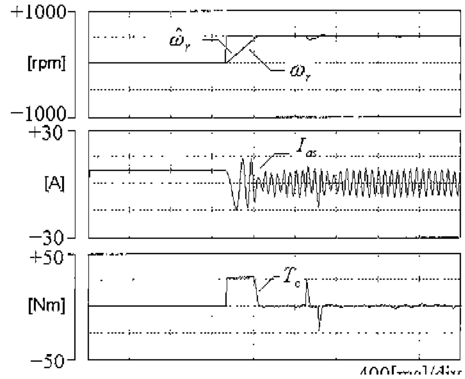
Fig. 3 The block diagram of the overall control algorithm.

표 1 유도전동기의 제정수

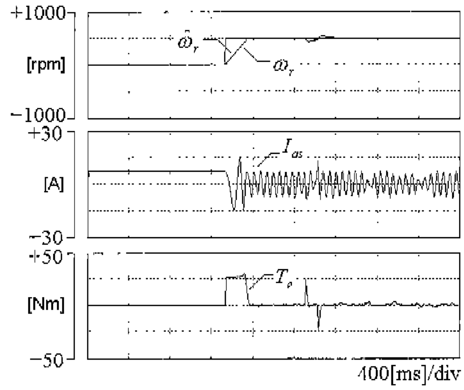
TABLE 1 Induction Motor Parameters.

| | | | |
|---|---|---|---|
| Rated Power | 22kW | Ls | 43.75mH |
| Rated Speed | 2000rpm | Lr | 44.09mH |
| Rated Torque | 120Nm | Lm | 42.1mH |
| Rs | 0.115 | Jm | $0.1618kgm^2$ |
| Rr | 0.0821 | P | 4 |

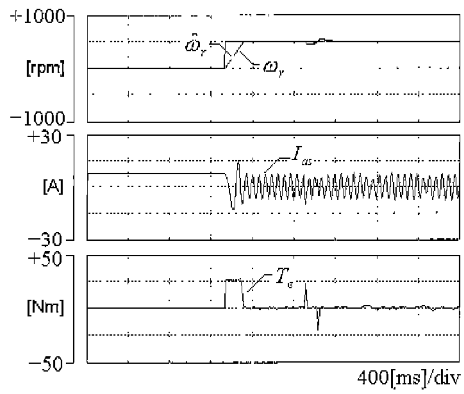
The step response of the speed sensorless algorithm is shown in Fig. 4 when the speed reference is changed from 0[rpm] to 500[rpm]. As shown in Fig. 4, we can know that the speed error of RLS-based NN algorithm is limited by 0.05% of the rating speed. Also, the proposed learning algorithm usually converges in a few iterations and the error is comparable to that of the well-known back-propagation algorithm.




(a)




(b)




(c)

그림 4 부하토크 인가시 속도 추정 특성 비교

Fig. 4 The characteristics comparison of speed estimation when applied to the load torque (0 → +500[rpm], TL=0.5[p.u.]).

(a) [Method-1] BP-based NN algorithm(72,000s)

(b) [Method-2] EKF-based NN algorithm(40,000s)

(c) [Method-3] RLS-based NN algorithm(2,000s)

Fig. 5 shows the comparison of the mean squared error versus the number of iteration for each method. The proposed learning algorithm usually converges in a few iterations at the same mean squared error.

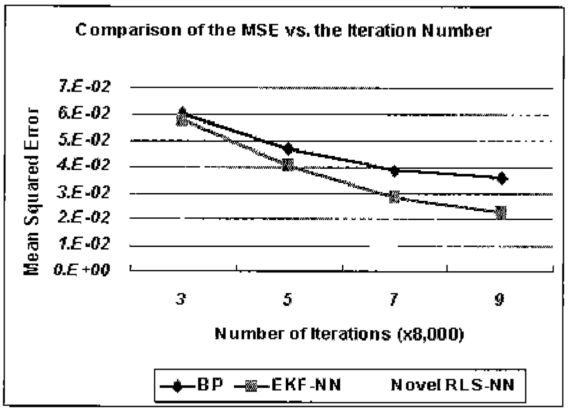



그림 5 평균제곱오차 대 반복회수의 비교

Fig. 5 Comparison of mean-squared error vs. the number of iterations for each NN algorithm.

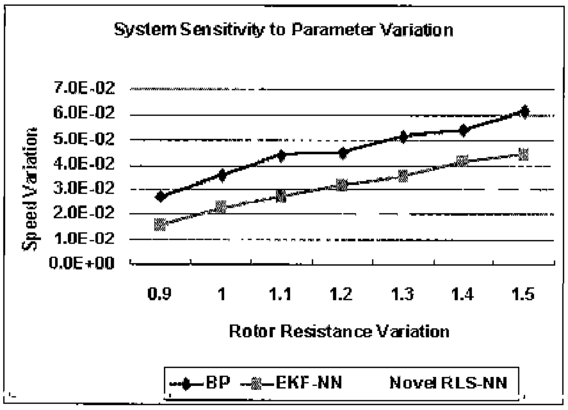



그림 6 파라미터 변동에 대한 시스템 민감도
: 속도변동 대 회전자 저항 변동

Fig. 6 System sensitivity to parameters variation : speed variation vs. rotor resistance variation for each NN algorithm.

Fig. 6 shows the system sensitivity to parameter variation. Fig. 7 shows the comparison of the mean squared error versus the noise sensitivity. We find that this proposed technique of MNN is much less affected by the initial weights and the learning parameters.

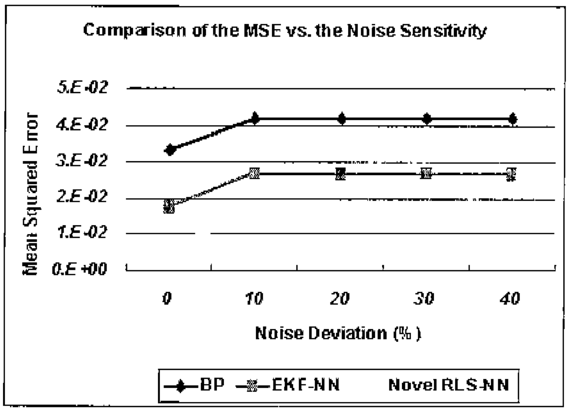



그림 7 평균제곱오차 대 노이즈 민감도의 비교

Fig. 7 Comparison of mean-squared error vs. the noise deviation for each NN algorithm.

## 5. EXPERIMENTAL RESULTS

For the high performance IM drives, the overall IM drive system in Fig. 8 is implemented with a TMS320C31 DSP control board and a PWM IGBT inverter.

For actual load emulation, the DC generator is coupled to the IM. Actual rotor angle and machine speed are measured from an incremental encoder with 4096[ppr] resolution for monitoring. The sampling time of current controller loop is 250[us] and that of the outer voltage regulating loop and speed loop is 2.5[ms]. The control algorithm including the proposed scheme was fully implemented with the software.

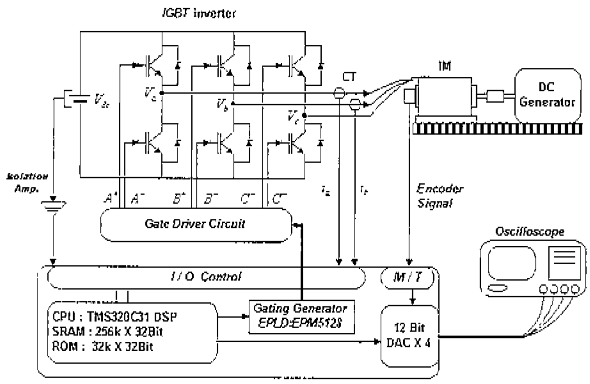
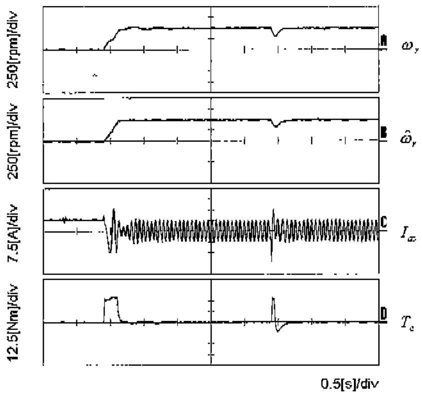
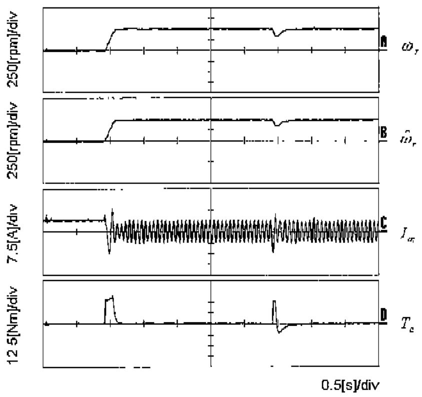



그림 8 전체 유도전동기 구동 시스템

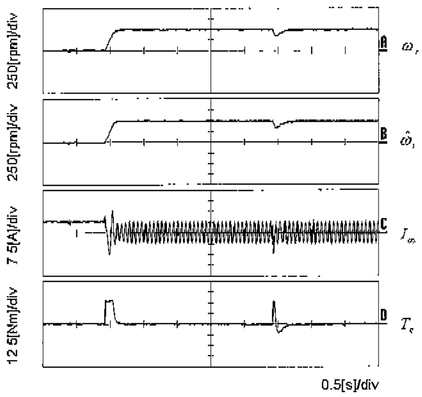
Fig. 8 The overall IM drive system.




(a)




(b)




(c)

그림 9 부하토크 인가시 속도응답 특성 비교

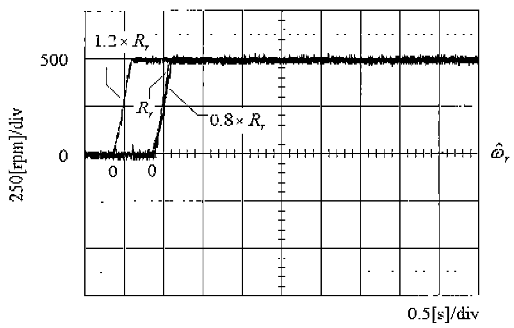
Fig. 9 The experimental waveforms of characteristics comparison of speed response when applied to the load torque(0 → +500[rpm], TL=1[p.u.]).

(a) [Method-1] BP-based NN algorithm(72,000s)

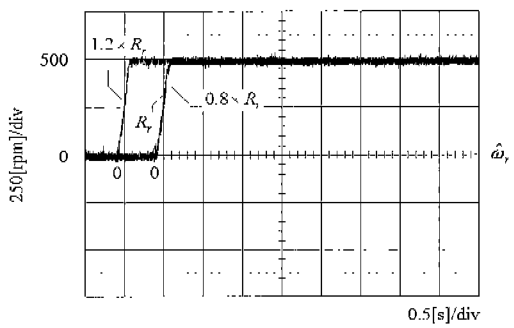
(b) [Method-2] EKF-based NN algorithm(40,000s)

(c) [Method-3] RLS-based NN algorithm(2,000s)

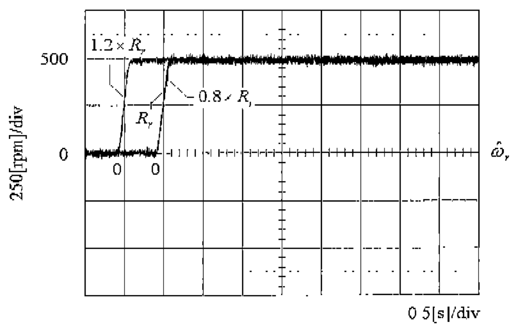
Experiments are conducted to evaluate the performance of the new speed sensor elimination algorithm based on the NN. The step response of the speed sensorless algorithm is shown in Fig. 9 when the speed reference is changed with load torque. It shows that the estimated speed is tracking the real one with good accuracy.




(a)




(b)




(c)

그림 10 회전자 저항의 변동에 따른 속도응답 특성

Fig. 10 The experimental waveforms of speed response characteristics according to the rotor resistance variation(0 → +500[rpm], TL=0).

(a) [Method-1] BP-based NN algorithm(72,000s)

(b) [Method-2] EKF-based NN algorithm(40,000s)

(c) [Method-3] RLS-based NN algorithm(2,000s)

Fig. 10 shows the characteristics of speed response with rotor resistance variation. As shown in those figure, the proposed algorithm works well in spite of the load torque variation and parameter variation.

## 6. 결 론

We have studied learning algorithm for multi-layered feed-forward type neural networks. Neural Network algorithm can be divided into three categories for speed sensorless control of induction motor drives.

1) Back Propagation-based NN algorithm
2) Extended Kalman Filter-based NN algorithm
3) Recursive Least Square-based NN algorithm

Comparisons of the three algorithms are made through a system identification problem. The number of iterations required to converge and the mean-squared error between the desired and actual outputs is compared with respect to each method.

In this paper, a new approach for the on-line learning process of multi-layer neural network using the recursive least squares type algorithm is proposed. This method minimizes the global sum of the squared errors between the actual and the desired output values iteratively. This new technique is less sensitive to the initial weights and to the learning parameters.

The theoretical analysis and experimental results are discussed. The convergence of both the BP and EKF-based NN algorithms depend heavily on the magnitude of the initial weights. If chosen incorrectly, both algorithms will take a long time to converge and may even diverge. The RLS-based NN algorithm is less affected by the initial weights and the learning parameters.

## 참 고 문 헌


[1] Narendra, K. S. and Patthasarathy, Identification and control of dynamical systems using neural networks, IEEE Trans. Neural Networks, Vol. 1, pp. 4-27, 1990.

[2] Keigo Watanabe, Toshio Fukuda and Spyros G. Tzafestas, Learning Algorithms of Layered Neural Networks via Extended Kalman Filters, Int. J. Systems Sci., vol. 22, No. 4, pp. 753-768, 1991.

[3] Rumhart, D. E. and McClelland, J. L., Parallel Distributed Processing, Vol. 1, MIT Press, Cambridge, MA, 1986.

[4] Wasserman, P. D., Neural Computing, Theory and Practice, Van Nostrand Reinhold, NY, 1989.

[5] Robert S. Scalero and Nazif Tepedelenlioglu, A Fast New Algorithm for Training Feedforward Neural Networks, IEEE, Trans. on Signal Processing, vol. 40, No. 1, pp. 202-210, Jan. 1992.


## <저 자 소 개>


**국윤상(鞠潤相)**

1972년 2월 26일생. 1994년 중앙대 전기공학과 졸업. 1996년 중앙대 대학원 전기공학과 졸업(석사). 1999년 동 대학원 졸업(박사). 1999년~현재 ㈜파워테크 연구원.

**김윤호(金倫鎬)**

1945년 5월 23일 생. 1978년 서울대 전기공학과 졸업. 1980년 New York 주립대 졸업(석사). 1984년 Texas A & M 졸업(박사). 현재 중앙대 전기공학과 교수. 당 학회 부회장.

**최원범(崔源範)**

1960년 9월 22일생. 1982년 중앙대 전기공학과 졸업. 1984년 중앙대 대학원 전기공학과 졸업(석사). 1992년 중앙대 대학원 전기공학과 졸업(박사). 1994~1995년 중앙대 전기공학과 강사. 1996~현재 여주대학 자동차과 조교수.