Feed-forward Control of Thermal Power Plants Using Neural Networks

Eki, Yurio
Department of Electrical and Electronic Systems Engineering, Kyushu University : Graduate Student (Omika Works, Hitachi LTD.)

Hirasawa, Kotaro
Department of Electrical and Electronic SystemsEngineering, Kyushu University : Professor

Murata, Junichi
Department of Electrical and Electronic Systems Engineering, Kyushu University : Professor

Hu, Jinglu
Department of Electrical and Electronic Systems Engineering : Research Associate

https://doi.org/10.15017/1498313

出版情報：九州大学大学院システム情報科学紀要. 3 (1), pp.13-21, 1997-12-22. Faculty of Information Science and Electrical Engineering, Kyushu University

九州大学大学院
システム情報科学研究科報告
第3巻 第1号 平成10年3月

Research Reports on Information Science and
Electrical Engineering of Kyushu University
Vol. 3, No. 1, March 1998

Feed-forward Control of Thermal Power Plants Using Neural Networks

Yurio EKI*, Kotaro HIRASAWA**, Junichi MURATA** and Jinglu HU**

(Received December 22, 1997)

Abstract: In thermal power plants, it is an important theme to improve the control performance of main steam pressure and temperature etc. during load up/down. This paper focuses on temperature control that is the most difficult problem due to the non-linearity and long dead times of power plants. Model Reference Adaptive Control (MRAC) is applicable to the feed-forward control of power plants, but there are some problems. The most serious problem is that persistently exciting (PE) condition is not satisfied, and so it is difficult to estimate plant parameters using the well-known recursive least squares method. It is proposed in this paper that Jacobians of the neural networks (NN) are applied to identify the above mentioned plant parameters and control law is obtained by two methods, that is, one is the method to use the Jacobians of the NN plant model which is obtained by off line forward model learning, the other is the method to utilize the Hessian of the cost function. This method is evaluated by a detailed simulator that represents accurately the dynamics of power plants, and usefulness and effectiveness of the proposed method is proved.

Keywords: Control systems, Non-linear control, Thermal power plants, Neural networks

1. Introduction

This paper describes the application of neural networks to the feed-forward control to thermal power plants. In thermal power plants, not only feedback control, but also feed-forward control is necessary to keep main steam temperature, pressure etc. to the set values during load up/down. It is difficult to determine this feed-forward control signal in the complex system such as a thermal plant. We have studied the application of MRAC[1] to the problem mentioned above. But in the process control where the process value changes slowly, it has become clear that PE condition is not satisfied, so it is difficult to estimate the plant parameters by recursive least squares method of MRAC. In place of MRAC, we consider in this paper the application of NN to the control of the thermal power plant. NN has been applied in many industrial fields such as, for example, pattern recognition, robotics etc., but only a few examples have been reported in the process control fields. Application of NN to the power system control involves many problems to be solved. The main problem is that frequent load up/down for training NN is not permissible, so it is difficult to train on line. Two off line NN training schemes can be employed in controller design, however they still have unsolved problems:

(a) Generalized learning[2]. An inverse NN model of the plant is trained and then used as a controller, which does not always give a well-trained good NN controller.

(b) Forward model learning. An inverse of a trained NN model of the plant is derived by a certain method and is employed as a controller, where the inversion is a relatively difficult task. Details of (a) and (b) will be discussed later.

In this paper we develop two methods to obtain the on line control law which are based on the off line forward model learning. One is the method to use the Jacobian of the NN plant model which is obtained by off line forward model learning, the other is the method to utilize the Hessian of the cost function. Temperature control of thermal power plants is discussed in the following sections, which is the most difficult theme for control because of non-linearity and long dead times.

We confirm that the proposed method is very effective by a detailed simulator that represents accurately the dynamics of plant.

The paper is organized as follows: Section 2 briefly describes the plant model. In Section 3, several existing NN control schemes are surveyed with an emphasis on their drawbacks in their application to power plants control. The proposed NN control scheme is described in detail in Sections 4-7. Section 8 gives simulation results of the proposed methods. Finally, Section 9 is devoted to discussions and conclusions.

* Department of Electrical and Electronic Systems Engineering, Graduate Student (Omika Works, Hitachi LTD.)
** Department of Electrical and Electronic Systems Engineering

2. Plant model

The conventional controller configuration and plant model are described in this Section, which serves as a basis for the proposed NN controller. **Fig. 1** shows the outline of the conventional temperature control system. Fuel supply signal consists of the following three items.

(a) Statistic Feed-forward signal (SF) which corresponds to Mega Watt Demand (MWD).

(b) Proportional and Integral (PI) signal obtained by the feedback of the difference between set-point and measured temperature y(t).

(c) Transient Feed-forward signal (TF) which compensates the control lag of the PI control.

The TF and PI parts are replaced with an NN based controller, whose output signal is denoted as Dynamic Feed-forward Control signal (DFC). Our objective in this paper is to determine the DFC of **Fig. 1** by an NN model. Approximating the plant by linearization, plant model is represented by (1),

$$A(z^{-1})y(t)=z^{-d}B(z^{-1})u(t), \quad (1)$$
$$A(z^{-1})=1+a_1z^{-1}+a_2z^{-2}+\cdots+a_nz^{-n},$$
$$B(z^{-1})=b_0+b_1z^{-1}+b_2z^{-2}+\cdots+b_mz^{-m}.$$

Applying (1) to the temperature control of **Fig. 1**, the input variable, the output variable and the disturbance added to the input correspond to DFC (u(t)), temperature (y(t)) and MWD (w(t)), respectively. So the plant is rewritten by (2),

$$A(z^{-1})y(t)=z^{-d}B(z^{-1})u(t)+C(z^{-1})w(t), \quad (2)$$
$$A(z^{-1})=1+a_1z^{-1}+a_2z^{-2}+\cdots+a_nz^{-n},$$
$$B(z^{-1})=b_0+b_1z^{-1}+b_2z^{-2}+\cdots+b_mz^{-m},$$
$$C(z^{-1})=c_0+c_1z^{-1}+c_2z^{-2}+\cdots+c_kz^{-k},$$

w(t): disturbance⋯MWD,
u(t) : control law⋯fuel (DFC=PI=TF),
y(t) : controlled object⋯temperature,
(set-point—measured temperature)
d : dead time.

First it is necessary to determine the value of n, m, k in (2) before applying NN to the power plant. Response of temperature from fuel is approximately expressed by first order lag and dead time, and that from MWD by first order lag. So, the simplified model of the power plant is shown in **Fig. 2**.
Therefore the relation between y(t), u(t) and w(t) is given by

$$y=\left(\frac{K_1e^{-\tau s}}{1+T_1s}u-K_1K_2\left(\frac{1}{1+T_2s}-\frac{e^{-\tau s}}{1+T_1s}\right)w\right)\frac{1}{T_3s} \quad (3)$$

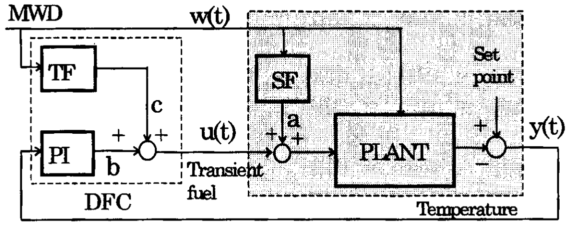


Fig. 1 Outline of main steam temperature control system

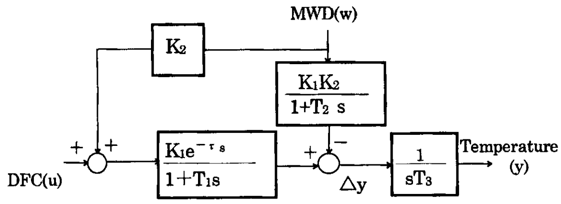


Fig. 2 Simplified model

By the z-transformation of (3), the following difference equation is obtained:

$$(1+a_1z^{-1}+a_2z^{-2}+a_3z^{-3})y(t)=z^{-d}(b_0+b_1z^{-1})u(t)$$
$$+(c_0+c_1z^{-1}++c_dz^{-d})w(t). \quad (4)$$

Applying Diophantine equation $(1=A(z^{-1})S(z^{-1})+z^{-d}R(z^{-1}))$ to (4), the following equation is obtained:

$$y(t+d)=\{A(z^{-1})S(z^{-1})+z^{-d}R(z^{-1})\}y(t+d)$$
$$=R(z^{-1})y(t)+B(z^{-1})S(z^{-1})u(t)$$
$$+C(z^{-1})S(z^{-1})w(t+d). \quad (5)$$

The order of S, R, BS, CS is d-1, 2, d, 2d-1, respectively, so R, BS and CS are given by

$$R(z^{-1})=a_0+a_1z^{-1}+a_2z^{-2}, \quad (6)$$
$$B(z^{-1})S(z^{-1})=b_0+b_1z^{-1}+\cdots+b_dz^{-d}, \quad (7)$$
$$C(z^{-1})S(z^{-1})=c_0+c_1z^{-1}++c_{2d-1}{}^{(2d-1)}. \quad (8)$$

3. Application of NN

An NN is a mean to describe the input/output relationship and the first step is to use an NN to identify the plant model (5). The plant model (5) should be represented by the Eq.(9) due to its nonlinearity,

$$y(t+d)=f[u(t),u(t-1),\cdots,u(t-d),w(t+d),$$
$$w(t+d-1),\cdots,w(t-d+1),$$
$$y(t),y(t-1),y(t-2)]. \quad (9)$$

Therefore utilizing an NN to identify the plant model is to construct Eq.(9) by the NN. From now on, time t is indicated by a suffix.

Some training and control methods have been already proposed, for example, feedback error learning (**Fig. 3(a)**)[3], special learning (**Fig. 3(b)**[2], generalized learning (**Fig. 3(c)**)[2], forward model learning (**Fig. 3(d)**), and forward and inverse model learning (**Fig. 3(e)**). Feedback error learning and special learning can be executed only when the plant model is known. This paper is focused on the problem where the plant is unknown. In the following, the learning and control methods are summarized which are appropriate for the case where the plant model is difficult to make, and the problems with these existing methods will be clarified.

A. Generalized learning

The input to the NN is the plant output, and the desired output is the control signal, i.e., by this learning method an inverse model of the plant is obtained. In this case, learning corresponds to determination of the following non-linear function that is the inverse of (9).

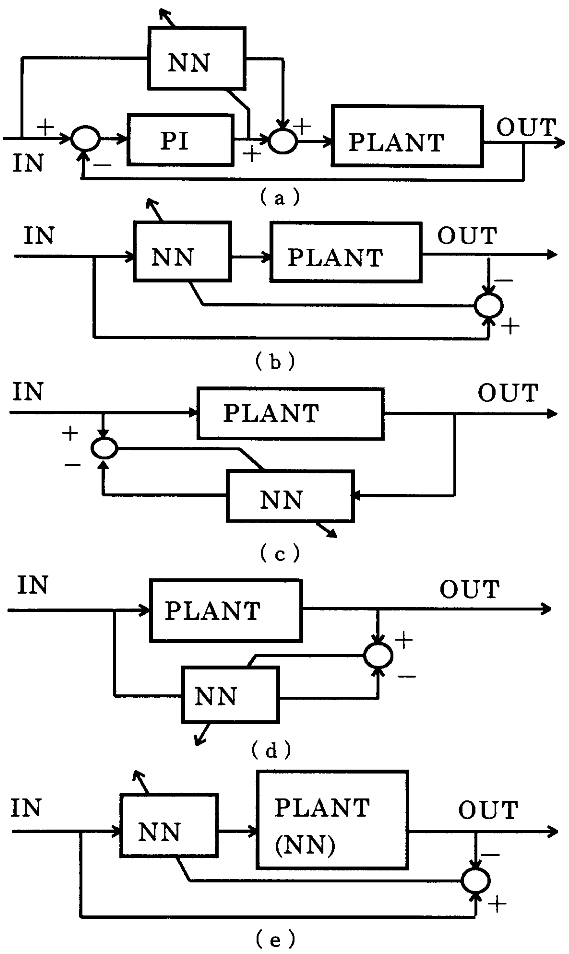


Fig. 3 Various training method

$$u_t=g[u_{t-1},u_{t-2},\cdots,u_{t-d},W_{t+d},W_{t+d-1},\cdots,W_{t-d+1},y_{t+d},y_t,y_{t-1},y_{t-2}] \quad (10)$$

After training the non-linear function, it is used in the on line control. But in this case function g contains errors (inevitable in NN), and so ut has some errors, where u_{t+1} is a function of u_t, in the same way u_{t+2} is a function of u_{t+1}, and so on. As the result, errors of control input u_t are accumulated. We have confirmed this by simulations.

B. Forward model learning

The input to the NN is the control signal, and the desired output is the plant output, i.e., this learning method corresponds to the creation of the forward model of the plant. Thus non-linear function (9) is obtained. Therefore control input u_t can be obtained by the inverse of NN. Although the learning error is smaller than that of generalized learning, it is difficult to obtain control input by inverting Eq.(9) because of its non-linearity.

C. Forward and inverse model learning

This learning method is shown in **Fig. 3(e)**. In this case, the NN is trained by the deviation between the target value and output of the plant. In this case Jacobian of the plant is necessary, but it is impossible to know the Jacobian if the plant is unknown. But when the plant is replaced by the plant model trained by forward model learning, the NN controller can be trained by using Jacobian calculated by the plant model. But in this case the NN is trained through two stages by which the error is accumulated and so good control performance is not expected. We have also confirmed this by simulations.

D. Determination of control law from forward model

Since forward model learning provides NN with small errors, an effective procedure for inverting them will be very useful in designing NN based controllers. Iterative inverse method (IIM)[5] was proposed for the calculation of control input from the forward model. The output of NN ; y_{t+d} is decided by input $y_t,\cdots,u_t,\cdots,w_{t+d}\cdots$ (sequential data of output, control input and disturbance) and weights W. W is decided by the forward model learning. Key point in IIM is to determine u_t by the same method as the determination of W with W being fixed. But this method has following three problems. i.e., iterative calculation is necessary, and we have to determine appropriate learning rate and iteration number which depend on the situations. There are some other methods as for the determination of control law from the forward model. e.g., simplex method, but it also has the same problems.

We propose two methods to solve these problems. The first one is the method named Jacobian method based on the assumption that the plant model is approximated linearly, and the second one called Hessian method uses Hessian in order to compute the control input.

4. Learning of NN

The feature of training data measured from the thermal power plant is that it has a long duration and changes very slowly as compared with those from the servo systems such as robots. (refer to **Fig. 12**). We have to choose appropriate method for NN learning. These are various alternatives concerning the learning of NN. (a) Structure of neural network: Layered network with external memory, Layered network with first order lag, Recurrent network. (b) Learning algorithm: BP (Back Propagation), BPTT (Back Propagation Trough Time), RTRL[6] (Real Time Recurrent Learning), RS[7] (Random Search). (c) Input method: Random input, Sequential input. (d) Updating timing of the weights: updating after presentation of all the data (1 cycle), updating per each data presentation. We studied what combination of them is appropriate for our purpose. Various combinations are evaluated by simulations. Simulation results show that the combination of (a) layered networks with external memory, (b) BP, (c) random input and (d) updating per each data presentation is the best.

We used the off line forward model learning to model the power plant. **Fig. 4** shows the configuration of NN.

5. Iterative inverse method

The IIM proposed by A. Linden et. al is summarized. **Fig. 5(a)** shows the forward model learning. Plant model is shown by $y_{t+d}=f(y_t, y_{t-1}, \cdots, u_t, u_{t-1}, \cdots, w_{t+d}, w_{t+d-1}, \cdots, W)$. Inputs of NN are $y_t, \cdots, u_t, \cdots, w_{t+d}, \cdots$, and NN output y_{nn} is compared to the supervised output y_{t+d}, and weights W is adjusted to minimize the difference between them as follows,

$$W^{(n)}=W^{(n-1)}-\eta'\cdot\frac{\partial(y_{t+d}-y_{nn})^2}{\partial W}, \tag{11}$$

where

η': learning rate of W.

Fig. 5(b) shows the principle of IIM, u_t can be calculated with W being fixed in almost the same way as the determination of W. Although W is trained off line in order to model the power plant, u_t is calculated on line to control the plant. The method of calculating ut in **Fig. 5(b)** is as follows.

The cost function is defined by Eq.(12),

$$J=\frac{1}{2}((d_{t+d}-y_{t+d})^2+\rho\cdot u_t^2), \tag{12}$$

where

ρ: Trade off coefficient between control accuracy and control energy. Inverse calculation of u_t is performed as follows in the same way as the adjusting of W,

$$\begin{aligned} u_t^{(n)} &= u_t^{(n-1)}-\eta\cdot\frac{\partial J}{\partial u_t}, \\ &= u_t^{(n-1)}-\eta\cdot\left((y_{t+d}-d_{t+d})\cdot\frac{\partial y_{t+d}}{\partial u_t}+\rho\cdot u_t^{(n-1)}\right), \end{aligned} \tag{13}$$

where

η: Learning rate of u_t.

Derivative $\partial y_{t+d}/\partial u_t$ in Eq.(13) is calculated as follows for our NN depicted in **Fig. 4**,

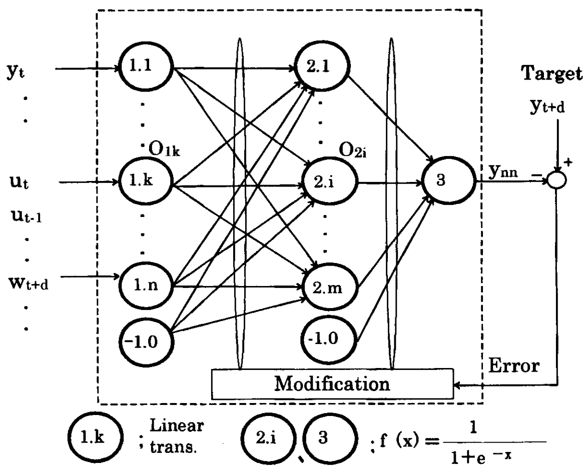


Fig. 4 Configuration of neural network

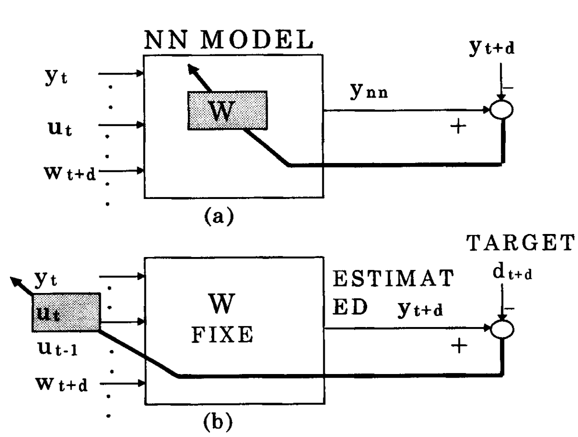


Fig. 5 Iterative inverse method

$$\frac{\partial y_{t+d}}{\partial u_t}=\sum_l \frac{\partial y_{t+d}}{\partial O_{2l}}\cdot\frac{\partial O_{2l}}{\partial u_t},$$
$$=\sum_l\{f'(net_3)\cdot W_l\cdot f'(net_{2l})\cdot W_{lk}\},$$
$$=f'(net_3)\cdot\sum_l W_l\cdot f'(net_{2l})'W_{lk}, \qquad (14)$$
$$net_{2l}=\sum_k O_{1k}\cdot W_{lk},\quad net_3=\sum_l O_{2l}\cdot W_l.$$

But, iterative equation (13) has the following essential problems when it is used on line.

(a) Iterative calculation is to be finished before measuring the new data.

(b) Determination of appropriate learning rate η is necessary.

(c) Determination of iteration number is needed.

When weight W is calculated off line according to Eq.(11) in order to model the plant, then trial-and-error search for good learning rate and iteration number is permissible. But calculation of u_t is executed on line, so the time expenditure due to the iterative calculation mentioned above is not preferable, especially the above trial-and-error process is not permissible.

6. Jacobian method

In this and the following sections, two new methods for calculating control input ut on line are presented in order to overcome those problems in the iterative inverse method mentioned in the previous section. The problems will be resolved if the inversion of the plant model can be performed either non-iteratively or with a fewer number of iterations. The first one is the Jacobian method that enables non-iteratine inversion based on the concept that the linearized parameters (6)−(8) are equal to the Jacobian of the NN obtained by the forward model learning. The other one is the Hessian method whose feature is the high speed of the calculation of u_t by use of the second order derivative.

Recall that the linearized plant was represented by Eq.(5)−(8). Because of linearity of the equations, inversion of Eq.(5) can be easily performed, i.e. the equation is readily solved for u_t. Therefore control input ut that minimizes the cost function Eq.(12) can be derived without any iterative calculation. However the parameters a_i, b_j, c_k in Eq.(6)−(8) depend on the operating point, and their on line estimation is required. Here, it is important to point out that a_i is equivalent to $\partial y_{t+d}/\partial y_{t-i}$, b_j is equal to $\partial y_{t+d}/\partial u_{t-j}$, and that c_k corresponds to $\partial y_{t+d}/\partial w_{t+d-k}$. Since we have an NN forward model of the plant, we can easily calculate the above Jacobians, $\partial y_{t+d}/\partial y_{t-i}$, $\partial y_{t+d}/\partial u_{t-j}$, and $\partial y_{t+d}/\partial w_{t+d-k}$. This is the basic idea underling our Jacobian method. The detailed procedure is described below.

A. Calculation of Jacobians

From **Fig. 4**, Jacobian is calculated as follows,

$$\frac{\partial y_{t+d}}{\partial x_k}=\sum_l \frac{\partial y_{t+d}}{\partial O_{2l}}\cdot\frac{\partial O_{2l}}{\partial x_k},$$
$$=\sum_l\{f'(net_3)\cdot W_l\cdot f'(net_{2l})\cdot W_{lk}\},$$
$$=f'(net_3)\cdot\sum_l W_l\cdot f'(net_{2l})\cdot W_{lk}. \qquad (15)$$

where x_k stands for any of y_t, y_{t-1}, $\cdots$, u_t, u_{t-1}, $\cdots$ w_{t+d}, w_{t+d-1}, $\cdots$, and

$$net_{2l}=\sum_k O_{1k}\cdot W_{lk},\quad net_3=\sum_l O_{2l}\cdot W_l.$$

B. Determination of control input

To get a control input that minimizes the cost function (12), we differentiate it with respect to u_t ant put it to zero :

$$\frac{\partial J}{\partial u_t}=(y_{t+d}-d_{t+d})\frac{\partial y_{t+d}}{\partial u_t}+\rho u_t \qquad (16)$$

Since $\partial y_{t+d}/\partial u_t=b_0$, substituting Eq.(5)−(8) into y_{t+d} in Eq.(16) and solving it for u_t, we get the following control law.

$$u_t=(-\sum_{i=0}^{2}a_i y_{t-i}+\sum_{i=1}^{d}b_i u_{t-i}+\sum_{i=0}^{2d-1}c_i w_{t+d}-d_{t+d})$$
$$/(b_0+\rho/b_0) \qquad (17)$$

If the dead time d is properly given, then b_0 is non-zero. Coefficient ρ in the cost function is usually set to be non-negative. Therefore the control law (17) is well defined. Here, the coefficients a_i, b_i and c_i are given by the Jacobians that are derived from the NN model. The control input u_t requires values of $w_{t+d}, w_{t+d-1}, \cdots, w_{t+1}$. Since these are future external signals and thus not available, their predicted values are used instead.

The key point in this paper is to employ control law (17), together with the jacobian (15), and they are calculated at each time step. **Fig. 6** shows the configuration of the Jacobian method. Dotted lines and real lines show off line learning and on line control, respectively.

7. Hessian method

We propose the Hessian method to improve the learning speed and to avoid the design parameters to be tuned appropriately depending on the problems. In the cost function (12), let the approximation value of u_t satisfying $\partial y_{t+d}/\partial u_t=0$ to be U_t, its current

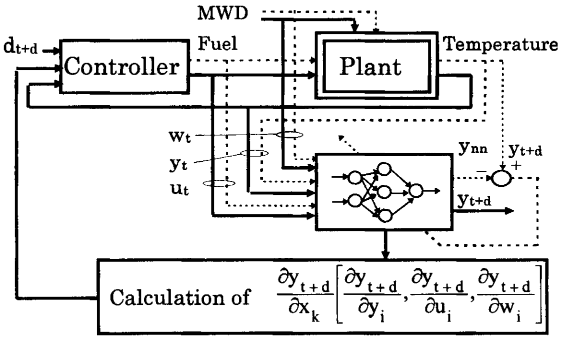


Fig. 6 Configuration of Jacobian method

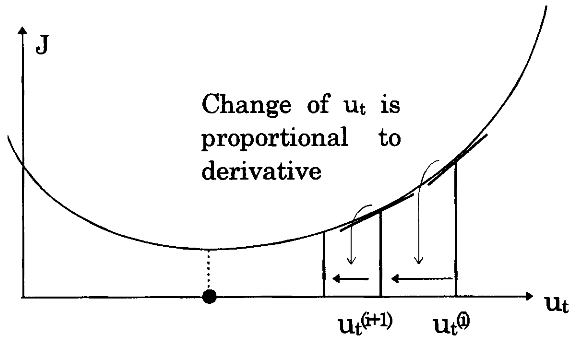


Fig. 7 Iterative inverse method

estimate to be U_t^0 and $U_t - U_t^0$ to be ε.

Expanding $\frac{\partial J}{\partial u_t}\Big| U_t$ by using Tailor expansion, we get

$$\frac{\partial J}{\partial u_t}\Big| U_t = \frac{\partial J}{\partial u_t} U_t^0 + \varepsilon \cdot \frac{\partial^2 J}{\partial u_t^2}\Big| U_t^0. \tag{18}$$

Putting $\frac{\partial J}{\partial u_t}\Big| U_t = 0$ gives

$$\varepsilon = -\frac{\partial J}{\partial u_t}\Big| U_t^0 / \frac{\partial_2 J}{\partial u_t^2}\Big| U_t^0. \tag{19}$$

Therefore u_t is given in the following,

$$U_t = U_t^0 + \varepsilon = U_t^0 - \frac{\partial J}{\partial u_t}\Big| U_t^0 / \frac{\partial^2 J}{\partial u_t^2}\Big| U_t^0, \tag{20}$$

where

$$\frac{\partial J}{\partial u_t} = (y_{t+d} - d_{t+d}) \cdot \frac{\partial y_{t+d}}{\partial u_t} + \rho \cdot u_t, \tag{21}$$

$$\frac{\partial^2 J}{\partial u_t^2} = \left(\frac{\partial y_{t+d}}{\partial u_t}\right)^2 + (y_{t+d} - d_{t+d}) \cdot \frac{\partial^2 y_{t+d}}{\partial u_t^2} + \rho. \tag{22}$$

Differentiating (14), we have

$$\frac{\partial^2 y_{t+d}}{\partial u_t^2} = f''(net_3) \cdot (\sum_l W_l \cdot f'(net_{2l}) \cdot (W_{lk})^2$$
$$+ f'(net_3) \cdot \sum_l W_l \cdot f''(net_{2l}) \cdot W_{lk}^2. \tag{23}$$

Putting $f(x) = \frac{1}{1+e^{-x}}$, its derivatives are given as

$$f' = f \cdot (1-f), \quad f'' = f \cdot (1-f) \cdot (1-2f). \tag{24}$$

So we can obtain the approximate value of u_t satisfying $\partial y_{t+d}/\partial u_t = 0$ from (20)−(24) and (14). Though the calculation is somehow complex, we can resolve the drawback of IIM in the on-line control. The ut obtained by Eq.(20) is assigned to U_t^0, and calculation of Eq.(20)−(24) and (14) is repeated, then accuracy of calculation is improved.

Hessian method can be considered from another point of view. Tailor expansion of the cost function in the neighborhood of U_t^0 is expressed by Eq.(25),

$$J = J\Big| U_t^0 + (u_t - U_t^0) \cdot \frac{\partial J}{\partial u_t}\Big| U_t^0 + \frac{(u_t - U_t^0)^2}{2} \cdot \frac{\partial^2 J}{\partial u_t^2}\Big| U_t^0$$
$$+ \text{(higher order terms)} \tag{25}$$

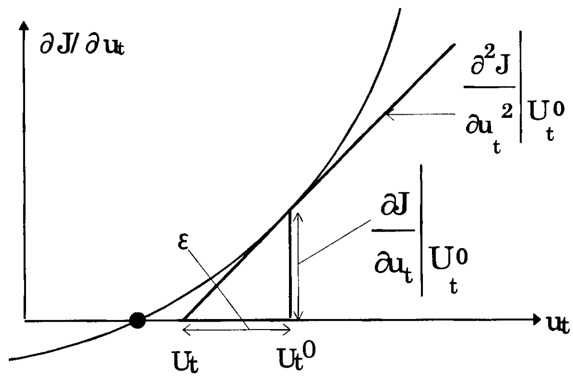


Fig. 8 Hessian method

Taking account of up to second order term of Eq. (25), J becomes a quadratic function of U_t. So minimizing J as the function of U_t, we can get (20). In other words IIM uses steepest descent method by first order derivative, while Hessian method uses second order derivative of the cost function. This corresponds to the fact that the learning of Gauss-Newton method is much faster than the usual back propagation method in the calculation of weights. It should be noted that weights' calculation time by Gauss-Newton method increases extraordinary with the number of weights. So it is not useful in the case of many weights. On the other hand as for ut, it is very useful because only one parameter U_t is to be calculated. Relation between IIM and Hessian method is shown in **Fig. 7** and 8. The u_t that minimizes the cost function is obtained by IIM iteratively, and in the Hessian method u_t is calculated that approximates $\partial y_{t+d}/\partial u_t = 0$ by using second order derivative. **Fig. 9** shows the configuration of Hessian method. Dotted lines and real lines show off line learning and on line control, respectively.

8. Simulation results

Simulation system consists of a controller model and a power plant model. The power plant model

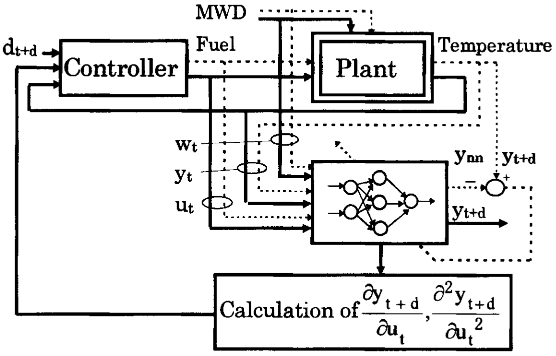


Fig. 9 Configuration of Hessian method

used in the simulations is constituted by the detailed simulator that represents accurately the dynamics of a typical power plant. **Fig. 10** shows the simulation flow diagram.

A. First step : Acquisition of plant data

Dead time of this plant simulator was about 50 sec., and considering that the sampling interval was 10 sec, we determined d=5. **Fig. 11** shows the training data. In **Fig. 11**, 0−125% etc. shows the full scale of each variable (the same convention is used in **Fig. 12−17**). This data shows the time sequence of y_t (temperature), u_t (control input) and w_t (MWD), when the load is changed, for example, 50%→ 70%→ 100%→ 70%→ 50%→ 100%→ 50% at load change rate 2%/min. or 5%/min. The sequential data ($y_t,\cdots,u_t,\cdots,w_{t+d},\cdots$) to be fed to (9) are obtained from these data.

B. Second step : Off line learning

The NN is trained off line by the data of the combination of input ($y_t,\cdots,u_t,\cdots,w_{t+d},\cdots$) and output y_{t+d}. **Fig. 12** shows training curves which illustrate the learning error (deviation between the desired value and measured value of the NN output) and some of weight values (training iterations : 200000). Error is about 7°C at the beginning of learning (somehow difficult to read it out from the chart), and decreases fast till 1000 iterations, after that decreases gradually, and final error is about 0.3°C and weights converge too.

C. Third step : On line control

We studied control performance using Jacobian of NN obtained in the second step and control law (15)−(17). **Fig. 13** shows the result of temperature control in the case of load change rate : 2%/min., where only on the coventinal PI control (TF=0 in **Fig. 1**) is used on condition that the load changes 50%→ 70%→ 100% → 70%→ 50%. Temperature deviation is large and becomes larger in the case of 5%/min. load change rate.

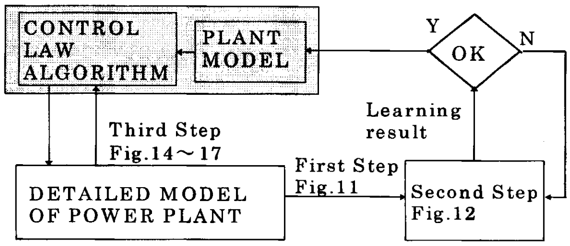
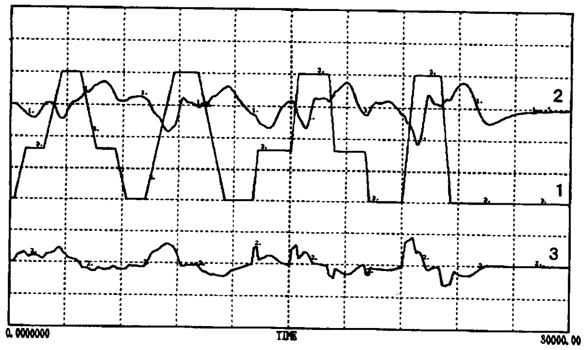
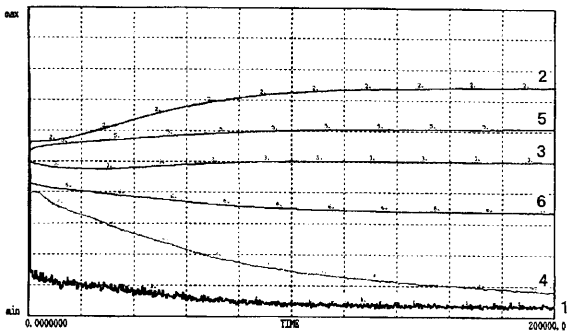


Fig. 10 Simulation flow diagram



1. MWD: 0−125% **2. TEMP. : 400−600℃**
3. DFC: -20−80% **Horizontal axis: 30000sec.**

Fig. 11 Examples of Learning Data



1. Deviation: 0−10℃ **2. Weight ($W_{1,1}$):-2−2**
3. Weight ($W_{1,15}$): -2−2 **4. Weight ($W_{15,1}$): -2−2**
5. Weight (W_1): -2−2 **6. Weight (W_{15}): -2−2**
Horizontal axis: 200000 iterations

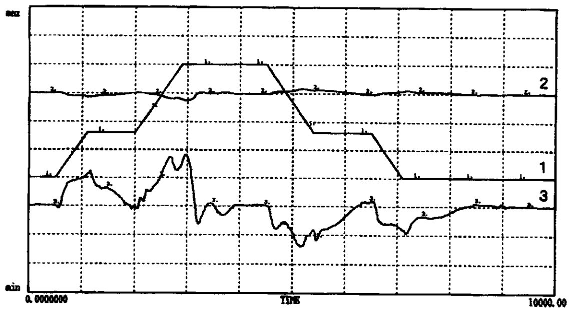
Fig. 12 Learning process of NN

The simulation results by IIM are shown in **Fig. 14** and 15. **Fig. 14** shows the result for the 2%/min. load up/down and NN learning with η=0.1 and the number of iteration=5. In this case the control input (fuel) and the plant output (temperature) are not preferable because of their hunting. **Fig. 15** shows the result with η=0.1 and iteration=100. This case is preferable compared to the case of iteration=5. In IIM there are two parameters i.e., learning rate and the number of iterations, so we have to determine them in order to get favorable control input. Moreover large η gives hunting in the neighborhood of optimal value, on the other hand small η needs large iterations. Therefore the determination of η is a somehow difficult task.

Fig. 16, 17 show the simulation results derived from Jacobian method, each corresponds to load change rate 2%/min. and free load swing. Control performance is improved very much as is evident from comparison between **Fig. 13** and **Fig. 16**. Hessian method was simulated in the same case and as good performance was obtained as Jacobian method. In the Hessian method, though u_t was calculated only once, temperature deviation and settling of control input were nearly equal to the case of the **Fig. 15**.

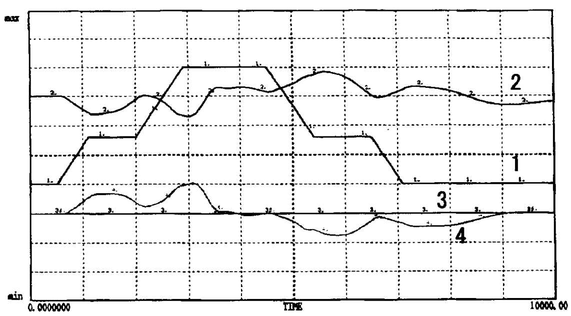
9. Conclusions

We have already shown that MRAC can not be applied to control the thermal power plant effectively because the persistently exciting conditions is not satisfied[8]. So, various kinds of neural networks have been studied in order to determine their best architecture and learning algorithm. Especially in this paper, two types of new control laws named Jacobian method and Hessian method are proposed based on the above architecture and learning algorithm. Those methods are advanced versions of iterative inverse method proposed by A. Linden et. al., and the first feature of those methods is the improvement of on line control, and the number of iterative calcula-



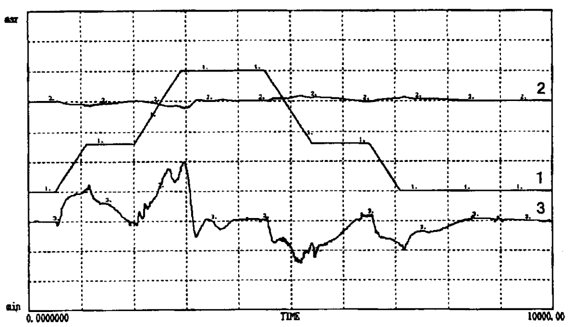
1. MWD: 0−125% 2. TEMP.: 400−600℃
3. DFC: -12−28 Horizontal axis: 10000sec.

Fig. 15 Load change by IIM (2) (2%/min.)



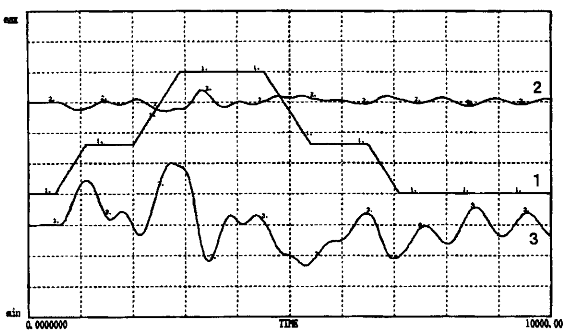
1. MWD: 0−125% 2. TEMP: 400−600℃
3. TF: -12−28 4. PI value: -12−28
Horizontal axis: 10000sec.

Fig. 13 Load change by PI control (2%/min.)



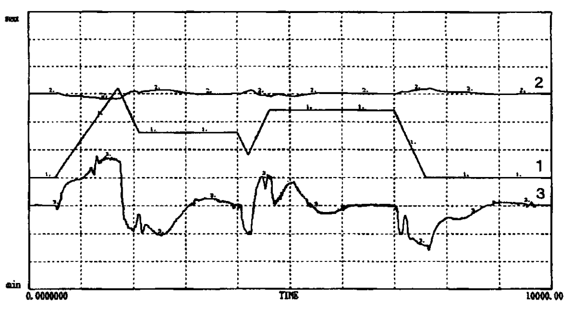
1. MWD: 0−125% 2. TEMP.: 400−600℃
3. DFC: -12−28 Horizontal axis: 10000sec.

Fig. 16 Load change by Jacobian method (2%/min.)



1. MWD: 0−125% 2. TEMP.: 400−600℃
3. DFC: -12−28 Horizontal axis:10000sec.

Fig. 14 Load change by IIM (1) (2%/min.)



1. MWD: 0−125% 2. TEMP.: 400−600℃
3. DFC: -12−28 Horizontal axis: 10000sec.

Fig. 17 Load change by Jacobian method (free load swing) tions for finding optimal control law is decreased dramatically. In other words, optimal characteristics ot iterative inverse method is obtained in a short computation time using those methods without any adjustment. From the simulation results usefulness and effectiveness of the Jacobian method and Hessian method are proved.

References

1) I. D. Landau, M. Tomitsuka ; Theory & Practice of Adaptive Control System. Oomusya, Tokyo, 1981

2) D. Psaltis, A. Siders, A. Yamamura ; A Multilayered Neural Network Controller. IEEE Control System Magazine, pp. 17~21, APRIL, 1988

3) H. Gomi, M. Kawato ; Learning Control of a Closed Loop System Using Feed back-Error -Learning. Vol. 4, No. 1, pp. 37~47, 1991 (in Japanese)

4) K. S. Narendra, K. Parthasarathy ; Identification and Control of Dynamical Systems Using Neural Networks. IEEE TRANSACTION ON NEURAL NETWORKS, VOL. 1, NO. 1, MARCH, pp. 4 ~27, 1990

5) A. Linden, K. Kindermann ; Inversion of multilayered nets. Int. Joint Cont. on Neural Networds (Washington D. C.), pp. 425~430, June 1989

6) R. J. Williams, D. zipser ; A Learning Algorithm for Continuously Running Fully Recurrent Neural Networks. Neural Computation, Vol. 1, pp. 270~280, 1989

7) J. Matyas ; Random Optimization. Automation and Remote Control, Vol. 26, No. 2, pp. 244~251, 1965

8) Y. Eki, K. Hirasawa M. Nomura ; Feed-forward Control of Thermal Power Plant Main Control System by a Combined Method of Neural Networks and MRAC. T. IEE Japan, Vol. 117-C, No. 3, 1997 (in Japanese)